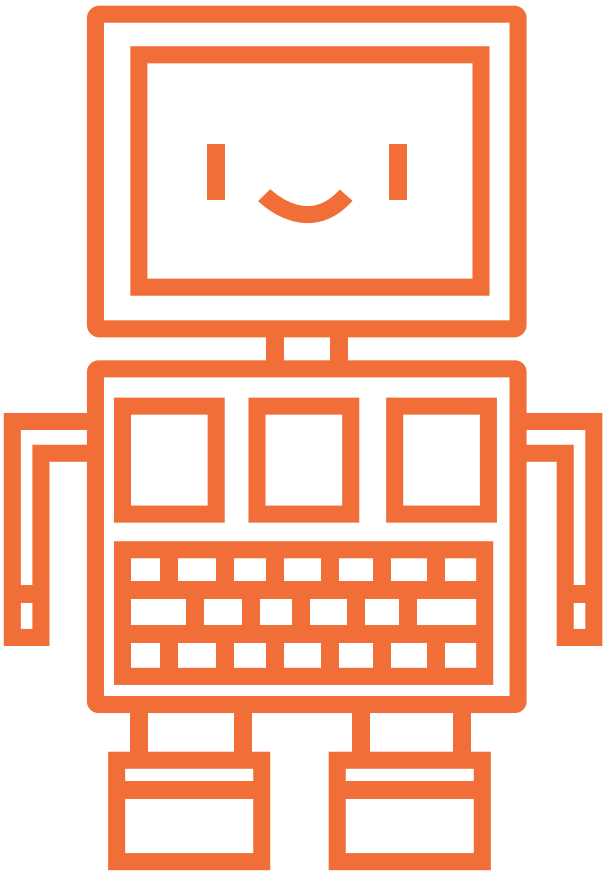
#MİLLİ TEKNOLOJİ HAMLESİ

DENEYAP TÜRKİYE

ROBOTİK VE KODLAMA

LİSE



TÜBİTAK

TÜRKİYE TEKNOLOJİ TAKIMI

DENEYAP

Teknoloji Atölyeleri

ROBOTİK VE KODLAMA

LİSE

Prof. Dr. İbrahim ÇETİN
Dr. Öğr. Üyesi Memet ÜÇGÜL
Prof. Dr. Ercan TOP
Prof. Dr. Erman YÜKSELTÜRK

TÜBİTAK

TÜBİTAK Deneyap Kitapları 4

ROBOTİK VE KODLAMA
LİSE

Prof. Dr. İbrahim ÇETİN
Dr. Öğr. Üyesi Memet ÜÇGÜL
Prof. Dr. Ercan TOP
Prof. Dr. Erman YÜKSELTÜRK

TÜBİTAK Deneyap Kitapları *DENEYAP TÜRKİYE*
Projesi kapsamında hazırlanmıştır.

ISBN 978-605-312-439-9
Yayıncı Sertifika No: 47703

TÜBİTAK Başkanı: Prof. Dr. Hasan MANDAL
Bilim ve Toplum Başkanı: Doç. Dr. Rukiye DİLLİ
Genel Yayın Yönetmeni: Fatma BAŞAR
Editör: Kübra BAL ÇETİNKAYA
Yardımcı Editör: İpek PİRPİROĞLU GENCER
Düzelti: Özlem KÖROĞLU, Adem ULUDAĞ, Şermin ASLAN, Mustafa ORHAN
Telif İşleri Sorumlusu: Öznur KILIÇKAYA

TÜBİTAK Bilim ve Toplum Başkanlığı
Tunus Caddesi No: 80 Kavaklıdere 06680 Ankara
Tel: (312) 298 96 50
e-posta: deneyap@tubitak.gov.tr
https://yayinlar.tubitak.gov.tr/deneyap-atolyesi

İçindekiler

Sunuş

İçinde bulunduğumuz çağda öğrenci rolleri değişmiştir. Öğrencilerden öğrendiği kavram, prosedür ve temel bilgileri başka bağlamlara aktarması, bu bilgileri kendi amaçları doğrultusunda yeniden inşa etmesi ve yeni bilgiler oluşturması beklenmektedir. Bu amaç doğrultusunda, bu kitapta öğrencilere bilgi işlemsel düşünme ve problem çözme gibi üst düzey düşünme becerileri kazandırmaya yönelik etkinlikler sunulmuştur. Günümüzde bilişim teknolojileri okur-yazarlığı sadece teknolojik araçları kullanmak olarak ifade edilmemektedir. Bilişim teknolojileri okur-yazarlığı daha kapsamlı olarak düşünülmektedir ve artık bu araçların üretimini de içermektedir. Birçok araştırmacı herkesin bilgisayar biliminin temel kavramlarını bilip kullanması gerektiğini söylemiştir. Bu bağlamda bilgi işlemsel düşünme 21. yüzyılda yaşayan bütün bireyler için temel bir beceri olarak kabul edilmektedir.

Robotlar ve robotik programlama, öğrencilerin bilgi işlemsel düşünme becerisini geliştirmek için uygun bir ortam sunmaktadır. Bu ortam uygun bir pedagoji ile birleştirildiğinde öğrencilerin problem çözme, yaratıcılık ve bilgi işlemsel düşünme gibi üst düzey bilişsel becerilerini geliştirmesine yardımcı olabilir. Bu kitapta öğretmenlere, öğrencilerin bahsi geçen becerilerini robotik programlama vasıtasıyla geliştirmeleri sürecinde rehberlik etmek hedeflenmektedir. Öğretmenlerin öğrencilerin gelişmesine yardımcı olabilmesi için bir öğretim modeli kullanması faydalı olabilir. Bu amaç doğrultusunda bu kitabın yazarları tarafından "Gözle, Uygula, Tasarla, Üret ve Değerlendir" öğrenme döngüsü geliştirilmiştir. GUTÜD öğrenme döngüsünde temel kavramların oluşturulmasında etkinlik ve öğretmen temelli göster ve uygula yaklaşımı ön plana çıkarılırken, öğrencilerin verilen temel bilgiler ile ileri seviye zihinsel faaliyetlerde bulunması için keşfe dayalı yaklaşımlar ön plana çıkarılmaya çalışılmıştır. Gözle bölümünde öğretmen bir robotik konusunu uygulamalı olarak (göstererek) anlatır. Uygula bölümünde öğretmen öğrencilerden Gözle bölümünde gösterilen uygulamaların aynısını / bir benzerini ister veya onlarla birlikte yapar. Tasarla bölümünde öğrencilerden kendilerine verilen karmaşık bir problemin çözümünü tasarlamaları istenir. Üret bölümünde öğrencilerden tasarladığı planları kullanarak problem için algoritmik bir çözüm üretmesi istenir. Değerlendir bölümünde hedef, öğrencinin, öğrenme sürecinde yaşadıkları ve öğrendikleri üzerine düşünmesini sağlamaktır. Bu öğretim döngüsünde öğrencilerin kendilerine verilen görevlerden uygun olanlarını eşli programlama grupları içerisinde yapmaları önerilmiştir. Eşli programlamada, iki öğrenci bir robot veya bilgisayar karşısında yan yana oturarak tasarım, algoritma, kod yazma veya hata ayıklama için iş birlikli çalışır.

Kitapta robot olarak LEGO firması tarafından üretilen Mindstorms Education EV3 seti kullanılmıştır. Bu setle oluşturulan robotları programlamak için öncelikle blok tabanlı programlama ortamı olan EV3 yazılımı ve ardından metin tabanlı programlama ortamı olan MicroPython kullanılmıştır. Bu sayede lise öğrencileri robotik kavramlarına görece kolay olan blok tabanlı bir ortamla giriş yaptıktan sonra kendilerini metin tabanlı bir ortamda geliştirmeleri mümkün olacaktır. Ayrıca etkinlikler boyunca kullanılmak üzere 70 cm x 100 cm boyutunda çift taraflı MAT (Çalışma Alanını) tasarlanmıştır. Etkinliklerin yapılabilmesi için bu MAT'ın kullanılması gerekmektedir. Elinde MAT bulunmayan öğretmenler her etkinlik için MAT'taki şekillerin bir benzerini oluşturarak kullanabilir.

Bu kitap lise seviyesinde robotik kodlama eğitimi vermek isteyen öğretmenler için hazırlanmıştır. Kitap içerisinde basitten karmaşığa gidecek şekilde spiral bir yaklaşım kullanılarak konular verilmiştir. Konuların tamamı GUTÜD döngüsü çerçevesinde oluşturulmuştur. Her ne kadar kitaptaki amaç bu olmasa da, kodlama hakkında ön bilgisi olmayan öğretmenler de konuyu öğrenmek için bu kitabı kullanabilir. Bunun yanında ileri seviye ortaokul ve lise öğrencileri de programlamaya ve robotik kavramlarına giriş yapmak için

bu kitabı kullanabilir. Ortaokul öğrencileri kitabın metin tabanlı programlama kısmında zorluk çekebilirler. Kendi başlarına bu kitabın metin tabanlı programlama kısmını çalıştığında zorluk yaşayan ortaokul öğrencilerinin bu kısmı kullanması önerilmez. Ortaokul öğrencilerinin bu konuları öğrenmekte güçlük çekmesi normaldir. Bu öğrenciler aceleci davranmamalı ve yeterli olgunluğa erişene kadar beklemelidirler.
Kitap 12 haftalık bir ders düşünülerek planlanmıştır. Her haftaya bir bölüm düşecek şekilde on iki haftalık içerik hazırlanmıştır. Öğrenciler 8.haftadan itibaren proje konuları ile ilgili yönlendirilebilir. Gerek duyulması durumunda projeler hazırlamak için ek sürelerde ayrılabilir. Bu sırayı izlemek ve kitabın tamamını kullanmak istemeyen öğretmenler kitap içerisindeki ilgili bölümleri kendi dersleri için kullanabilirler. Fakat bu kitap bir referans eser olarak değerlendirilmemelidir. Kitapta konular bütünlük teşkil edecek şekilde tasarlanmıştır.

Sevgili öğretmenler kitabın size ve ülkemiz öğrencilerine faydalı olması dileğiyle!

Öğretim Kılavuzu

ROBOTİK KODLAMA DERSİ ÖĞRETİM KILAVUZU

İçinde bulunduğumuz çağda öğrenme, öğrenenlerin sınıf içinde verilen kavram, prosedür ve temel bilgileri istenildiği zaman belirli bir ölçüde, sınav veya başka vasıtalar ile yeniden göstermesi olarak algılanmamaktadır. Amerikan Ulusal Araştırma Konseyi (2012), 21. yüzyıl öğrenenleri için bilişsel, kişisel, kişiler arası alan olmak üzere üç temel yeterlik başlığı belirlemiştir. Öğrenenlerin öğrendiği kavram, prosedür ve temel bilgileri başka bağlamlara aktarması ve bu bilgileri kendi amaçları doğrultusunda yeniden inşa etmesi gerekmektedir. Bu amaç doğrultusunda, öğrenenler problem çözme gibi üst düzey düşünme becerileri kazandırmaya yönelik etkinlikler sunulmalıdır.

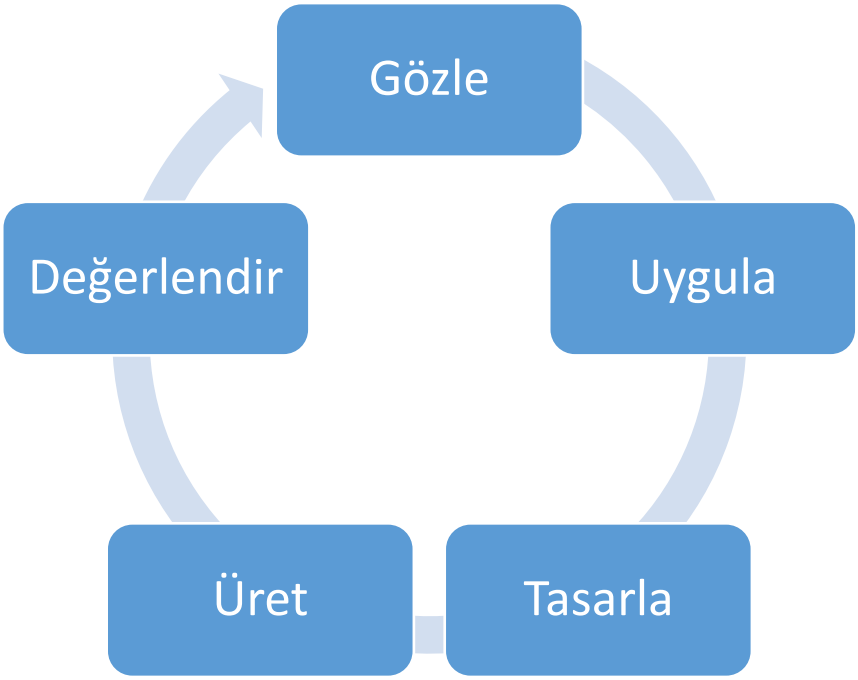
Problem çözme doğal, karmaşık ve anlamlı bir öğrenme/düşünme aktivitesi olarak tanımlanmaktadır. Problemlerin; otantik, öğrenenlerin yaşantılarıyla ilişkili, derin öğrenmeyi destekleyen ögeleri içermiş ve içerdiği bilgi ve becerilerin zorluğunun hiyerarşik olarak yapılandırılmış olması gerekmektedir (Jonassen, 2007). Öğrencileri gerçekçi ve bütünsel görevlere dâhil etmenin, onların uygun şema ve zihinsel modeller oluşturmasına yardımcı olacağına inanılmaktadır (Merrill, 2007). Problemler doğaları gereği yapısallık, karmaşıklık, dinamiklik ve meydana geldikleri bağlam açısından farklılıklar göstermektedir. Bu yüzden iyi oluşturulmuş içsel temsillerin, öğrenenlerin yeni edindikleri bilgi ve becerileri daha sonra farklı durumlarda uygulamalarını kolaylaştırdığına inanılmaktadır. Öğrenenler meslek yaşamlarında, daha önce hiç karşılaşmadıkları, farklı ve ileri seviye problemler ile karşılaşacakları için yaratıcı düşünmek durumundadırlar. Öğrenenler aynı zamanda nasıl öğrenmesi ve kendine nasıl yön çizmesi gerektiğini de öğrenmelidirler. Tüm bunlara ek olarak öğrenenler, iş birliğine dayalı çalışma ortamlarında nasıl çalışacaklarını ve nasıl iletişim kurmaları gerektiğini bilmelidirler.

Yaşadığımız çağda bilgi teknolojilerindeki değişimler ve bunun topluma ve günlük yaşama yansıması, mesleklerdeki değişimler ve fen bilimleri ile matematik alanlarındaki teknoloji kullanımı bireylerin bilişim teknolojileri okur-yazarı olmasını gerektirmektedir. Fakat bilişim teknolojileri okur-yazarlığı, sadece teknolojik araçları kullanmaktan ibaret değildir. Bu araçların üretimini de içermektedir. Wing (2006), herkesin bilgisayar biliminin temel kavramlarını bilip kullanması gerektiğini söylemiştir. Wing'e göre bilgi işlemsel düşünme, 21. yüzyılda yaşayan bütün bireyler için temel bir beceridir. Cuny, Snider ve Wing (2010) bilgi işlemsel düşünmeyi "Çözümlerin bir bilgi işleme birimi tarafından etkili şekilde yerine getirilebilecek formda sunulması amacıyla problemleri ve çözümleri formülleştirmeyi içeren düşünme süreci" olarak tanımlamışlardır. Bilgi işlemsel düşünme bir problem çözme sürecidir ve beş bileşeni içerir: (i) soyutlama, (ii) algoritmik düşünme, (iii) problemi bileşenlerine ayırma, (iv) çözümü verim açısından analiz etme ve (v) bir çözümü farklı bağlamlar için genelleme.

Bilgi işlemsel düşünme terimini ilk olarak kullanan ve programlamanın bütün öğrenciler tarafından öğrenilmesi gerektiğini bir teorik çerçeve ile ortaya koyan ilk kişi Seymour Papert'tır (Papert, 1991). Papert, Piaget ile çalışmış ve onun yapılandırmacılık (constructivism)

teorisinden etkilenmiştir. Papert (1993) yapılandırmacılığı temel alarak onun bir yorumu olan inşacılık (constructionism) yaklaşımını geliştirmiştir. Yapılandırmacılık, pedagojik anlamda basit bir şekilde ele alınacak olursa, öğrencinin öğrenme sürecindeki rolünü, pasif olarak bilgiyi alandan, aktif bir şekilde bilgiyi inşa eden olarak değiştirir. Yapılandırmacılık, felsefi anlamda basit bir şekilde ele alınacak olursa, dışsal objektif bilginin varlığını kabul etmez. Böyle bir bilgi olsa bile dışsal 'gerçeklik' öğrenen tarafından aynen kavranamaz. Öğrenen her hâlükârda kendi bilgisini inşa etmelidir. Yapılandırmacılık, öğretmeni bilgiyi bir zihinden diğerlerinin (öğrenenlerin) zihnine aktaran pedagojik bir araç olarak görmez. Öğretmen öğrenene buluş, keşif, inşa veya yeniden inşa sürecinde yardımcı olur ve onu pedagojik olarak destekler. Papert'ın inşacılığı, yapılandırmacılığın bu yorumlarını aynen alır ve bilginin inşa sürecinde öğrenenlerin somut bir ürün vasıtası ile soyut kavramsal bilgiye ulaşması gerektiğini savunur. Öğrenenler zihinlerindeki temeli kullanarak somut bir ürün geliştirirler. Bu ürünü geliştirirken soyut kavramları kullanmaları gerekir. Böylece soyut kavramlar somut ürünler vasıtası ile öğrenenin öğrenme alanına dâhil olur. Öğrenenin kendisi, diğer öğrenenler ve öğretmen bu somut ürün hakkında konuşarak öğrenenin soyut bilgiyi içselleştirilmesine yardımcı olabilir. Böylece öğrenenler sosyal bir ortamda somut ürünler vasıtası ile bilgilerini inşa ederler.

Bilgi işlemsel düşünmenin geliştirilmesi için çeşitli yaklaşımlar kullanılabilir. Bu yaklaşımlara örnek olarak; hiçbir bilgisayar gerektirmeyen bilgisayarsız bilgisayar bilimi, görsel programlama, metin tabanlı programlama ve robotik programlama verilebilir. Bu dersin içeriğini robotik programlama oluşturmaktadır. Bu derste öğrenenlerin bilgi işlemsel düşünme becerilerini robotik programlama vasıtasıyla geliştirmelerine yardımcı olmak hedeflenmektedir. Programlama başlı başına zor bir kavramdır ve çeşitli zihinsel zorluklar içerir (Çetin, 2013). Burada bu zorluklara değinilmeyecektir fakat bahsi geçen zorluklardan dolayı programlama öğretilirken temel pedagojik çerçevenin dikkatlice belirlenmesi gerekir. Bu dersin tasarımında temel olarak 21. yüzyıl yeterlikleri ve programlama eğitiminin alan yazını kullanılacaktır. Programlama eğitimiyle ilgili alan yazında, temel iki eğilim bulunmaktadır. Bunlardan ilki, dersi öğrencinin keşfedeceği şekilde tasarlayıp öğrenci merkezli bir yaklaşım sunmaktır. Bu yaklaşıma göre öğrenci süreçte aktif rol üstlenir. Öğretmen bilgiyi aktarmaktan ziyade, öğrencinin keşfetme veya oluşturma sürecinde ona yardım eden pozisyondadır. İkinci yaklaşım ise dersi öğretmenin anlatımı üzerine kurar. Burada öğretmen asıl sorumluluğu alır ve öğrencilere konuyu aktarmayı hedefler. Öğrencinin keşfetmesini temel alan ilk yaklaşımda, öğrenciler belirli bilgi işlemsel problemlerde yetkin olsalar da temel algoritmik kavramlarda ve dolayısıyla bunların farklı bağlamlara aktarımında sıkıntılar yaşamaktadır (Mayer, 2004). Bunun karşısında bilgi aktarımını temel alan bilgi işlemsel öğretim yaklaşımları ise çağımızın öğrenenlerini yetiştirmek için yetersiz kalmaktadır (Papert, 1991). Bu yüzden, alan yazında tartışıldığı üzere (Brown ve Campione, 1994; Grover, Pea ve Cooper, 2015; Mayer, 2004), bu derste temel kavramların oluşturulmasında aktarım yaklaşımı ön plana çıkarılırken, öğrencilerin verilen temel ile ileri seviye zihinsel faaliyetlerde bulunması ve farklı bağlamlara aktarılabilir bir bilgi oluşturabilmesi için keşfe dayalı yaklaşımlar ön plana çıkarılacaktır. Bu amaç doğrultusunda detayları aşağıda verilecek olan "gözle, uygula, tasarla, üret ve değerlendir" öğrenme döngüsü kullanılmıştır.



Şekil 1. Öğrenme Döngüsü

GÖZLE - UYGULA - TASARLA - ÜRET - DEĞERLENDİR ÖĞRENME DÖNGÜSÜ

Gözle: Bu bölüm iki kısımdan oluşur. Birinci kısımda öğretmen öğrencilerin geçmiş bilgilerini aktive etmek ve onların dikkat ve motivasyonlarını sağlamak ile görevlidir. Bunun için bir önceki derste yapılan etkinlikleri / çalışmaları kısaca özetleyebileceği gibi günlük yaşamdan ya da bilim insanlarının yaşantılarından ilgi çekici örnekler de kullanabilir, örneğin otomasyona giriş yapmak için El-Cezeri'nin hayatından ve ürettiği cihazlardan bahsedilebilir. Bu bölümün ikinci kısmında ise öğretmen bir robotik konusunu uygulamalı olarak (göstererek) anlatır. Bu kısımda öğretmen daha aktiftir. Uygulamayı yaparken öğrencilere sorular sorabilir ve öğrencilerin sorularını yanıtlayabilir.

Uygula: Bu bölümde öğretmen öğrenenlerden bir önceki bölümde gösterilen uygulamaların aynısını / bir benzerini ister veya onlarla birlikte yapar. Örneğin, hazırlanmış bir robotun ileri doğru belirli bir hızda ve miktarda hareket ettirilmesini gösteren öğretmen, öğrenenlerden robotu belirli bir hızda ve miktarda ters yönde hareket ettirmesini isteyebilir.

Tasarla: Bu bölümde öğrenciler daha aktif rol üstlenir. Öğretmen rehber pozisyonundadır. Öğretmen, öğrencilere takıldıkları noktalarda destek olacaktır. Öğrencilerin etkinlikten kopup, motivasyonlarının düşmesine izin vermemeye çalışacaktır. Fakat öğretmenin sağladığı destek gereğinden fazla da olmamalıdır. Bu bölümde öğretmen tarafından öğrenenlere bir problem verilir. Öğrenenlerden öncelikle bu problemin çözümünü tasarlamaları istenir. Tasarlama aşamasında, öğrenenler temel itibarıyla bilinenler ile istenenler arasındaki bağı kurarak bir plan

üreteceklerdir. Bu amaçla, öğrenenler (Bilgi işlemsel düşünme becerisi bileşenlerini kullanırlar):

- Bilinenleri ve istenilenleri ayrı ayrı belirler,
- İstenilenleri alt bileşenlere ayrılabiliyorsa ayırır,
- Bu problem veya alt problemlerin aynılarına veya benzerlerine daha önceden çözüm ürettiyse bunları tanımlar,
- Bu problemler veya alt problemlerin çözümü için bilgisayar biliminde daha önceden belirlenmiş çözümlerin (sıralama ve arama algoritmaları gibi) olup olmadığını belirler,
- Daha önceki adımlarda ortaya koyduklarını kullanarak bir çözüm planı üretirler.

Bu aşamadaki önemli nokta, öğrenenlerin çözüme doğrudan başlaması yerine önce çözüm hakkında düşünmesi ve bir çözüm planı üretmesidir. Öğrenenler her defasında yukarıda bahsi geçen beş adımı yapmak istemeyebilir veya bu adımları yaparken sıkılabilirler. Bu durumlarda, öğrenenlerin adımları bire bir uygulaması yönünde onları zorlamak yerine onlardan problemi doğrudan çözmeye başlamadan önce problem hakkında düşünmesi ve planlama yapması istenebilir.

Üret: Bu bölümde öğrenciler aktif rol üstlenir. Öğretmen rehber pozisyonundadır. Öğretmen öğrencilere takıldıkları noktalarda destek olacaktır. Destek, Vygotsky'nin (1978) Yakınsal Gelişim Alanı (Zone of Proximal Development) kavramında belirttiği gibi bireyin yardım ile gerçekleştirebileceği, ancak henüz bağımsız olarak yapamayacağı bir durum oluştuğunda sağlanmalıdır. Üret aşamasında, öğrenenlerden bir önceki adımda tasarladığı planı kullanarak probleme algoritmik bir çözüm üretmesi istenir. Öğrenenler bilgisayar ve robot başında çalışarak gerekli donanımsal ve yazılımsal çözümleri geliştirirler.

Değerlendir: Buradaki değerlendirme ile anlatılmak istenen doğrudan öğrencinin başarısının notlandırılması değildir. Temel hedef, öğrenenin, öğrenme sürecinde yaşadıkları ve öğrendikleri üzerine düşünmesini sağlamaktır. Bu sayede, öğrenen problem çözme süreci, dersin konusu ve kendisi ile ilgili gözlemler yaparak yeni öğrenmeler, kendini değerlendirme ve planlama açısından fırsatlar elde edecektir. Öğrenenlerden şu soruları yanıtlamaları istenebilir:

- Verilen problemi tanımlayınız (problemi kendi cümleleri ile ifade etme).
- Problemin çözümü için hangi stratejileri kullandınız ve neden bu stratejileri seçtiniz?
- Problemi çözerken ne gibi sıkıntılar yaşadınız ve bunların üstesinden gelmek için neler yaptınız?
- Kullandığınız yöntemler, bu sıkıntıları gidermekte başarılı oldu mu?
- Grup arkadaşınızla ihtilafa düştüğünüz durumlar oldu mu ve bunların üstesinden gelmek için neler yaptınız?
- Grup arkadaşınızdan ne öğrendiniz?

Öğrenenlerin buradaki soruların tamamına cevap vermesi beklenmemektedir. Bu sorulardan, verilen etkinlikten elde ettikleri deneyimlere bağlı olarak, kendilerine uyanları cevaplayabilirler. Cevaplar, öğrenenlerden yazılı olarak da istenebilir. Fakat öğrenenler, belirli

bir süre sonra sürekli aynı sorulara cevap vermekten sıkılabilir/sıkılacaktır. Bu durumda, belirli derslerin sonunda öğrenenlerden dersteki deneyimlerini genel olarak değerlendirmeleri istenebilir. Bunun yanında, değerlendirme sürecini daha etkili hâle getirmek amacıyla çeşitli etkinlikler yapılabilir. Örneğin, öğrenenler bir halka şeklinde dizilir. Öğretmen elinde bulunan topu (veya benzeri bir cismi) öğrenenlerden birine atarak onun yukarıdaki sorulardan bir veya birkaçına yanıt vermesini ya da dersteki deneyimlerini genel olarak değerlendirmesini isteyebilir. Cevap veren öğrenen, elindeki topu bir başka arkadaşına atar ve arkadaşı da bir veya birkaç soruyu yanıtlar. Etkinliklerin ilerleyen zamanlarında, top yerine rastgele açıyla kendi etrafında dönecek şekilde programlanmış olan robot, öğrenci halkasının ortasında bir sonraki öğrencinin seçimini yapabilir. Bu durum, değerlendirme doyum noktasına ulaşana kadar, öğrencileri sıkmadan, devam ettirilebilir.

EŞLİ PROGRAMLAMA

Öğrenenler kendilerine verilen görevlerden uygun olanlarını (yukarıda bahsedilen uygula ve üret adımları) eşli programlama grupları içerisinde yapacaklardır. Eşli programlamada iki öğrenen, bir robot veya bilgisayar karşısında yan yana oturarak tasarım, algoritma, kod yazma veya hata ayıklama için iş birlikli çalışır. Eşli programlama eşli araba yarışlarına benzetilebilir. Eşli araba yarışlarında sürücü arabayı kullanırken kılavuz sürücüye yön tayini konusunda yardımcı olur. Eşli programlamada bilgisayarı veya robotu kullanan kişiye sürücü denir. Sürücünün görevi robotun istenenleri gerçekleştirmesi için tasarımı ve kodlamayı yapmaktır. Sürücünün yanındaki kişiye kılavuz denilir. Kılavuz, bilgisayarı veya robotu kullanmaz. Kılavuzun görevi çıkan problemler veya ana problem için çözüm üretmek, bu süreçte ortaya çıkan hataları belirlemek ve sürücünün nasıl çalıştığını değerlendirmektir. Eşli programlamada, araba yarışından farklı olarak, sürücü ve kılavuz düzenli olarak yer değiştirir. Öğrenenlerin her iki görevden de öğreneceği şeyler vardır. Bu yüzden rol değiştirme çok önemlidir. Öğretmen öğrenenlerin periyodik olarak görev değiştirmesini sağlamalıdır.

- İki öğrenen bir bilgisayar veya robot karşısına oturur,
- Bir öğrenen kodları yazarken diğeri kodları değerlendirir ve öneride bulunur,
- Belirli zaman aralıklarıyla öğrenenler rollerini değiştirir.

Eşli programlamada bir diğer önemli nokta, hangi iki öğrencinin eş olarak atanacağıdır. Burada, iyi bilen ve az bilen gibi, farklı bilgi veya beceri gruplarından öğrenenlerin bir araya getirilmesi iyi bilenin az bilene öğretmesi açısından faydalı olarak görülebilir. Fakat pratikte bu fayda gerçekleşmemektedir. Bundan ziyade, iyi bilen bir müddet sonra diğerini kendine ayak bağı olarak görme eğilimi ve görece daha az bilen de iyi bilen ile iletişim kurmakta sıkıntı yaşayıp etkinliklerden kopma eğilimi göstermektedir. Benzer bilgi ve beceri düzeyinde olan gruplar daha verimli çalışmaktadır. Bu yüzden benzer bilgi ve beceri düzeyinde olan öğrenenlerin eş olarak belirlenmesi önemlidir. Bazı durumlarda eş olarak tayin edilen öğrenenler birbirleriyle ciddi anlaşmazlıklar ve çatışmalar yaşayabilirler. Çatışma yaşayan öğrencilerin aynı grupta kalması sağlıklı olmayacaktır. Bu durumu fark eden öğretmen, çatışma yaşayan eşleri farklı öğrenenlerle yeniden eşlemelidir.

GRUPLAR ARASI İLETİŞİM VE ROBOT YARIŞLARI

Etkinlikler boyunca gruplar arası bilgi alışverişine izin verilmelidir. Gruplar arasında paylaşımcı bir ortam oluşturulmalıdır. Fakat bu paylaşım, komple bir çözümün paylaşımı şeklinde olmamalıdır. Öğrenenler; çözüm yolları, stratejiler ve eksik bilgiler gibi konular için paylaşım yapabilirler. Fakat verilen bir problemin bütün çözümü paylaşılmamalıdır. Bu konu hakkında öğretmen, öğrenenleri daha önceden bilgilendirmelidir ve onların ne tür bir paylaşım içerisinde olduğunu takip ederek gereğinden fazla olan paylaşımları engellemelidir.

Robot yarışları, öğrenenleri güdüleyen önemli bir öğretme/öğrenme yöntemidir. Bu eğitim programı içerisinde zaman zaman gruplar arası yarışlar düzenlenmiştir. Fakat bu yarışlar, sınıfın paylaşımcı ortamını zedeleyecek içerikte uygulanmamalıdır.

ROBOTİK KİTLER VE PROGRAMLAMA ORTAMI

Lego Mindstorms Education EV3

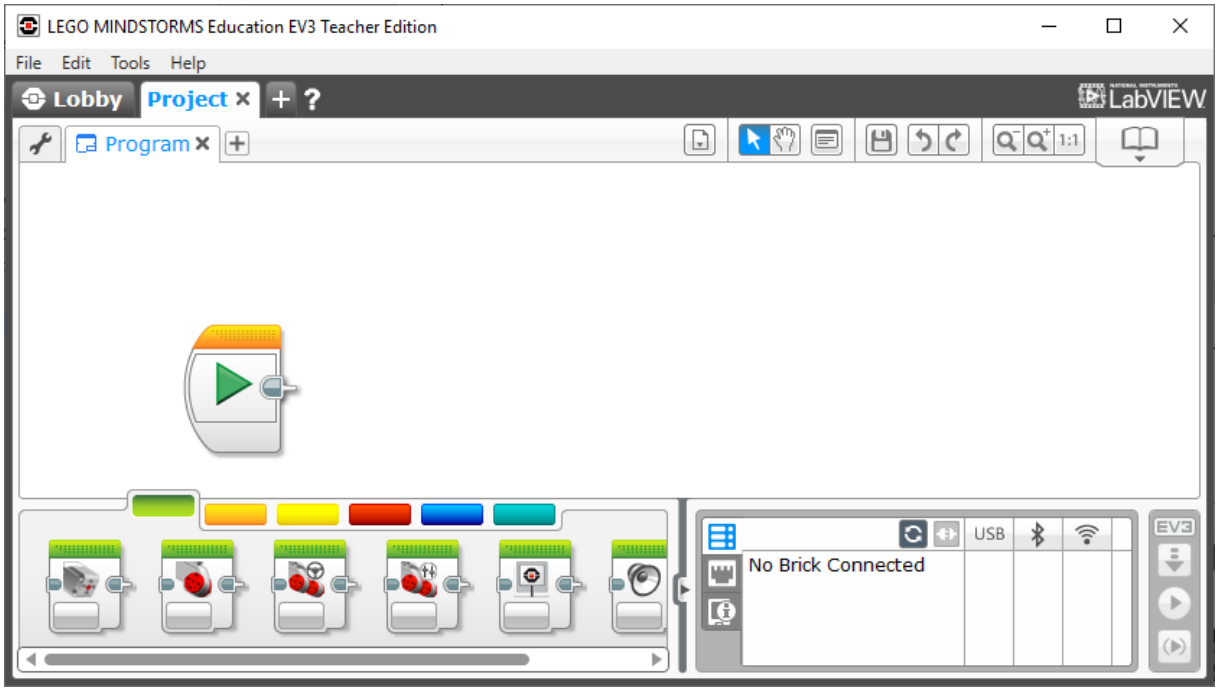
Bu eğitimde LEGO firmasının 2013 yılında piyasaya sürdüğü Mindstorms Education EV3 sürümü kullanılacaktır. 541 parçadan oluşan bu setin içinde EV3 programlanabilir tuğla, renk sensörü, ultrasonik sensör, dokunma sensörü ve jiroskop sensör dâhil birçok teknik parça mevcuttur. Temel setle ve temel set + ekstra parça seti ile inşa edebilen farklı robot tasarımları mümkündür. Aşağıdaki resimde EV3 robot seti görülmektedir.

Resim 1. LEGO® MINDSTORMS® Education EV3 robot seti

EV3 Yazılımı

EV3 yazılımı robotları programlamak için kullanılan ücretsiz bir blok tabanlı programlama ortamıdır. Öğrenciler blok hâlinde var olan veya kendilerinin oluşturdukları komutları sürükle bırak yöntemiyle taşıyarak programları oluştururlar. Komutlar bloklar hâlinde bulunduğundan dolayı, öğrencilerin söz dizimini öğrenmek için fazladan zaman ayırmaları gerekmez ve

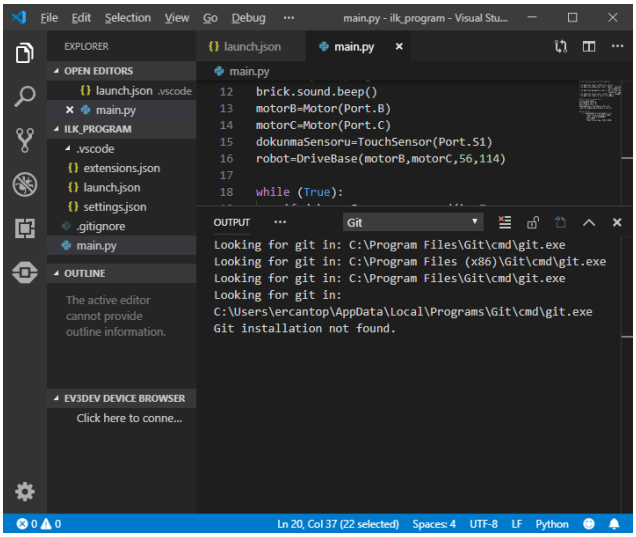
program yazarken hata yapma ihtimalleri ortadan kaldırılır. Bu sayede öğrenciler zihinsel kaynaklarını problem çözmeye yönlendirebilirler. EV3 yazılımının programlama arayüzü aşağıdaki resimde görüldüğü gibidir.



Resim 2. EV3 Yazılımı

MicroPython

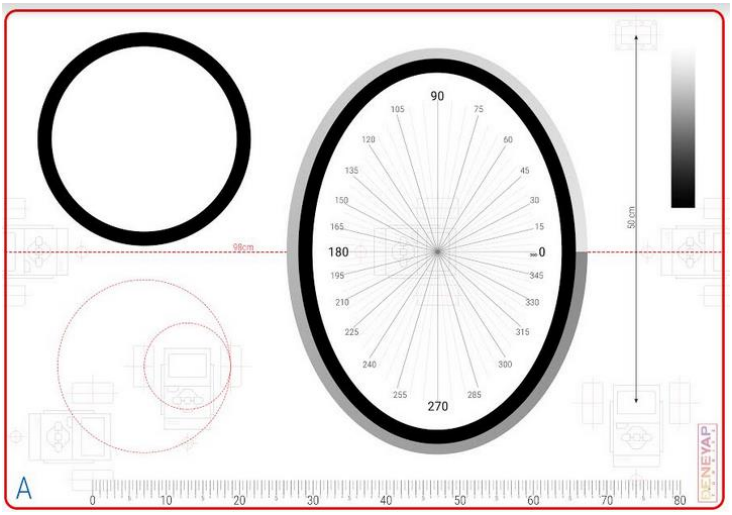
"MicroPython" sayesinde "Python" programlama dili ile Lego Mindstorms EV3 robotu programlanabilir. Burada metin tabanlı programlama yaklaşımı kullanılmaktadır. MicroPython'da öğrenciler yazacakları her bir komutu en ince detayına kadar tam olarak bilmek zorundadırlar. Bu yüzden robotik programlamaya giriş yapmak EV3 yazılımına göre daha zor olmaktadır. Fakat metin tabanlı programlama yaklaşımı profesyonel programcılar tarafından yıllardır kullanılmaktadır. Metin tabanlı programlama yaklaşımına alıştıktan sonra karmaşık / ileri seviye programların oluşturulması bu ortamlarda daha kolaydır. Aşağıdaki resimde görülen "Visual Studio Code" programlama editörü kullanılarak MicroPython'da programlar yazılabilir.



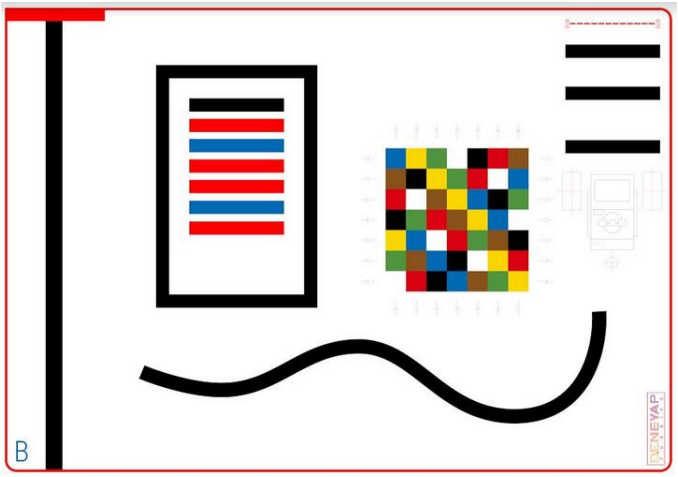
Resim 3. Visual Studio Code Ortamı

Eğitimde Kullanılacak Mat (Çalışma Alanı)

Öğrenciler etkinlik alanına geldiklerinde aşağıda resimleri verilen 70 cm x 100 cm boyutundaki çift taraflı matı (çalışma alanını) masalarının üzerinde göreceklerdir. Etkinlikler boyunca bu mat kullanılacaktır.



Resim 4. Mat Ön Yüz



Resim 5. Mat Arka Yüz

Kaynakça

Amerikan Ulusal Araştırma Konseyi (National Research Council). (2012). *Education for Life and Work: Developing Transferable Knowledge and Skills in the 21st Century*. Washington, DC: The National Academies Press. https://doi.org/10.17226/13398.

Brown, A. L., & Campione, J. C. (1994). Guided discovery in a community of learners. In K. McGilly (Ed.), Classroom lessons: Integrating cognitive theory and classroom practice (pp. 229–272). Cambridge: MIT Press.

Çetin, İ. (2013). Visualization: a tool for enhancing students' concept images of basic object-oriented concepts. *Computer Science Education*, *23*(1), 1–23.

Grover, S., Pea, R., & Cooper, S. (2015). Designing for deeper learning in a blended computer science course for middle school students. *Computer Science Education*, *25*(2), 199–237.

Jonassen, D. (2011). Designing for problem solving. In R. Reiser & J. Dempsey (Eds.), *Trends and Issues in Instructional Design and Technology*, (pp. 64–74). Boston, MA: Pearson Education.

Mayer, R. E. (2004). Should there be a three-strikes rule against pure discovery learning? *American psychologist*, *59*(1), 14–19.

Merrill, M. D. (2007). A task-centered instructional strategy. *Journal of Research on Technology in Education*, *40*(1), 5–22.

Papert, S. (1991). Situating constructionism. In I. Harel & S. Papert (Eds.), *Constructionism* (pp. 1–11). NJ: Ablex.

Papert, S. (1993). *Mindstorms: children, computers, and powerful ideas* (2nd ed.). New York: Basic Books.

Vygotsky, L. (1978). Interaction between learning and development. *Readings on the Development of Children*, *23*(3), 34–41.

1. Hafta: Robotlarla Algoritma ve Hareket

Ön Bilgi:

- Temel bilgisayar ve donanım bilgisi

Haftanın Kazanımları:

- Öğrenciler robot kavramını temel düzeyde açıklar.
- Öğrenciler algoritma kavramını açıklar.
- Öğrenciler basit bir algoritma örneği oluşturur.
- Öğrenciler robot setini tanır.
- Öğrenciler robot setiyle robotik tasarım yapar.
- Öğrenciler robotik setini programlamak için grafik arayüzünü (EV3 yazılımı) kullanır.
- Öğrenciler alan, uzunluk, tur, açı ve çap (yarıçap) kavramlarını robotun hareketini sağlarken kullanır.
- Öğrenciler robotun hareket etmesi için gerekli programlama adımlarını oluşturur.

Haftanın Amacı:

Bu haftanın amacı eğitim süresince kullanılacak robot kavramının tüm öğrenciler tarafından doğru ve benzer şekilde anlaşılmasını sağlamak ve robot seti ile robot setinin programlanacağı EV3 yazılımının grafiksel arayüzünü tanıtmaktır. Ayrıca, robot seti kullanılarak mekanik robot tasarımları hazırlandıktan sonra programlamaya giriş yapılarak öğrencilerin algoritmanın ne olduğunu tanımlaması ve basit bir algoritma oluşturabilmesi de hedeflenmektedir. Öğrencilere robotlarının belirli bir mesafeyi alabilmesi için gerekli programlama adımlarını öğretmek ve öğrencilerin ölçme, uzunluk, denklem, çap ve çevre konularını verilen problemin çözümünde kullanmalarını sağlamak bu haftanın bir diğer amacıdır.

Kullanılacak Malzemeler:

Robot seti, bilgisayar, mat (çalışma alanı)

Haftanın İşlenişi:

Gözle: Robot üzerine tartışma, robotun bileşenlerini tanıma ve robot setinin programlanacağı yazılımın grafiksel arayüzünü kullanma

Uygula: Algoritma oyunları, robot yapma ve robotları programlama

Tasarla: İstenilen robotu tasarlama

Üret: Verilen görevleri programlama

Değerlendir: Haftanın içeriği ile ilgili yansıtma etkinliği

1. GÖZLE VE UYGULA

1.1. Gözle: Robot Kavramı Üzerine Tartışma

Rehber öğretmen, öğrencilerin "robot" kavramı üzerinde tartışmalarını sağlar.

- Robot nedir?
- Robotların özellikleri nelerdir?
- Robotlar nerelerde kullanılır?
- Uzaktan kumandalı bir oyuncak ile robot arasında ne gibi farklılıklar vardır?
- Robotlar hangi parçalardan oluşur?
- Mekanik nedir?
- Program/algoritma nedir?

gibi sorular yönelterek tartışmayı yönetir. Cevapları öğrencilerin bulması sağlanır. Rehber öğretmen; robot, algoritma ve program kavramları için aşağıdaki tanımları dikkate alarak öğrencileri yönlendirebilir veya gerektiği yerde (öğrenciler kavramı yanlış tanımladıklarında veya doğru olandan çok farklı bir tanım yaptıklarında) bu tanımları doğrudan kendisi yapabilir.

Robot, Türk Dil Kurumunun hazırladığı *Türkçe Sözlük*'te "belirli bir işi yerine getirmek için manyetizma ile kendisine çeşitli işler yaptırılabilen otomatik araç" şeklinde tanımlanır. *Oxford İngilizce Sözlük*'te ise "özellikle bir bilgisayar tarafından programlanan, karmaşık bir dizi işlemi otomatik olarak gerçekleştirebilen makine" şeklinde tanımlanır. Her iki tanımda da vurgulanan nokta, robotların bir görevi otomatik olarak yerine getirmesidir. Eğer bir makinenin hareketleri bir insan tarafından kontrol ediliyorsa bu makineye robot denilemez. Ama makine çevresini algılayabiliyor ve buna göre hareket ediyorsa, örneğin bir engeli algılayıp yön değiştirebiliyorsa, bu makine robot olarak adlandırılır (Kelly, 2010).

Robotlar çevrelerini sensörleri aracılığıyla algılar. Robotlar kullanılan sensöre bağlı olarak; nesnenin uzaklığı, nesnenin rengi, ışık miktarı, ses şiddeti, nem oranı gibi birçok çevresel veriyi algılayıp işlemcileri ile yorumlayabilir ve programları dâhilinde bu verilere tepkide bulunabilirler. Bu durum göz önüne alındığında robotun tanımı, "çevresini algılayabilen ve algıladığını yorumlayarak bağımsız tepkiler verebilen makine" şeklinde yapılabilir. Bir makinenin robot olup olmadığını değerlendirmek için aşağıdaki özellikleri taşıyıp taşımadığına bakılabilir:

- Sensör: Çevresini sensörü/sensörleri aracılığıyla algılayabiliyor mu?
- Program: Bir programı var mı? Programı ile verileri işleyip karar verebiliyor mu?
- Eylem: Programı sayesinde farklı eylemler gerçekleştirebiliyor mu?

Algoritma: Sonlu bir süre içerisinde istenilen bir sonucu elde etmek için açık ve net bir şekilde tanımlanmış ve sıralanmış adımlar bütünüdür. Aslında her öğrenci matematik derslerinde algoritmaları kullanmıştır. Örneğin alt alta eldeli toplama bir algoritmadır. Bu işlemin sonucunu bulmak için uygulanacak adımlar kişiden kişiye değişmeyecek netliktedir. Bu adımlar verildiğinde alt alta eldeli toplamayı uygulayan herkes aynı sonuca ulaşır. Peki, alt alta eldeli toplamanın adımları nelerdir? Burada öğrencilerden temel olarak şunları söylemeleri beklenir:

(i) sayıların alt alta yazılması

(ii) kolonlardaki sayıların toplanması ve sonuç olarak yazılması

(iii) elde değeri oluştuğunda bunun bir soldaki kolonun toplama değerine eklenmesi

(iv) ilk üç kuralın toplama işlemi tamamlanana kadar devam ettirilmesi.

Program: Temel olarak bir algoritmanın bilgisayar veya robot için uygulanmasıdır. Programlar bilgisayara veya robota yapmaları gereken şeyleri (algoritmanın adımlarını) bir programlama dili vasıtası ile aktarırlar. Öğrenciler de bu derste robotları kodlamak için bir programlama dili kullanacaktır. Programlamanın en önemli noktasıysa problemin çözümü için algoritma geliştirmektir. İzlenecek yol doğru belirlenebilirse yani algoritma doğru hazırlanabilirse bilgisayar ve robotlar programlama dilleri aracılığıyla problemi çözebilir. Bu bilgilendirmeler yapıldıktan sonra algoritma uygulamaları bütün öğrencilerle beraber yapılır.

1.2. Uygula: Algoritma Öğreniyorum

Sınıftaki öğrencilere üzerinde 1 ile 24 arasındaki sayıların yazılı olduğu kartlar verilir. Öğrenci sayısı 24'ten azsa 1, 2, 3, 4, 6, 8, 12, 24 sayılarını mutlaka içeren, istenilen sayıda kart dağıtılır. Öğrenci sayısı 24 ise her öğrenciye bir kart verilir. Öğrenci sayısı 24'ten fazlaysa birden fazla öğrenciye aynı sayı gelecek şekilde kartlar ayarlanır. Dağıtılan kartlar arasında yine 1, 2, 3, 4, 6, 8, 12, 24 sayılarının hepsi bulunmalıdır. Öğrenciler kartlarındaki sayıyı saklayarak grup hâlinde bekler. Öğrencilerden biri "robot" olarak görevlendirilir ve sadece bu "robot-öğrenciye" diğerleri duymayacak şekilde aşağıdaki algoritma verilir:

1) Her bir öğrenciye git,
   a) Sayısını sor,
   b) Eğer öğrencinin kartındaki sayı 24'ü tam bölen bir sayıysa bu öğrenciyi gruptan ayır ve tahtanın önüne getir.

Robot-öğrenci diğer öğrencilere sadece kartlarındaki sayıyı sormalıdır ve ipucu oluşturacak cümleler kurmamalıdır. Robot-öğrencinin işlemini tamamlamasının ardından rehber öğretmen, diğer öğrencilerden, robot-öğrencinin tahtaya çıkardığı öğrencilerin sayılarına bakarak hangi işlemin yapılmış olabileceğini söylemelerini ister. Öğrenciler yapılan işlemi bildikten sonra onlardan bu işlemin algoritmasını yazmaları istenir.

1.3. Uygula: Çarpım Eşleri Algoritması

Öğrencilerden, çarpımları 24 olan çarpım eşlerinin (1-24, 2-12, 3-8, 4-6) bulunması için bir algoritma yazmaları istenir. Öğrenciler buldukları algoritmaları sınıfça tartışabilirler.

Son olarak algoritmanın bir görevi gerçekleştirmesi için net bir şekilde tanımlanmış ve sıralanmış adımlardan oluşması gerektiği vurgulanır. Eğer görevleri açıkça tanımlanmadıysa veya adımların sıralaması uygun değilse robotun veya bilgisayarın istenilen görevi yerine getiremeyeceği belirtilir.

1.4. Gözle: Robot Yapımı

Robot kavramıyla ilgili temel tartışmalar yapılıp algoritma uygulaması tamamlandıktan sonra rehber öğretmen öğrencilere LEGO MINDSTORMS EV3 setini tanıtır. Robotu yaparken dikkat edilmesi gereken noktaları belirtir. Parçaların sağlam olduğundan fakat düşme, çarpma gibi

durumlar sonucu kırılıp yıpranabileceklerinden bahseder. Parçaların kaybolmaması için robot seti ile gelen veya var olan tasnif kutusunun kullanılması önerir. Daha sonra öğrencilere robot setleriyle temel tasarım (driving base) robotunu yapmaları için yeteri kadar süre verilir. Rehber öğretmen, robotlar oluşturulduktan, sonra daha öncesinde hazırladığı bir programı robot üzerinde çalıştırarak gösterir. Burada amaç, robota basit bir görev yaptırarak öğrencilerin robota karşı ilgilerini artırmaktır.

Not

Temel tasarım (driving base) yönergesine aşağıdaki yollarla ulaşılabilir:
i) https://le-www-live-s.legocdn.com/sc/media/lessons/mindstorms-ev3/building-instructions/ev3-rem-driving-base-79bebfc16bd491186ea9c9069842155e.pdf
ii) EV3 yazılımı > Lobby > Building Instructions > Building Ideas > Driving Base
iii) Robot setiyle birlikte gelen kitapçığa bakılabilir.

1.5. Gözle: EV3 Yazılımı

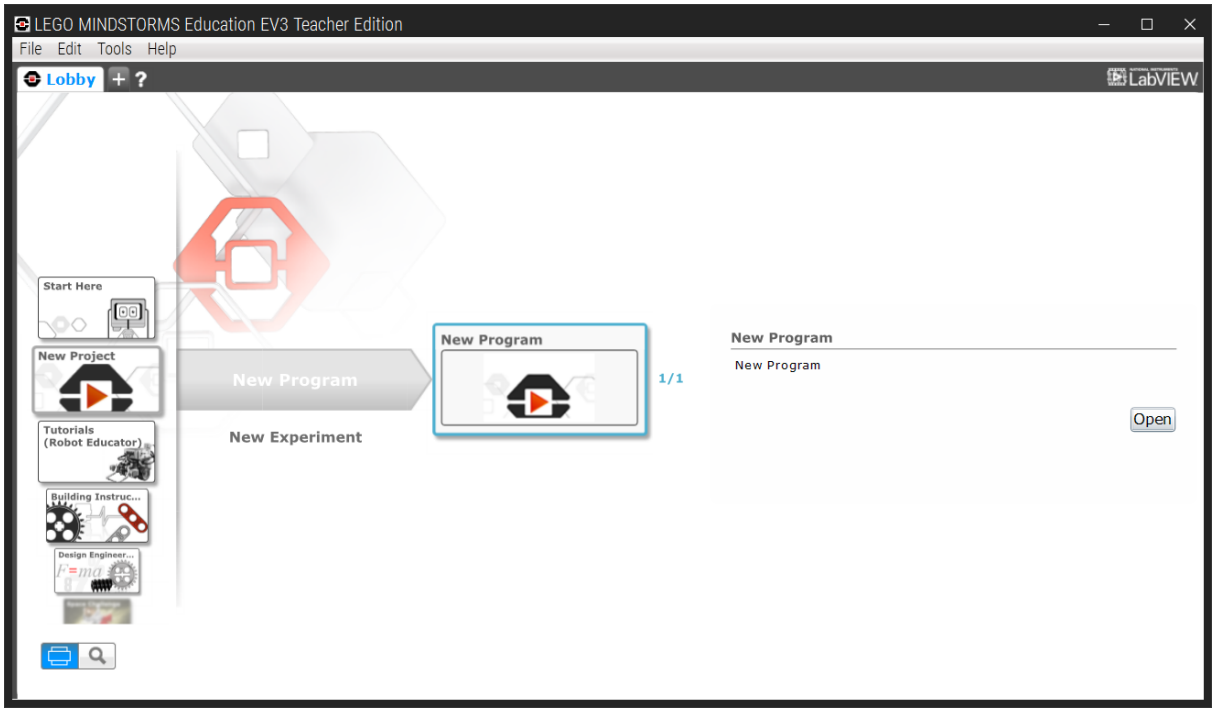
Robot setiyle yeteri kadar zaman geçirildikten sonra hazır bir robot üzerinde robot setinin parçaları gösterilerek öğrencilere parçalar hakkında temel bilgiler verilir. Daha sonra EV3 robot setlerinin programlanması için gerekli yazılımın bilgisayarlarda kurulu olup olmadığı kontrol edilir. LEGO MINDSTORMS EV3 robot setinin "akıllı tuğla (intelligent brick)" üzerinden de programlanabildiğinden, ancak robotların programlanması için eğitim süresince bilgisayarların kullanılacağından bahsedilir.

Not

Lego Mindstorms Education EV3 yazılımına https://education.lego.com/en-us/downloads/retiredproducts/mindstorms-ev3-lab/software adresinden uygun işletim sistemi seçilerek veya Windows işletim sistemi için aşağıdaki adresten direkt ulaşılabilir.
https://le-www-live-s.legocdn.com/downloads/LME-EV3/LME-EV3_Full-setup_1.4.5_en-US_WIN32.exe

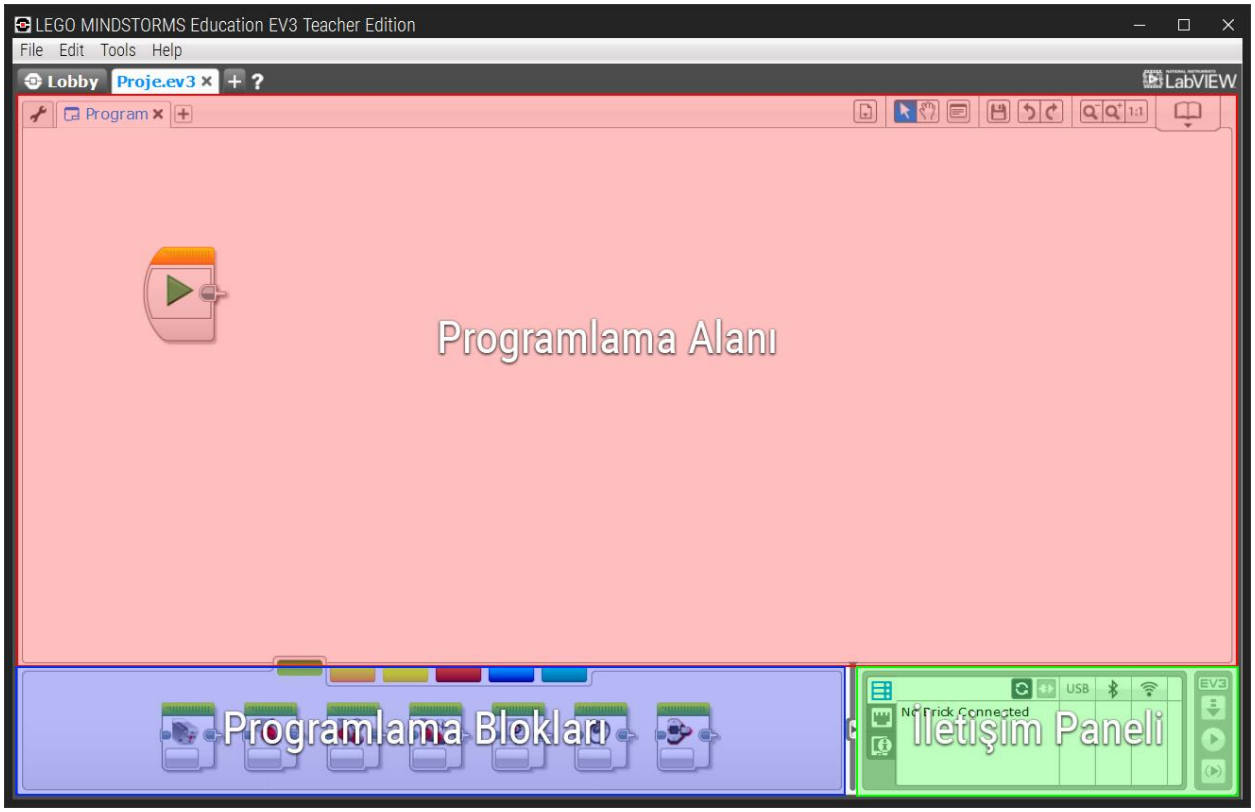
Robot setinin programlanacağı yazılımın grafiksel arayüzü (EV3 yazılımı) açılır ve arayüzde neler olduğu gösterilir. Grafik arayüzü ile akıllı tuğla arasındaki bağlantının USB veya Bluetooth ile gerçekleştirilebileceği vurgulanır. Önce kablo bağlantısı ile sonra da Bluetooth ile bağlantıların nasıl yapılacağı anlatılır. Daha sonra EV3 yazılımının grafiksel arayüzünün bölümleri tanıtılır.

Aşağıdaki resimde görüldüğü gibi EV3 yazılımı *Lobby* ekranı ile açılır. Bu ekranda sol kısımdaki etkinlik sekmeleri kullanılarak yeni bir program penceresi açılır (New Project > New Program > Open).



Resim 6. EV3 Yazılımı Lobby Ekranı

EV3 yazılımının arayüzü temel olarak üç kısımdan oluşur. Robotun programlanabilmesi için program bloklarının programlama alanının içerisine sürüklenip bırakılması gerekir. Programlama alanında, varsayılan olarak aşağıdaki resimde görüldüğü gibi bir başla bloğu (yeşil üçgen) bulunur. Programlama alanında birbirine yaklaştırılan bloklar birbirine yapışır. Böylece sürükle-bırak yaklaşımı ile istenilen programın oluşturulması sağlanır.



Resim 7. EV3 Yazılımı Programlama Arayüzü

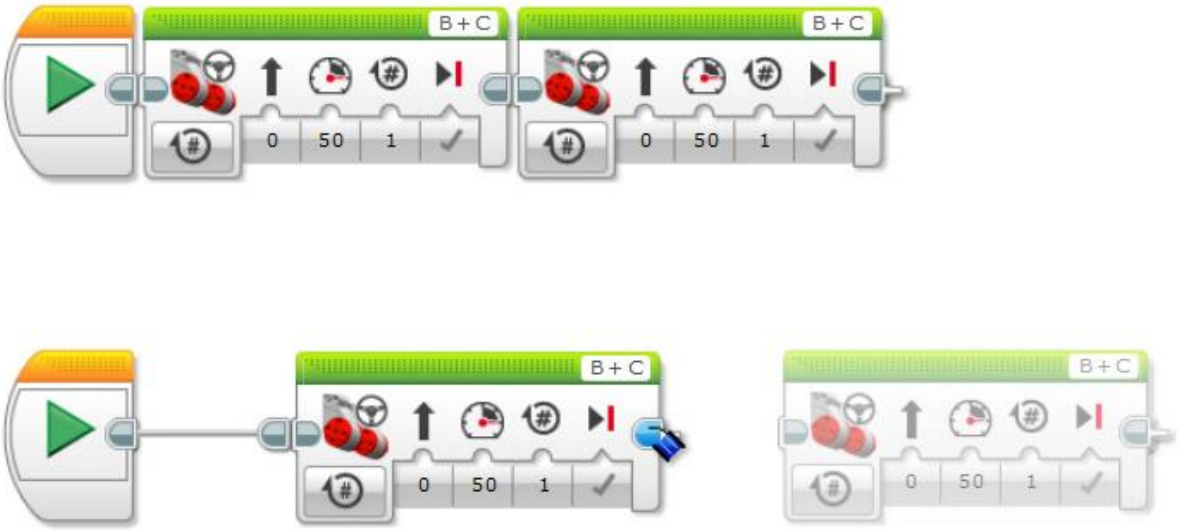
Yukarıdaki resimde görüldüğü gibi programlama blokları bölümü farklı renklerdeki altı sekmeden oluşur:

- Yeşil Sekme (Action/Eylem Sekmesi): Robotun hareket etmesini sağlayan motor kontrol blokları, görüntü, ses ve tuğla ışığı blokları bu sekmede yer alır.
- Turuncu Sekme (Flow Control/Akış Kontrol Sekmesi): Yeni bir başla program bloğunun yanı sıra bekle, döngü, anahtar ve döngü kesme blokları bu sekmede bulunur.
- Sarı Sekme (Sensor/Sensör Sekmesi): Robotun sensörlerinden ışık, uzaklık, motorların dönme açısı gibi verilerin alınmasını sağlayan program blokları bu sekmede yer alır.
- Kırmızı Sekme (Data Operations/Veri İşlemleri): Veriler üzerinde matematiksel ve mantıksal işlemlerin gerçekleştirilebileceği blokların ve değişken, sabit ve rastgele sayı üreten blokların bulunduğu sekmedir.
- Mavi Sekme (Advanced/İleri Düzey Sekmesi): İleri düzey programlama bloklarının, örneğin dosya erişimi, veri kaydı, Bluetooth bağlantısı gibi blokların bulunduğu sekmedir.
- Turkuaz Sekme (My Blocks/Benim Bloklarım Sekmesi): Program birçok bloktan oluşuyorsa belirli görevleri yerine getiren program blokları bir blok olarak tanımlanabilir. Tanımlanan bu blok gruplarına benim bloklarım sekmesinden ulaşılabilir.

İletişim paneli üzerinden USB, Bluetooth veya kablosuz bağlantı ile bağlı olan akıllı tuğlanın durumu ve senfsörlerden alınan anlık veriler takip edilebilir. Ayrıca programlama alanında hazırlanmış programın tüm bloklarının ya da seçili blokların akıllı tuğlaya yüklenip çalıştırılmasını sağlayan düğmeler de bu bölümde yer alır.

1.6. Gözle: Programlama Alanının Kullanılması

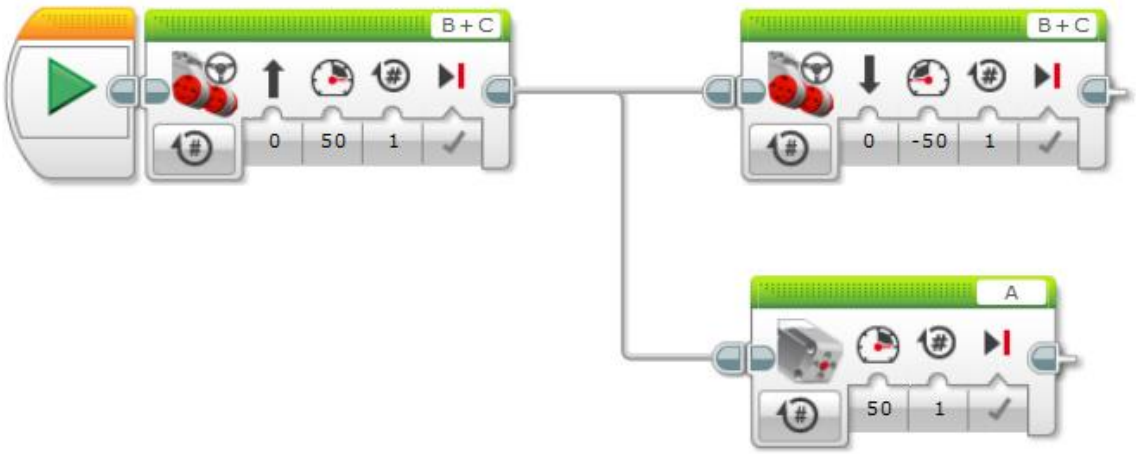
Robot, istenilen program bloğunun programlama alanına sürüklenip bırakılması ve blokların parametrelerinin değiştirilmesi ile programlanır. Program, blokların programlama alanında yer alan başla bloğuna tren vagonları gibi birbiri ardına eklenmesiyle oluşturulabilir. Program çalıştırıldığında, robot, başla bloğundan başlayarak program bloklarında belirlenen görevleri sırasıyla gerçekleştirecektir. Aşağıdaki resimde blokların yerleştirilmesi görülmektedir.



Resim 8. Blokların Yerleştirilmesi

Başla bloğu ile herhangi bir bağlantısı bulunmayan bloklar daha soluk gösterilir ve programın çalıştırılması sırasında programa dâhil edilmez. Bloklar arasındaki bağlantı, blokların yan yana eklenmesi ile sağlanabileceği gibi bir bloğun bittiği noktadan diğer bloğa bir kablo bağlantısı yapılarak da gerçekleştirilebilir. Bunun için fare bloğun bittiği noktanın üzerine getirilmeli ve imlecin değişmesinin ardından sürüklenip istenilen noktada bırakılmalıdır.

Robotun birden fazla işlemi aynı anda gerçekleştirmesi isteniyorsa, programlama alanında birden fazla başla bloğu kullanılarak, iki farklı programın eş zamanlı çalışması sağlanabilir. Ayrıca kablo yöntemi kullanılarak programların istenilen noktasında paralel işlemler başlatılabilir (Paralel işlemler ilerleyen haftalarda daha detaylı işlenecektir.). Paralel işlemler aşağıdaki resimde görülmektedir.



Resim 9. Paralel İşlemler

1.7. Gözle: Robotun Hareket Ettirilmesi

EV3 robot setinde robotu hareket ettiren iki çeşit motor bulunur. Bunlar aşağıdaki resimde görüldüğü gibi büyük (large) ve orta (medium) motor olarak adlandırılır. Set içerisinde iki büyük motor ve bir orta motor mevcuttur. Büyük motor robotu hareket ettirmek için kullanılır ve orta motora göre daha güçlüdür. Fakat orta motor büyük motordan daha hızlı çalışır. Her iki motorun içerisinde devir sensörü bulunur. Devir sensörü kullanılarak motorun devir sayısı bulunabilir. Motorun devir sayısı açı ile hesaplanabileceği gibi bir tam devir temel alınarak kaç devir döndüğü de bulunabilir.

Resim 10. Büyük ve Orta Motor

Not

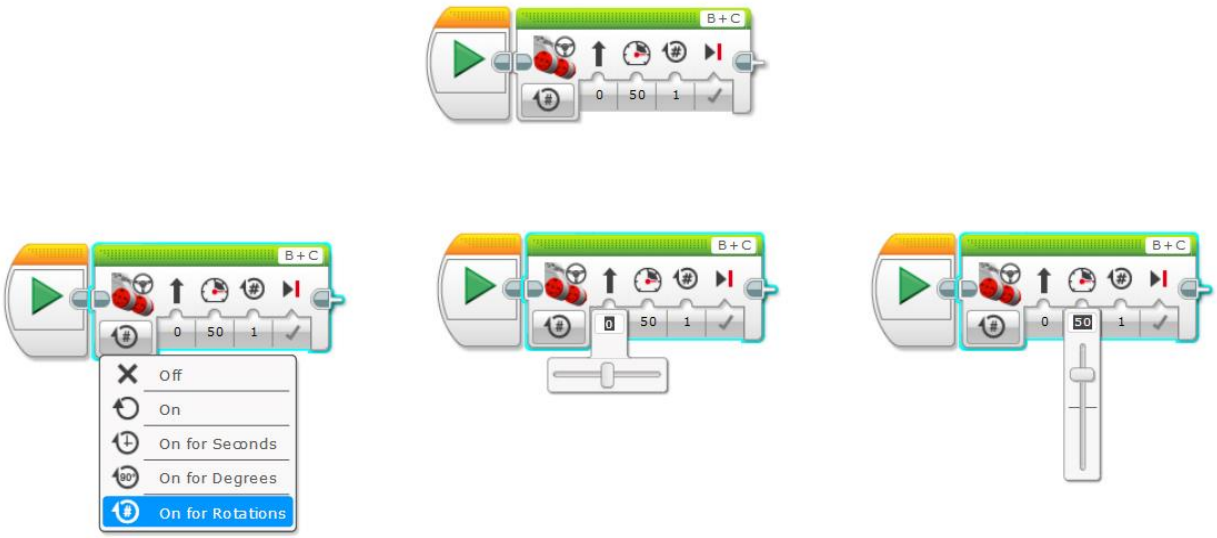
Uygulamaları yapabilmek için robotunun yeterli düzeyde şarj olmuş şekilde sınıfta bulunması gerekir. Ayrıca robotun sol büyük motorunun B portuna, sağ büyük motorunun C portuna bağlandığından emin olunmalıdır.

1.8. Hareket Blokları

Robotu hareket ettirecek program blokları yeşil sekmede (Action) yer alır. Orta motor, büyük motor, direksiyon hareketi (move steering) ve palet hareketi (tank move) olmak üzere dört farklı hareket bloğu bulunur..

1.8.1. Gözle: Direksiyon Hareketi (Move Steering)

Direksiyon hareketi bloğu robotun iki motorunu aynı anda kontrol eder. Her iki motorun pozitif veya negatif yönde dönmesini, dolayısıyla robotun ileri veya geri gitmesini sağlar. Robotun dönmesi ise motorlardan birinin daha hızlı dönmesi ile sağlanır. Bloğun sağ üst köşesinde (örneğin aşağıdaki resimlerde B+C olduğu gibi) hangi portlara bağlı olan motorların kontrol edileceği görülebilir. Eğer motorlar farklı portlara bağlanmışsa bu kısma tıklanarak portlar değiştirilebilir.



Resim 11. Direksiyon Hareketi Bloğu ve Ayarlanması

Öncelikle motorların çalışma modu seçilir (sol alttaki şekil): motorların durdurulması (off), sürekli (on), belirli bir süre (on for second), belirli bir açı değerinde (on for degrees) ve belirli bir tur sayısı (on for rotation). Seçilen moda bağlı olarak blok girişleri değişecektir.

Robotun sağa veya sola dönmesi direksiyon değerinin -100 ile 100 arasında değiştirilmesi ile sağlanır (altta ortadaki şekil). Robotun hızı güç değerinin -100 ile 100 arasında değiştirilmesi ile azaltılabilir ya da artırılabilir (sağ alttaki şekil). Negatif değerler robotun geri gitmesini sağlayacaktır. Seçilen moda bağlı olarak (süre, açı, tur) motorların ne kadar döneceği

ayarlanabilir. Son olarak motorlar dönme hareketini gerçekleştirdikten sonra durdurulabilir veya boşa alınabilir.

Not

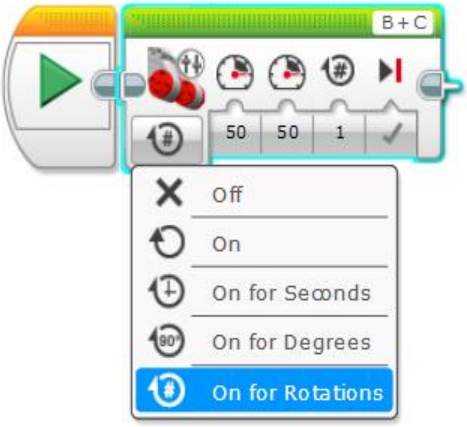
Özellikle derece modunda çalışırken, motorların 180 derece dönecek şekilde programlanmasının, robotun 180 derece dönmesi olarak algılanmaması gerektiği vurgulanmalıdır. Öğrencilerle bu durum tartışılarak oluşabilecek yanlış algı önlenmelidir.

1.8.2. Uygula: Tekerin Yarıçapını Hesaplama

Direksiyon hareketi bloğu anlatıldıktan sonra robotun bir miktar düz gidip kendi etrafında dönmeden geri geldiği bir program çalıştırılıp gösterilir. Öğrencilerden benzer bir uygulamayı tur, saniye ve açı modlarını kullanarak yapmaları istenir. Daha sonra masalarındaki matı kullanarak tekerin bir tur döndürüldüğünde aldığı mesafeyi cm cinsinden bulmaları beklenir. Etkinlik sonunda öğrencilerden tekerin yarıçapını bulmaları ve çevresini hesaplamaları istenir.

1.8.3. Gözle: Palet Hareketi (Move Tank)

Palet hareketi bloğu, direksiyon hareketi bloğuna hayli benzer. Palet hareketi bloğu iki motorun birbirinden bağımsız olarak farklı güç değerlerinde çalıştırılabilmesine imkân sağlar. Böylece robot olduğu yerde (merkezi etrafında) veya bir tekerinin etrafında dönebilir.



Resim 12. Palet Hareketi Bloğu

1.8.4. Uygula: Palet Hareketi

Palet hareketi bloğu anlatıldıktan sonra öğrencilerden bu bloğu kullanarak robotun dönmesini sağlayan bir program oluşturmaları istenir. Motorların güç değeri arasındaki fark arttıkça robotun dönüşünde meydana gelen değişikliği gözlemlemeleri beklenir. Öğrenciler robotun olduğu yerde veya bir tekerinin etrafında dönebilmesi için motor güçlerinin nasıl ayarlanması gerektiğini, mat üzerinde yer alan kesikli kırmızı çizgilerle belirtilen daireleri kullanarak keşfeder. Öğrencilerden robotlarını belirli bir mesafe (örneğin 3 tur) düz gittikten sonra kendi etrafında dönüp tekrar başlangıç noktasına gelecek şekilde programlamaları istenir.

2. TASARLA

Gün içerisinde robotun tekerinin çapını öğrenen öğrencilerden, bu bilgiyi kullanarak robotlarının 50 cm düz ilerleyip durmasını, sonra takılı olan kolu indirip LEGO parçaları ile yapılmış bir kutuyu tutarak kendi etrafında 180 derece dönmesini ve başlangıç noktasına getirmesini sağlayan bir program tasarlamaları istenir. Tasarlama aşamasında öğrencilerin aşağıda örnek olarak verilen iki adıma benzer bir süreci gerçekleştirmesi sağlanır.

Not
LEGO parçaları ile yapılacak kutu yönergesine aşağıdaki yollarla ulaşılabilir: i) https://le-www-live-s.legocdn.com/sc/media/lessons/mindstorms-ev3/building-instructions/ev3-cuboid-dc93b2e60bed2981e76b3bac9ea04558.pdf ii) EV3 yazılımı> Lobby> Building Instructions> Building Ideas> Cuboid iii) Robot setiyle birlikte gelen kitapçığa bakılabilir.

Tanımlama: Öğrencilerden öncelikle robotun istenilen hareketleri yapabilmesi için neler gerektiğini belirlemeleri ve maddeler hâlinde yazmaları beklenir. Bu aşamada kâğıt ve kalem kullanarak programın algoritmasını veya akış diyagramını hazırlamaları istenir.

Örneğin robot;

- 50 cm ileri gidecek ve duracak,
- Kolunu kutunun üzerine indirecek,
- Kendi etrafında kutuyla birlikte dönecek,
- Başlangıç noktasına geri dönecek.

Fikir üretme: Bu aşamada öğrencilerin tanımlamada belirlenen işlemlerin nasıl yapılabileceği ile ilgili fikir yürütmesi beklenir. Örnek olarak öğrenciler aşağıdaki maddelere benzer fikirler üretebilir:

- Öncelikle robotun hangi hareketleri yapacağı planlanmalıdır. Planlanan her bir hareketin tanımı yapılmalıdır (Örneğin, ilk harekette 50 cm ileri gidip durur ve etrafında 180 derece döner.). Bu aşamada öğrenciler hareketlerin programlanmasını hızlıca deneyebilirler.
- Robot kolunun indirilmesi için orta motor kullanılır.
- Kolun kutuyu kavraması sağlanır.
- Robotun kendi etrafında 180 derece dönmesi sağlanır.
- Robotun başlangıç noktasına geri gelmesi sağlanır.

Resim 13. Robot Kolunun Kutuyu (LEGO Parçasını) Taşıması

Robotu planlanan şekilde hareket ettirmek için mat üzerindeki ilgili bölge kullanılabilir. Taşınacak kutu oluşturulurken setle gelen LEGO parçaları kullanılabilir. Kutunun taşınması ile ilgili oluşabilecek muhtemel sorunlarla ilgili öğrencilerden EV3 robot setinde bulunan parçaları kullanarak çözüm üretmeleri istenir.

Not
Orta motor ile kolun, temel tasarım robotuna takılması için gerekli yönergeye aşağıdaki yollarla ulaşılabilir: i) https://le-www-live-s.legocdn.com/sc/media/lessons/mindstorms-ev3/building-instructions/ev3-medium-motor-driving-base-e66e2fc0d917485ef1aa023e8358e7a7.pdf ii) EV3 yazılımı> Lobby> Building Instructions> Building Ideas> Medium Motor Driving Base iii) Robot setiyle birlikte gelen kitapçığa bakılabilir.

3. ÜRET

Tasarla ve üret bölümlerinde öğrenciler aktif rol üstlenerek verilen problemi çözerler. Rehber öğretmen öğrencilere yalnızca zorlandıkları noktalarda destek olur. Öğrenciler bilgisayar ve robot başında çalışarak gerekli yazılım çözümlerini geliştirirler (Eğitim programındaki bu ve benzeri bütün örnekler/programlar rehber öğretmenlere verilecektir).

4. DEĞERLENDİR

Günün sonunda öğrencilerle halka oluşturulur ve aşağıdaki sorular üzerinden bir tartışma yürütülür:

- Robotun hızının değiştirilmesi ne gibi sonuçlara yol açtı? (Hız-zaman ilişkisi tartışılır)
- Palet hareketi (move tank) ile direksiyon hareketi (move steering) arasında fark var mıdır?
- Sizce teker büyüklüğünün bir önemi var mıdır?

- Programınızda tur sayısını değiştirmeden sadece teker büyüklüğünün değiştirilmesi ne gibi sonuçlara yol açar?

Bu soruların cevaplarına göre farklı robotların da tasarlanabileceğine dair örnekler verilebilir.

5. İLAVE ETKİNLİK

5.1. Yarışma

Öğrencilerin daha eğlenceli bir şekilde çalışmasını sağlamak amacıyla bir yarışma düzenlenebilir. Yarışma grup olarak yapılır. Yarışmanın sadece eğlenmek için yapıldığı, sonucun başka bir şekilde değerlendirilmemesi gerektiği vurgulanır.

Öğrencilerden ön tekerleri matın kısa kenar çizgisi üzerinde bulunan bir robotun hareket ederek karşı kenar çizgisine yine ön tekerleri gelecek şekilde ilerlemesini sağlayacak bir program oluşturmaları istenir. Matın A yüzeyindeki 98 cm'lik kesikli kırmızı çizgi bu amaçla kullanılabilir. Robotun rota üzerinde kendi etrafında tam 180 derece dönmesi gerektiği unutulmamalıdır.

Amaç, robotun en kısa sürede görevi tam olarak (hatasız şekilde) gerçekleştirmesidir. Rehber öğretmen görevin tamamlanması için azami bir süre belirleyerek yarışma öncesinde öğrencilere duyurur. Grupların görevi tamamlama süreleri rehber öğretmen tarafından takip edilir ve en kısa sürede görevi tamamlayan grup yarışmayı kazanır (Rehber öğretmen kendisine verilen örnek çözümü inceleyebilir ama çözümü öğrencilerle paylaşmamalıdır, öğrencilerin görevi kendilerinin yapmasına izin vermelidir.).

2. Hafta: Robotlarla Ses, Metin, Resim

Ön Bilgi:

- Öğrenciler robot kavramını temel düzeyde bilir.
- Öğrenciler robot setiyle farklı robotik tasarımlar yapmıştır.
- Öğrenciler robot setini programlamak için grafik arayüzünü kullanmıştır.
- Öğrenciler alan, uzunluk, eşitlik, denklem ve çap (yarıçap) kavramını öğrenmiştir.
- Öğrenciler robotun hareket etmesi için gerekli programlama adımlarını oluşturmuştur.

Haftanın Kazanımları:

- Öğrenciler akıllı tuğlanın çeşitli sesleri (dosya, frekans ve nota) çıkarması için gerekli programlama adımlarını oluşturur.
- Öğrenciler akıllı tuğla ekranının görüntüsünü (metin, şekil, imaj) düzenlemek için gerekli programlama adımlarını oluşturur.
- Öğrenciler robotu programlarken döngü mantığını kurup uygulayabilir.

Haftanın Amacı:

Bu haftanın amacı öğrencilerin öncelikle EV3 yazılımının "Eylem (Action)" sekmesinde bulunan "Ses (Sound)", "Ekran (Display)" ve "Tuğla Durum Işığı (Brick Status Light)" bloklarını sağlamaktır. Öğrencilere, haftaya başlarken robotun hareket etmesi için gerekli programlama adımları hatırlatılır, ardından öğrencilerin Eylem (Action) sekmesiyle ilgili blokları programlama sırasında çeşitli amaçlar için kullanırken gerekli düzenlemeleri yapabilmesi sağlanır.

Kullanılacak Malzemeler:

Robot seti, bilgisayar

Haftanın İşlenişi:

Gözle: Eylem (Action) sekmesinde bulunan Ses (Sound), Ekran (Display) ve Tuğla Durum Işığı (Brick Status Light) bloklarının özelliklerini ve bu blokları kullanmak için gereken "döngü" kavramını inceleme.

Uygula: Her bir bloğun bileşenlerini programlayarak uygulama ve döngü kavramı ile örnek uygulama geliştirme (döngüsüz ve döngülü programlar oluşturma).

Tasarla: Robotun istenilen işlemleri yapabilmesi için gereken bileşenleri tanımlama ve planlama.

Üret: Robotun planlanan işlemleri yapabilmesi için gereken robot tasarımını ve programlama adımlarını oluşturma

Değerlendir: Haftanın içeriği ile ilgili yansıtma etkinliği

1. GÖZLE VE UYGULA

1.1. Gözle: Hareket Bloklarını Hatırlama

Geçen hafta robotun hareket ettirilmesi ile ilgili motorlar hakkında bilgiler verilmişti. Hem bu bilgileri hatırlamak hem de robot tasarımını tekrar hazır hâle getirmek için rehber öğretmen özet bilgiler vererek örnek bir uygulama yapar. Robotun kare çizerek hareket etmesini sağlayan uygulamanın adımlarını programlayarak öğrencilere gösterir (**Dikkat**: Herhangi bir sensör kullanılmamıştır).

Resim 14. Örnek Program

1.2. Uygula: Sonsuz İşareti Şeklinde Hareket Etme

Bu uygulama için iki nesne (örneğin iki su şişesi) sınıf içinde müsait bir alana, aralarında 40-50 cm olacak şekilde yerleştirilir. Robotun bu nesnelere dokunmadan etraflarında "8" veya "sonsuz sembolü" çizerek hareket etmesi sağlanır. Robotun izleyeceği örnek yol aşağıdaki resimdeki gibi olabilir (Çözüm rehber öğretmenlere verilecektir, öğrencilerle paylaşılmamalıdır).

Resim 15. Örnek Yol

1.3. Gözle: Ses (Sound) Bloğu

Hareket bloklarının ardından Ses (Sound) bloğu öğrencilere anlatılır ve bazı örnekler gösterilir.

Robot iki tür ses çalabilir: (i) "bip" sesi gibi basit bir ton, (ii) önceden kaydedilmiş "alkış" gibi bir ses veya "merhaba" gibi bir kelime. Programda bir ses bloğu kullanıldığında robot "konuşabileceği" için daha etkileşimli ve gerçekçi görünecektir.

Programlama paletinden bir ses bloğu seçilir ve programlama alanına yerleştirilir. Blok yerleştirildikten sonra mod seçilir ve hangi ses duyulmak isteniyorsa ona göre ayarlar yapılır. Ses bloğunun dört modu vardır:

- Play File: "Merhaba" gibi önceden kaydedilmiş bir sesi çalar.
- Play Tone: Belirli bir süre için belirli bir frekansta bir ton çalar.
- Play Note: Belirli bir süre için piyanodan bir nota çalar.
- Stop: Çalmakta olan tüm sesleri durdurur.

Yukarıdaki modlara göre aşağıdaki seçenekler düzenlenir.

File Name

Dosya Oynat (Play File) modunda, Dosya Adı (File Name) alanı tıklanarak menüden bir ses seçilebilir. Hayvanlar, renkler, iletişim ve sayılar gibi kategorilerden bir ses seçilebilir. Ayrıca, Araçlar (Tools) menüsünde yer alan Ses Düzenleyicisi (Sound Editor) kullanılarak ses kaydedilebilir ve eklenebilir.

Volume

Çalınmak istenen sesin seviyesini ayarlamak için “0” (yumuşak) ile “100” (yüksek) arasında bir sayı girilebilir.

Play Type

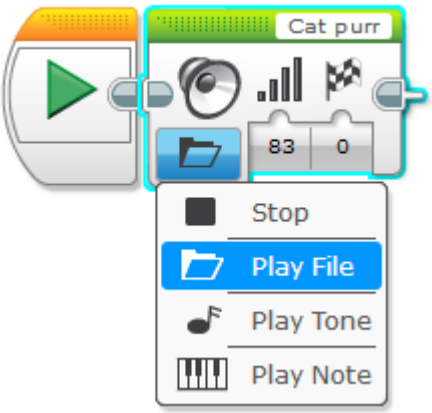
Ses çalmaya başladığında ne olacağını kontrol etmek için Çalma Türü (Play Type) ayarı kullanılabilir. Ses çalmayı durdurana kadar programı duraklatmak için tamamlanmasını bekle (0) ögesi seçilebilir. Ses çalarken programın bir sonraki bloğu çalıştırmaya devam etmesi için bir kez çal (1) ögesi seçilebilir. Tekrarla (2) seçilirse, program kalan blokları çalıştırırken ses tekrar eder. Çoğu program için tamamlanmasını bekle (0) seçeneği seçilecektir.

Note - Tone

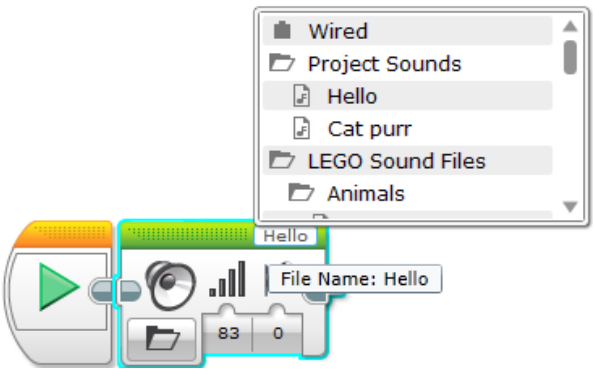
Tercih edilen moda bağlı olarak, piyano klavyesinden bir nota veya hertz (Hz) cinsinden bir ton (ton) seçilebilir. İnsan kulağı 440 Hz tonu (varsayılan frekans değeri) net bir şekilde duyar. Bu ton robota etkileşim katmak için kullanılabileceği gibi oluşturulan programları test etmek için de kullanılabilir. Örneğin belirli bir programlama bloğunun çalışmasının tamamlandığını göstermek için ses çalınabilir.

Duration

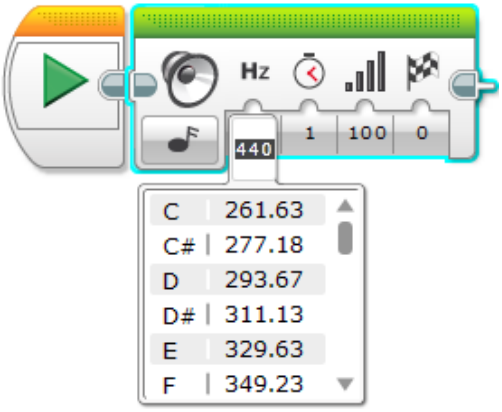
Süre (Duration) kutusuna notanın veya zil sesinin çalması istenen süresi saniye cinsinden girilebilir.



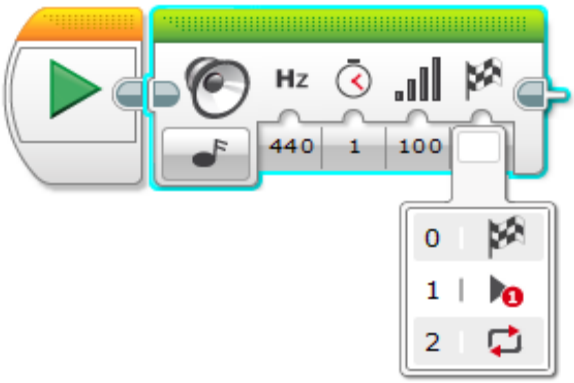
Resim 16. Play File Adım 1



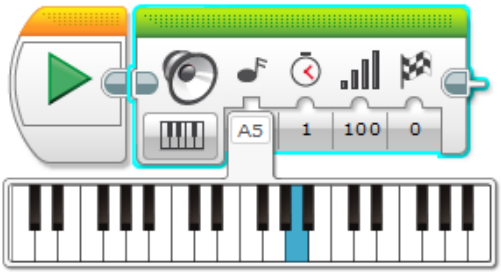
Resim 17. Play File Adım 2



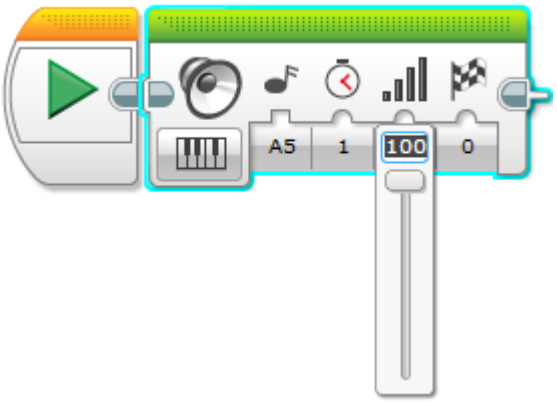
Resim 18. Play Tone Adım 1



Resim 19. Play Tone Adım 2

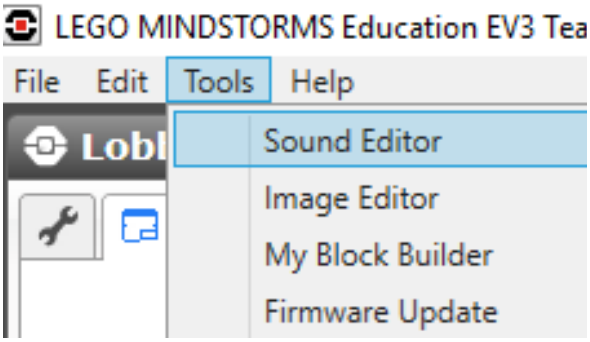


Resim 200. Play Note Adım 1

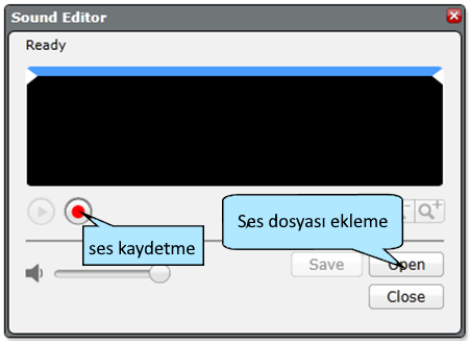


Resim 21. Play Note Adım 2

Kişisel ses dosyası oluşturmak veya ses dosyası eklemek için EV3 yazılımında Araçlar (Tools) menüsünden Ses Düzenleyicisi (Sound Editor) seçilir. Açılan pencereden kayıt düğmesine tıklanılarak ses kaydı yapılır. Bir isim verilerek kaydedilen ses dosyasına, Ses (Sound) bloğunun Dosya Oynat (Play File) kısmından ulaşılabilir.



Resim 22. Ses Kayıt Adım 1



Resim 23. Ses Kayıt Adım 2

1.4. Uygula: Ses Bloğu (Sound Block)

Nota çalma: Öğrencilerden ardışık dört notayı çalmaları istenir. Örnek program aşağıdaki resimde görülmektedir.

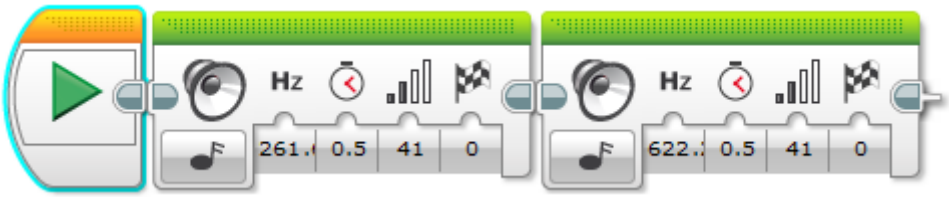
Resim 24. Örnek Program

Kendi hoş geldin mesajını oluşturma: Öğrencilerden kendi hoş geldin mesajlarını oluşturup bu mesajı robota söyleten bir program oluşturmaları istenir.

1.5. Gözle: Siren Sesi

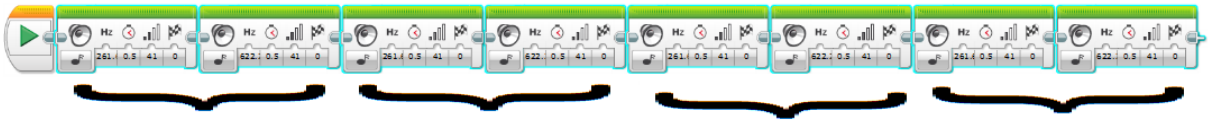
Öğrencilere daha önce siren sesi duyup duymadıkları sorulur. Öğrenciler sadece polis, itfaiye ve ambulans sirenlerinden bahsederlerse AFAD'ın sitesindeki https://www.afad.gov.tr/ikaz-alarm-isaretleri ikaz alarm seslerinden de bahsedilir. Verilen örneklerdeki araçların (ambulans, polis arabası vb.) siren seslerinin ortak özelliği öğrencilere sorulur. Eğer uygun cevap veremezlerse siren seslerinin tekrarlayan daha kısa seslerden oluştuğu vurgulanır.

Robotun programlanmasıyla siren veya ikaz sesleri çıkarabileceğinden bahsedilir. Aşağıdaki resimde görülen program ekranda gösterilir. Program robota yüklenir ve çalıştırılır. Robotun iki farklı frekansı 0,5 saniye süreyle ardışık olarak çaldığı açıklanır.



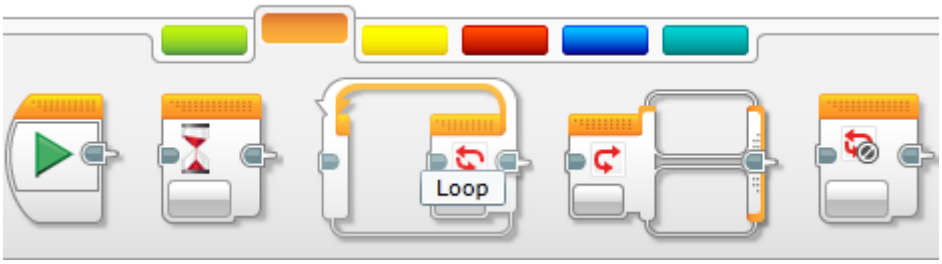
Resim 25. Örnek Program

Rehber öğretmen, 4 saniyelik siren sesi için robotun her iki frekansı sıra ile dörder defa çalması ve programın da buna göre düzenlenmesi gerektiğinden bahseder. Aşağıdaki resimde örnek program görülmektedir.



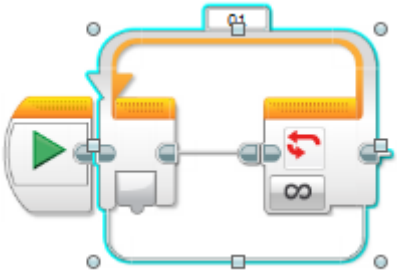
Resim 26. Örnek Program

Rehber öğretmen, tekrarlanan işlemler için Akış Kontrol (Flow Control) sekmesinden Döngü (Loop) bloğunun seçilmesi gerektiğini söyler.



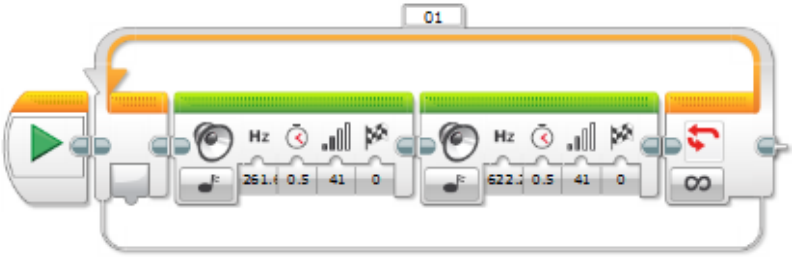
Resim 27. Flow Control Sekmesi

Daha sonra, Döngü bloğunun başla komutuna eklenmesi gerektiği ifade edilir. Öğrenciler ilk defa döngü kavramıyla karşılaştığı için programlama sürecinde kullanılan döngülerle ilgili kısa bilgiler paylaşılır. Özellikle döngülerin tekrarlı adımlar yapılırken kullanıldığı söylenir.



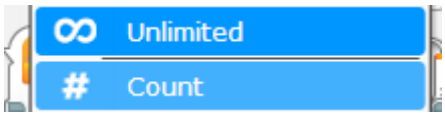
Resim 28. Döngü (Loop) Bloğu

Döngü (Loop) bloğunun içerisine tekrar etmesi istenilen iki frekans eklenir.

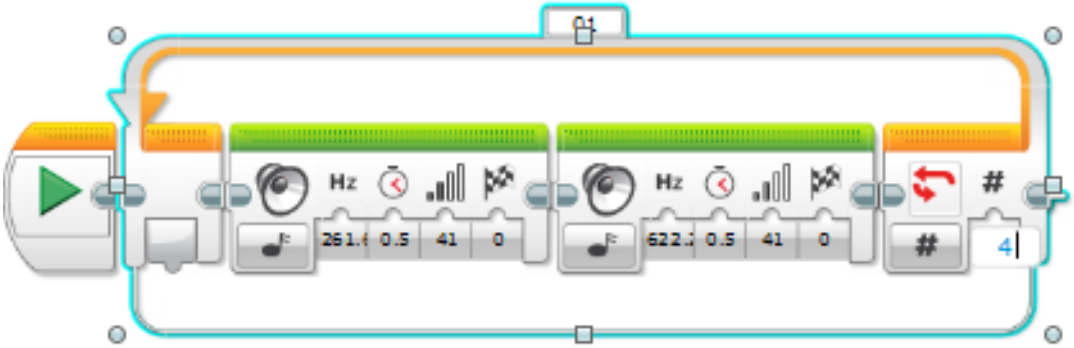


Resim 29. Örnek Program

Program robota yüklenir ve öğrencilere iki sesin sürekli olarak ardı ardına çaldığı gösterilir. Eğer her iki sesin ardışık olarak dörder defa çalması arzu edilirse döngüdeki Sınırsız (Unlimited) simgesine tıklanıp Sayma (Count) seçeneğinin seçilmesi ve değerinin 4 yapılması gerekir.



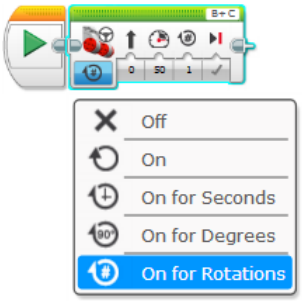
Resim 30. Adım Sayısının Belirlenmesi 1



Resim 31. Adım Sayısının Belirlenmesi 2

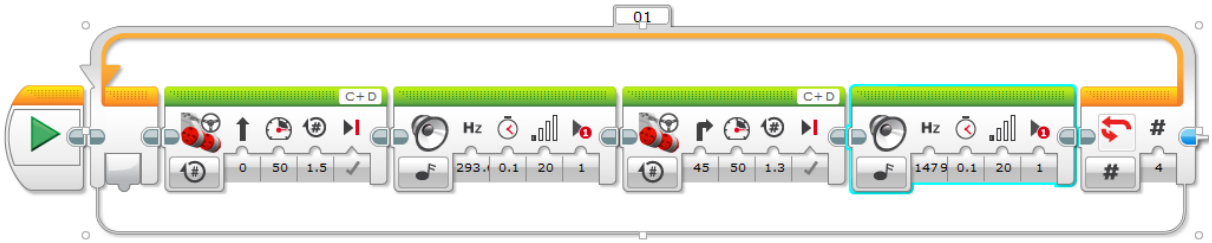
1.6. Uygula: Kare Çizen Robot

Öğrencilerden robotun 20 cm'lik bir kare üzerinde ilerlemesi için bir program oluşturmaları istenir. Programı oluştururken Döngü (Loop) bloğunu kullanmaları; robotun 20 cm ilerleyince bir ses çıkarmasını, olduğu yerde sağa veya sola (90 derece) dönünce de farklı bir ses çıkarmasını sağlamaları istenir. (Öğrenciler robotun 20 cm ilerleyebilmesi ve sağa veya sola (90 derece) dönebilmesi için gerekli bilgileri önceki derste öğrenmişlerdir). 20 cm'yi ayarlayabilmek için "On for Degrees" veya "On for Rotations" seçeneklerinden birinin kullanılmasının gerektiği hatırlatılmalıdır.



Resim 32. Dönüş Şeklinin Belirlenmesi

Aşağıdaki resimde programın tamamlanmış hâline örnek verilmiştir (Dönüş sayısı için verilen 1,3 değeri robota ve fiziksel ortama göre değişiklik gösterebilir):



Resim 33. Örnek Program

1.7. Gözle: Display (Ekran) Bloğu

Robotu hareket ettirme ve ses çalmanın yanı sıra bir EV3 programı "akıllı tuğla" nın ekranını kontrol edebilir. Ekran 178 piksel genişliğinde ve 128 piksel yüksekliğindedir. (Piksel, ekranda görüntüyü oluşturan küçük noktalardır).

Resim 34. EV3 Ekranı

Rehber öğretmen öğrencilere yukarıdaki resmi gösterir.

- Ekranın sol üst köşesinden sağ üst köşesine kadar 178 nokta olduğunu ve herhangi bir noktanın ekranın sol kenarından uzaklığını belirtmek için Ekran (Display) bloğunda "x" ifadesinin kullanıldığını söyler.
- Ekranın sol üst köşesinden sol alt köşesine kadar 128 nokta olduğunu ve herhangi bir noktanın ekranın üst kenarından uzaklığını belirtmek için Ekran (Display) bloğunda "y" ifadesinin kullanıldığını söyler.
- Ekranda herhangi bir noktayı belirtmek için de noktanın ekranın sol kenardan kaç nokta uzakta olduğunu (yani x değeri) ve noktanın ekranın üst kenarından kaç nokta uzakta olduğunu (yani y değeri) bilmeleri gerektiğini söyler.

Display bloğuyla oynamak eğlenceli olduğu kadar programı test etmenin etkili yollarından biridir. Örneğin sensörün düzgün çalışıp çalışmadığını görmek için ekranda bir sensör ölçümü görüntülenebilir.
Akıllı tuğla ekranında bir görüntüyü (gülen yüz), metni ("Merhaba!") veya bir şekli (dolu bir daire gibi) görüntülemek için Ekran (Display) bloğu kullanılır. Tek bir Ekran bloğu ekrana bir defada birden fazla görüntü veya metin satırı koyamaz, bu nedenle ekranı oluşturmak için bazen bir dizi Ekran bloğunun kullanılması gerekebilir.

<u>Ekran (Display) Bloğu Ayarları</u>
Bir ekran bloğu "akıllı tuğla" ekranına bir görüntü eklediğinde, program bir sonraki bloğa geçer (örneğin bir move bloğu). Akıllı tuğla ekranı, başka bir ekran görüntülemek için yeni bir ekran bloğu kullanılana kadar görüntüyü göstermeye devam eder. Bir program sona erdiğinde akıllı tuğla hemen menüye döner. Bu, programdaki son blok bir Ekran bloğu ise program sona ereceği için ekranda ne olduğunu görmek için zaman kalmayacağı anlamına gelir. Ekranda ne olduğunu görmek ve programın hemen sona ermesini engellemek için Bekle (Wait) bloğu eklenir.

- Image: Ekranda mutlu bir yüz gibi seçilmiş bir resim gösterir.
- Shapes: Ekranda çizgi, daire, dikdörtgen veya nokta gösterir.
- Text: Ekranda bir metin satırı gösterir.

- Reset Screen: Ekranı temizler ve ekran blokları olmayan bir program çalıştırıldığında normalde görülen "MINDSTORMS" logosunu gösterir.

Alt Modlar (Sub Modes)

Bazı modlarda alt seçenekler vardır. Ekran bloğunda şekiller modu seçilirken dört alt moddan birinin tercih edilmesi gerekir (çizgi, daire, dikdörtgen ve nokta). Daire modu, ekran bloğunun EV3 ekranında bir daire şekli göstermesini sağlar. Dairenin konumunu, yarıçapını, dolgusunu ve rengini yapılandırmak için bloktaki ayarlar kullanılabilir.

Dosya İsmi (File Name)

Görüntü modundayken gözler, ifadeler, nesneler ve LEGO gibi kategorilerden bir görüntü seçmek için Dosya Adı (File Name) alanı kullanılır. Araçlar (Tools) sekmesinden Resim düzenleyicisine gidilerek kişisel resimler oluşturulabilir veya yüklenebilir.

Ekranı Temizle (Clear Screen)

Ekranı Temizle (Clear Screen) ayarı, yeni bir şey göstermeden önce ekranı boşaltmayı (doğru olarak ayarlandığında) veya ekranda zaten olanlara (yanlış olarak ayarlandığında) yeni bir şey eklemeyi seçmeyi sağlar. Ekranda birden fazla nesne göstermek için bir dizi ekran bloğuna ihtiyaç olacaktır. İlk blok yeni bir şey göstermeden önce ekran temizlenmeli ve diğer bloklar ekrana bir şey eklemelidir. Bunu başarmak için ilk blokta Clear Screen ayarını "true (doğru)" yapmak ve izleyen bloklarda "false (yanlış)" olarak ayarlamak gerekir.

Yarıçap ve Doldurma (Radius and Fill)

Bazı ayarlar, ekran bloğunun farklı modlarına özgüdür. Örneğin "radius (yarıçap)" ayarı bir dairenin boyutunu belirtir ve "fill (doldurma)", düz bir daire yapılmasına (doğru) veya yalnızca anahat çizilmesine (yanlış) izin verir.

Renk (Color)

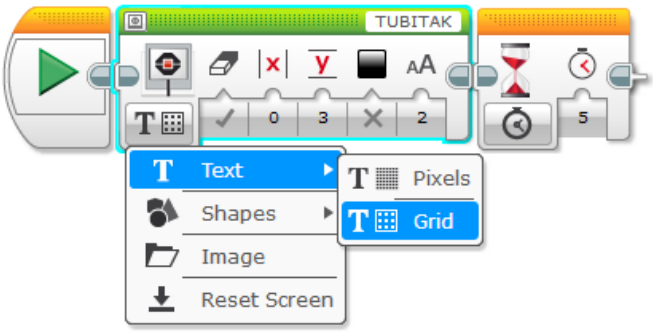
Renk (Color) ayarı siyah (yanlış) olarak ayarlanmak istenirse ve daha önce bir alan siyah bir daireyle doldurulduysa, renkler beyaz (doğru) olarak ayarlanarak üstüne metin eklenebilir.

Metin ve Yazı Boyutu (Text and Font Size)

Metin (Text) modunda, metin alanına, "MINDSTORMS" gibi, görüntülenmek istenen bir metin satırı girilir. Metin satırı sayılar içerebilir ve "font size (yazı tipi boyutu)" değeri "0" (küçük), "1" (kalın) veya "2" (büyük) olarak ayarlanarak metnin boyutu değiştirilebilir.

1.7.1 Gözle: Text

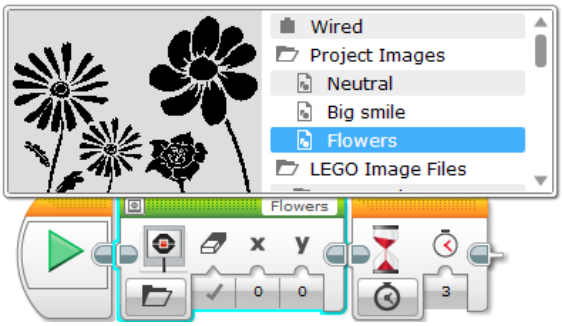
Text seçeneği ile ekrana yazı (örneğin TUBITAK) yazılabilir fakat "wait (bekle)" eklenmezse yazı akıllı tuğla ekranında görünmez. Ekranda gösterilirken hata yaşanabileceği için Türkçe karakterler kullanılmamalıdır. Aşağıdaki resimde display bloğu örneğinde "x" değeri "0" (sıfır) y değeri "3" olarak girilmiştir. TUBITAK yazısı ekranın hemen sol kenarından (Çünkü "x" değeri sıfır girilmiştir.) ve ekranın üstünden 3 pixel boşluk bırakılarak (Çünkü "y" değeri 3 girilmiştir.) yazılacaktır.



Resim 35. Ekran Yazısı Yazdırma

1.7.2. Gözle: Resim/Image Dosyaları

Hazır resim dosyaları eklenebilir, benzer şekilde "wait (bekle)" eklenmezse akıllı tuğla ekranına görüntü gelmez. Aşağıdaki resimde display bloğu örneğinde "x" değeri 0 (sıfır), "y" değeri 0 (sıfır) olarak girilmiştir. Flowers dosyası ekranın sol üst köşesinden itibaren (Çünkü "x" ve "y" değeri sıfır girilmiştir.) görüntülenecektir.

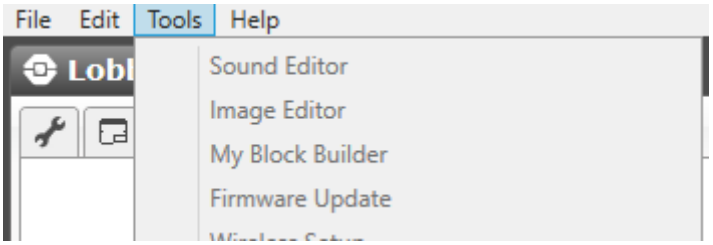


Resim 36. Ekrana Resim Basma

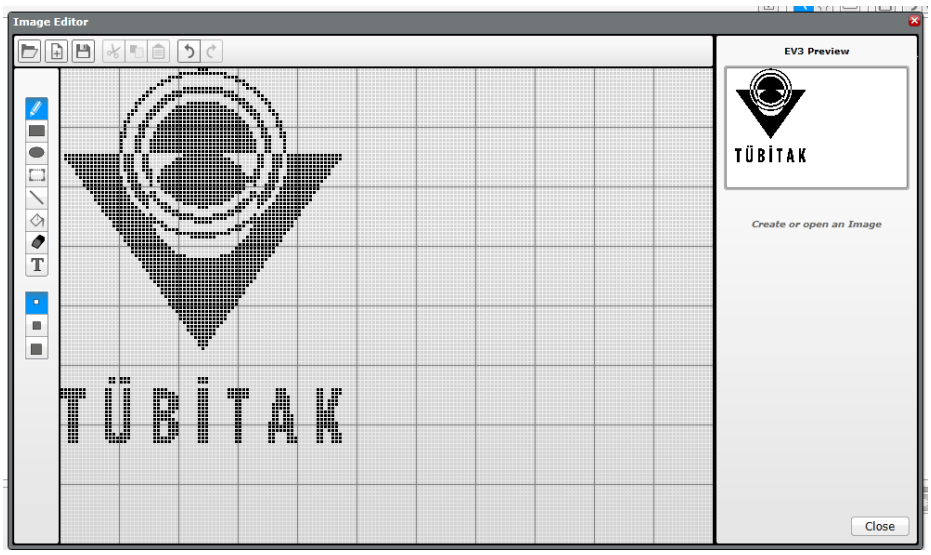
1.7.3. TÜBİTAK Logosunu Yükleme

TÜBİTAK logosunu yüklemek için gerekli adımlar aşağıdaki resimlerde görülebilir. TUBİTAK logosunu yüklemek için EV3 yazılımında Araçlar (Tools) menüsünden Resim Editörü (Image Editor) seçilir. Açılan pencereden Aç (Open) düğmesine tıklanılarak resim yüklenmeye başlanır. Bir isim verilerek kaydedilen resim dosyasına Ekran (Display) bloğunun Project Images kısmından ulaşılabilir.

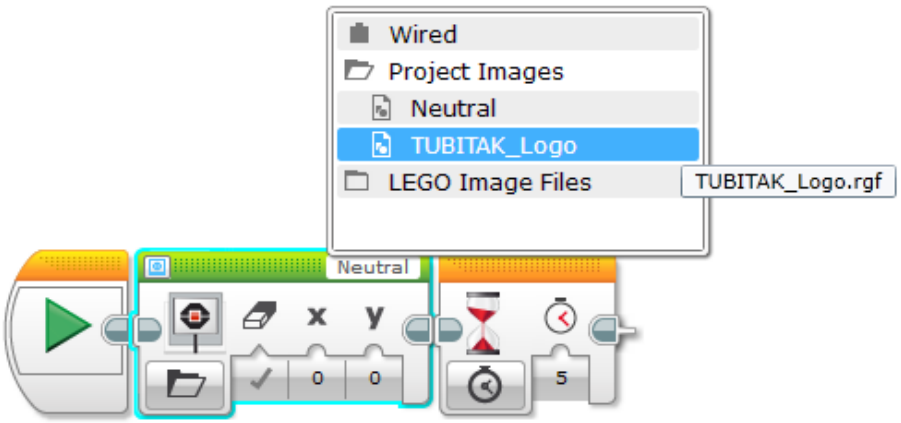
.



Resim 37. Logo Ekleme Adım 1



Resim 38. Logo Ekleme Adım 2



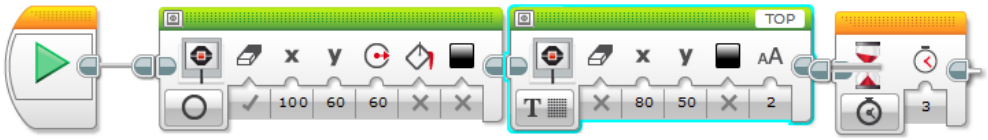
Resim 39. Logo Ekleme Adım 3

1.8. Uygula: Display Bloğu

1.8.1. Metin Yazdırma

Öğrencilerin bir daire içine isimlerini veya soy isimlerini yazdırabilecekleri bir uygulama yapılır. Aşağıdaki resimde ilk "display" bloğunda dairenin merkezi için "x" değeri 100 ve "y" değeri 60 girilmiştir. Yarıçap olarak da 60 girilmiştir. Bu daire tanımlaması ile merkezi ekranın

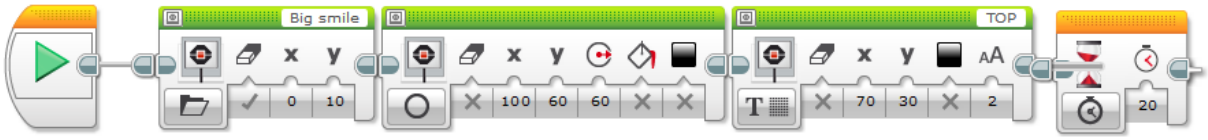
sol kenarından 100 pixel ve ekranın üst kenarından 60 pixel uzaklıkta olan 60 pixel yarıçapında bir daire çizilecektir.



Resim 40. Örnek Program

1.8.2. Resim (Image) Gösterme

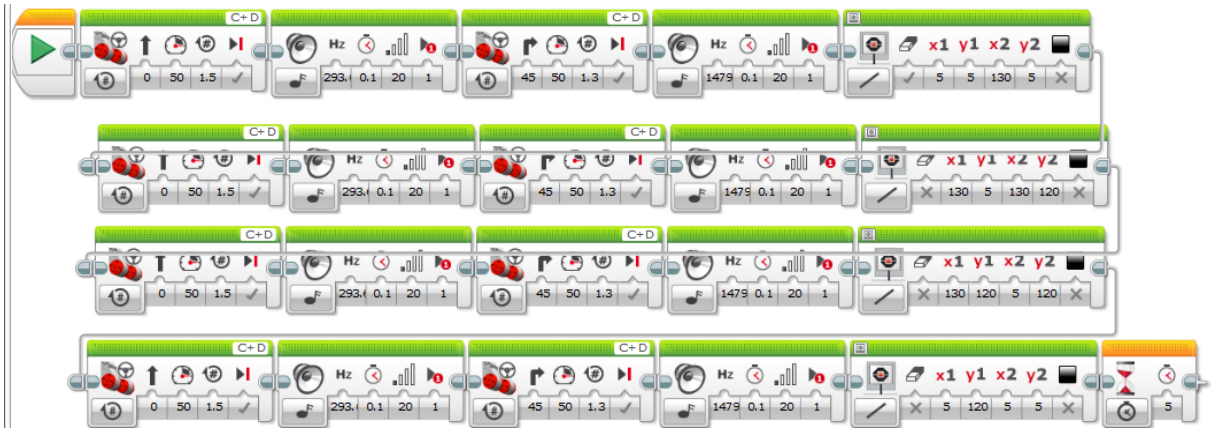
Öğrencilerin bir daire içine "big smile" resmi ile isimlerini veya soy isimlerini yazdırabilecekleri bir uygulama yapılır. Örnek program aşağıdaki resimde görülmektedir.



Resim 41. Örnek Program

1.8.3. Şekil Oluşturma

Bu aşamada, 1.6. Uygula başlığında yapılan kare çizen robot örneği tekrar düzenlenir. Her çizgi tamamlandığında akıllı tuğlanın ekranında o çizgi de gösterilir. Kare tamamlanınca ekranda 5 saniye kalması sağlanır.



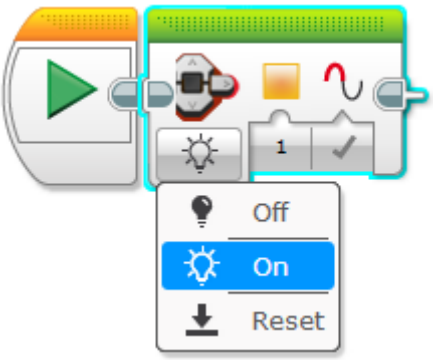
Resim 42. Örnek Program

1.9. Gözle: Tuğla Durum Işığı (Brick Status Light) Bloğu

Öncelikle Eylem (Action) sekmesinde bulunan Tuğla Durum Işığı (Brick Status Light) bloğu programa eklenerek başlanır.

EV3 düğmelerini çevreleyen Tuğla Durum Işığı (Brick Status Light) normalde yeşildir. Bir program çalışırken yanıp söner. Tuğla Durum Işığı bloğu ile ışığın nasıl yanacağı seçilebilir. Seçilebilecek üç mod aşağıdaki gibidir:

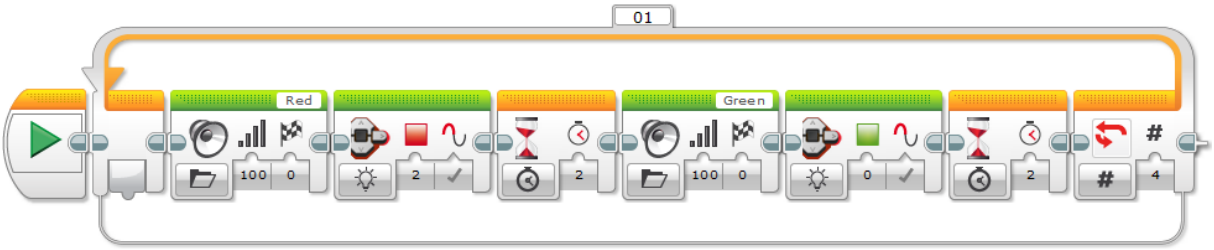
- On: Işığı açar ve bir renk seçilmesini sağlar: yeşil (0), turuncu (1), kırmızı (2). Darbe ayarı ile ışığın yanıp söneceği mi (doğru) yoksa sürekli açık mı kalacağı (yanlış) seçilebilir.
- Off: Işığı kapatır.
- Reset: Tuğla Işığı (Brick Status) bloğu ile herhangi bir işlem yapılmadan program çalıştırıldığında yanıp sönen yeşil ışığın gösterilmesini sağlar.



Resim 43. Tuğla Durumu Işığı Ayarları

1.10. Uygula: Tuğla Durum Işığı (Brick Status Light) Bloğu

Bu uygulamada sırası ile kırmızı ve yeşil ışığın yakıldığı bir program hazırlanır. Her bir ifade arasında iki saniye beklenir ve bu işlem beş defa tekrar edilir. Örnek program aşağıdaki resimde görülmektedir.



Resim 44. Örnek Program

2. TASARLA

2.1. Dans Eden Robot

Tasarla aşamasında öğrencilerden robotlarını dans edecek şekilde programlamaları istenir. Robot dans ederken hareketler, müzik ve akıllı tuğla ekranındaki görseller senkronize bir şekilde kullanılmalıdır.

Öğrencilerden dans eden robotun kodunun nasıl yazılacağı üzerinde düşünmeleri istenir. Öğrenciler grup olarak tartışırlar. Gerektiği noktada rehber öğretmen onlara yardımcı olabilir. Fakat öğrencilere tam bir çözüm verilmemelidir. Gruplar çözümü kendileri üretmelidir, tam bir çözümün verilmesi öğrencilerin yaratıcılıklarını olumsuz yönde etkileyebileceğinden tavsiye edilmez. Dans eden robotu tasarlamak için öğrencilerin aşağıda örnek olarak verilen iki adıma benzer bir süreci gerçekleştirmeleri gerekir.

Tanımlama: Öncelikle öğrenciler robotun dans edebilmesi için neler gerektiğini belirlemelidir. Bu bir dans stili, farklı şekillerde hareket eden, nota çalan ve ekranda uygun ifade gösteren bir robot olabilir. Öğrenciler gerekli işlemleri maddeler hâlinde yazmalıdır.

Örneğin;

- Nota çalacak,
- Nota ile paralel olarak dans edecek (sağ, sol, ileri, sağ, sol, geri hareket edecek),
- Akıllı tuğla ekranında göz hareketleri ile sağa sola dönüş ifadesi; ileri gidince mutlu ifade, geri gidince mutsuz ifade gösterecek,
- Dansı sonlandıracak.

Fikir üretme: Bu aşamada öğrencilerin, tanımlamada belirlenen işlemlerin nasıl yapılabileceği ile ilgili fikir yürütmesi beklenir. Örnek olarak öğrenciler aşağıdaki maddelere benzer fikirler üretebilir:

- Öncelikle robotun hangi hareketleri yapacağı planlanmalıdır. Planlanan her bir hareketin tanımı yapılmalıdır (Örneğin robot önce 0,5 saniye sağa doğru döner ve durur. Sonra iki katı hızlı bir şekilde yine 0,5 saniye sağa doğru döner ve durur.). Bu aşamada öğrenciler hareketlerin programlanmasını hızlıca deneyebilirler.
- Sonra her bir harekette tuğlanın çıkaracağı ses (dosya, frekans veya nota) belirlenir. Öğrencilerin internetten müzik dosyaları indirmelerine veya kendi müziklerini programlamalarına izin verilmelidir.
- Hareket ve ses ile tuğla ekranında hangi görsellerin gösterileceği de belirlenmelidir.
- Bu sürecin sonunda öğrenciler dans hareketlerini, müziğini, tuğla görüntülerini ve durum ışığı bilgilerini birlikte anlatabilmelidir.

3. ÜRET

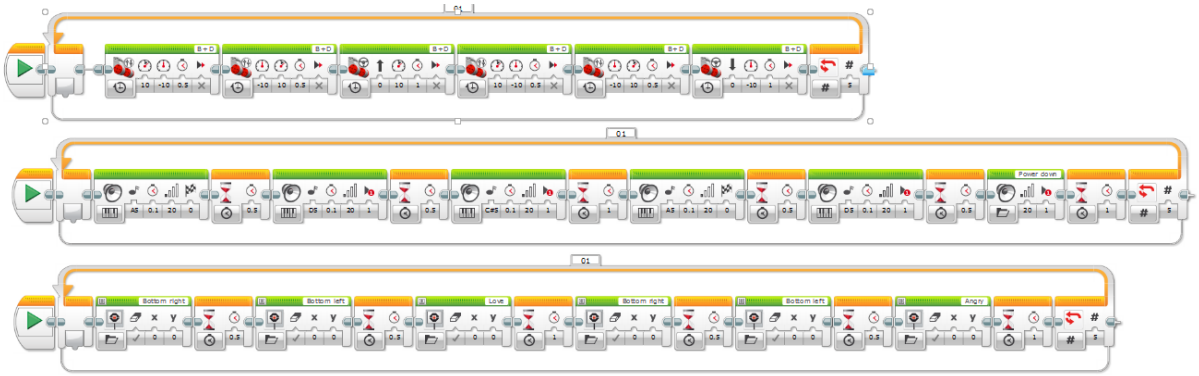
3.1. Dans Eden Robot

Bu bölümde de öğrenciler aktif rol üstlenir. Rehber öğretmen yalnızca yönlendirir ve öğrencilere takıldıkları noktalarda destek olur. Yani rehber öğretmen yalnızca ihtiyaç anında destek sağlamalıdır. Üret aşamasında, öğrencilerden bir önceki adımda tasarladıkları dans eden robot planını kullanarak probleme daha yapılandırılmış bir çözüm önerisi geliştirmeleri istenir. Öğrenciler bilgisayar ve robot başında çalışarak gerekli yazılım çözümlerini geliştirirler. Burada öğrencilerin en çok zorlanacağı konulardan biri ses, hareket ve tuğla ekranı görüntüsü senkronizasyonunun sağlanmasıdır.

Püf noktalar (Kendilerinin keşfetmesine veya sorarak öğrenmelerine imkân vermek için bunları başlangıçta öğrencilere söylemek uygun değildir.):

- Hareketin süresi ile resmin (image) ekranda bekleme süresi aynı olmalıdır.
- Notanın çalma süresinden bağımsız olarak, bekleme süresi hareketin süresi ile aynı olmalıdır. Örneğin nota 0,1 saniye çalıyorsa ve hareket 0,5 saniye sürüyorsa, bir sonraki hareketin notasını oynatmak için hareket süresi kadar (0,5 saniye) beklemek gerekir.

Aşağıdaki resimde nota çalıp dans eden (sağ, sol, ileri, sağ, sol, geri hareket eden) ve göz hareketleri ile sağa veya sola dönüşü beraber yapan, ileri gidince mutlu, geri gidince mutsuz ifade gösteren robotu programlamak için gerekli kodlar verilmiştir. Bu kod blokları rehber öğretmen için örnek bir dans stilidir. Öğrenciler farklı şekillerde hareket eden, nota çalan ve akıllı tuğla ekranında uygun ifade gösteren programlar yapabilir (Rehber öğretmen kendisine verilen örnek çözümü inceleyebilir ama çözümü öğrencilerle paylaşmamalıdır, öğrencilerin görevi kendilerinin yapmasına izin vermelidir).



Resim 45. Örnek Program

4. DEĞERLENDİR

Bu aşamada hedef, öğrencilerin öğrenme sürecinde yaşadıkları ve öğrendikleri üzerine düşünmelerini sağlamaktır. Bu sayede öğrenciler; problem çözme yetenekleri, dersin konusu ve kendileri ile ilgili gözlemler yaparak yeni bilgiler öğrenme, kendilerini değerlendirme ve sonraki çalışmalarını planlama açısından fırsatlar elde edecektir. Öğrencilerin şu soruları cevaplamaları istenebilir:

- Size bugün Tasarla ve Üret adımında verilen problemi nasıl tanımlarsınız? (problemi kendi cümleleri ile ifade etme).
- En çok hangi görevde zorlandınız? Bu zorlukların üstesinden nasıl geldiniz? (Problemin çözümü için hangi stratejileri kullandınız ve neden bu stratejileri seçtiniz?) Yeteri kadar tartışma ortamı oluşmazsa rehber öğretmen aşağıdaki soruları kullanarak tartışma ortamı yaratmaya çalışır.
    - Dans adımlarına nasıl karar verdiniz, bunları programlarken ne tür sıkıntılar yaşadınız ve bunların üstesinden gelmek için neler yaptınız?
    - Müzik, tuğla ekranı ve tuğla ışığına nasıl karar verdiniz; bunları programlarken ne tür sıkıntılar yaşadınız ve bunların üstesinden gelmek için neler yaptınız?
- Kullandığınız yöntemler bu sıkıntıları gidermede başarılı oldu mu?
- Grup arkadaşınızla fikir ayrılıkları yaşadınız mı? Bunların üstesinden gelmek için neler yaptınız?
- Grup arkadaşınızdan ne/neler öğrendiniz?

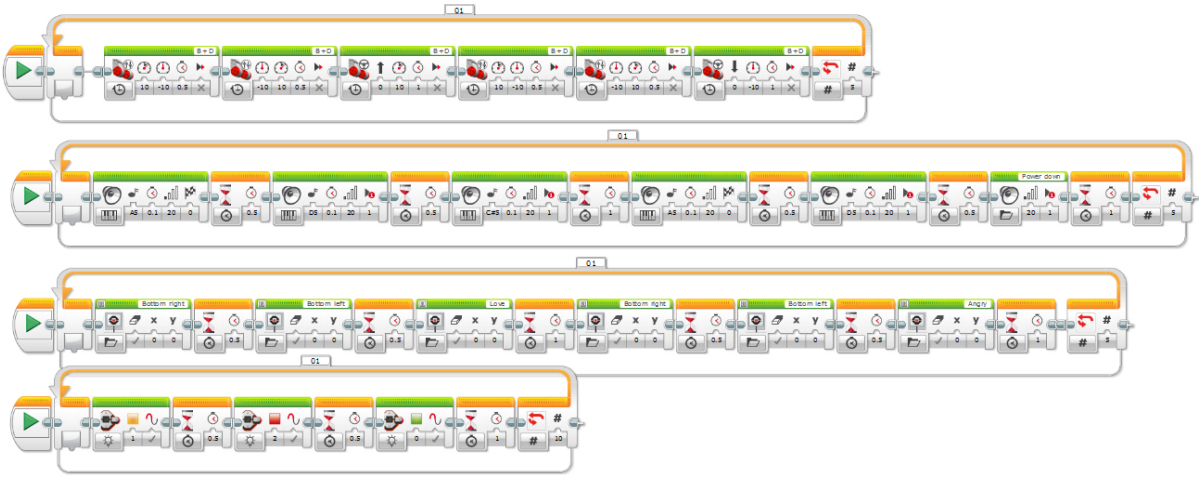
Değerlendirme, öğrencileri sıkmadan, her bir soru için verilen cevaplar tatmin edici bir düzeye ulaşıncaya kadar devam ettirilebilir.

5. İLAVE ETKİNLİK

5.1. Dans Eden Işıklı Robot

Bu ilave etkinlikteki amaç farklı şekillerde hareket eden, nota çalan, farklı şekillerde ışık yakan ve ekranında uygun ifade gösteren robotu programlamaktır. Bu program yukarıdaki etkinliğin devamı şeklinde yapılacaktır. Aşağıdaki resimde nota çalıp dans eden (sağ, sol, ileri, sağ, sol, geri hareket eden), ışık yakıp söndüren ve göz hareketleri ile sağa veya sola dönüşü aynı anda yapan, ileri gidince mutlu ifade, geri gidince ekranında mutsuz ifade gösteren robotu programlamak için gerekli kodlar verilmiştir. Bu kod blokları rehber öğretmen için örnek bir dans stilidir (Bu ve benzeri programların kodları rehber öğretmenlerle paylaşılacaktır).

.



Resim 46. Örnek Program

3. Hafta: Robotlarla Dokunma ve Açı Sensörü

Ön Bilgi:

- Öğrenciler robot setiyle farklı robotik tasarımlar yapmıştır.
- Öğrenciler robot setini programlamak için grafik arayüzünü kullanmıştır.
- Öğrenciler robotun hareket etmesi için gerekli programlama adımlarını oluşturmuştur.
- Öğrenciler akıllı tuğlanın ekranını düzenlemek ve tuğlanın çeşitli sesler çıkarmasını sağlamak için gerekli programlama adımlarını oluşturur.
- Öğrenciler robotu programlarken döngü mantığını kurup uygulayabilir.

Haftanın Kazanımları:

- Öğrenciler sensörleri tanır.
- Öğrenciler dokunma (touch) ve açı (gyro) sensörünün çalışma mantığını ve kullanma yöntemlerini bilir.
- Öğrenciler dokunma ve açı sensörlerinin kullanıldığı farklı uygulamaların programlama adımlarını oluşturur.
- Öğrenciler bir nesneye çarptıktan sonra robotun yeni bir hareket yapması için gerekli programlama adımlarını oluşturur.
- Öğrenciler robotun farklı uzaklıklardaki nesnelere göre davranması (hareket etmesi, ses çıkarma hızını ayarlaması, tuğla ekranında farklı simgeler göstermesi) için gerekli programlama adımlarını oluşturur.

Haftanın Amacı:

Bu haftaya başlarken duyu organlarımız ile analoji yapılarak sensör kavramına giriş yapılır. Robot setiyle birlikte gelen sensörler (dokunma, mesafe, açı ve renk gibi) kısaca tanıtılır. Daha sonra öğrencilerin dokunma ve açı sensörlerini verilen problemlerin çözümünde kullanması sağlanır.

Kullanılacak Malzemeler:

Robot seti, sensörler, bilgisayar.

Haftanın İşlenişi:

Gözle: Dokunma sensörünü kullanarak robotun harekete başlamasını sağlama, hareket hâlindeki bir robotu durdurma ya da açı sensörü ile robotu yönlendirerek hareket ettirme, açı sensörünü kalibre etme.
Uygula: Dokunma ve açı sensörlerini kullanarak farklı uygulamalar geliştirme ve her bir bloğun çeşitli bileşenlerini programlayarak uygulama
Tasarla: Dokunma ve açı sensörlerini kullanarak robotun düz gitmesini sağlayacak robot tasarımı için gerekli bileşenleri tanımlama ve planlama
Üret: Verilen görevlere göre robotu programlama
Değerlendir: Haftanın içeriği ile ilgili yansıtma etkinliği

1. GÖZLE VE UYGULA

1.1. Gözle: Sensör Nedir?

Robotların en önemli özelliklerinden biri algılamadır. Sensörler de bu amaçla kullanılan algılayıcılardır. Başka bir ifadeyle robotlar çevreden veri toplayan, bu veriyi işlemcileri ile işleyen, eyleme karar veren ve eylemi gerçekleştiren makinelerdir. Robotlar çevrelerindeki dünyayı algılamak için sensörlerini kullanırlar.

Rehber öğretmen sensörleri basitçe tanımladıktan sonra robotların sensörleri neden kullanması gerektiğini açıklamak adına: "Hareket eden bir robotun önündeki engelden kaçabilmesi için etrafındaki cisimleri algılayabilen bir sensöre ihtiyacı vardır." gibi farklı örnekler verebilir. Ayrıca konunun daha iyi anlaşılmasını sağlamak için insanların duyularıyla robotların sensörlerini birbiriyle ilişkilendirebilir. İnsanların görme, işitme, dokunma, koku ve tat alma gibi duyularını çeşitli amaçlarla kullanması gibi robotlar da duyuları elektronik sisteme aktarabilmek ve bunların sonucunda bir işlemi gerçekleştirmek için sensörlerini kullanılır. Yani "Robotların duyu organları sensörleridir." denilebilir.

Konuyu pekiştirmek için robotlarda sensörlerin kullanımına yönelik aşağıdaki linkte önerilen örnek videolar izlenebilir:

- Google's 5 Bets on the Future: Robots and Sensors: https://www.youtube.com/watch?v=dmtcug2ptg4
- Gesture Sensor Intelligent Control Programming Dancing Walking: https://www.youtube.com/watch?v=2QGGF4cXZ2I

Eğitim boyunca EV3 robot setinde bulunan dört temel sensör sırasıyla anlatılacaktır: touch (dokunma), ultrasonik (mesafe), jiroskop (açı) ve colour (renk). Sensörler öğrencilere gösterilerek kısaca bilgi verilir.

Bu hafta dokunma sensörü tanıtılacaktır. Dokunma sensörü, düğmesine ne zaman basıldığını ya da bırakıldığını, düğmeye bir defa mı yoksa birden fazla mı basıldığını algılayabilen basit ama son derece hassas bir araçtır. Öğrenciler "başla" ve "durdur" gibi basit komutlar vererek robotları bu sensör aracılığı ile hareket ettirebilecekleri gibi labirent benzeri karmaşık ortamlarda robotlarını dokunma sensörü ile yönlendirebilirler.

Resim 47. Dokunma Sensörü

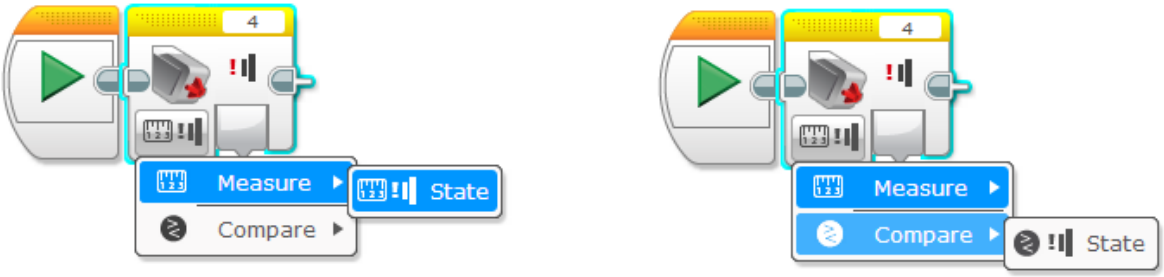
Gyro veya jiroskop olarak da adlandırılan açı sensörü ile robotun hareketi, hareket yönü veya eğimi algılanabilir. Öğrenciler bu sensörle açıları ölçerek navigasyon sistemlerinden oyun

kumandalarına kadar çeşitli araçlarda kullanılan teknolojileri keşfedebilir. Açı sensörünün temel özellikleri şöyle sıralanabilir:

- Üzerindeki ok yönündeki dönüşleri algılar,
- +/-3 derecelik bir hassasiyetle açıları ölçer,
- Saniyede maksimum 440 derecelik açı değişimini algılar,
- Yenileme hızı 1kHz'dir.

Resim 48. Açı Sensörü

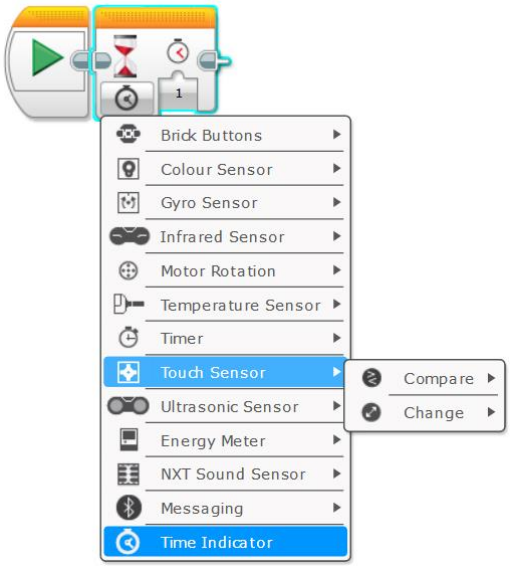
Bu aşamada EV3 yazılımının Sensor sekmesinde bulunan dokunma sensörü komutu incelenir. Dokunma sensörü varsayılan olarak 1. porta takılır. Eğer sensör robot üzerinde farklı bir porta bağlanmış ise, kod bloğunun sağ üst köşesinde yer alan port sekmesinden bağlı olduğu port değiştirilir. Dokunma sensöründe temel olarak Measure (ölçüm) ve Compare (karşılaştır) olmak üzere iki menü bulunur.



Resim 49. Dokunma Sensörü Ayarları

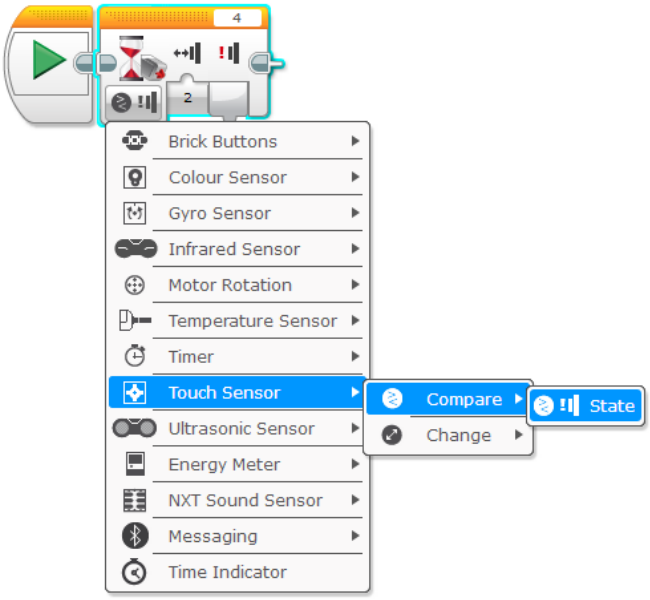
Ölçüm özelliğiyle sensör kontrol edilir. Dokunma sensörünün "bırakıldı" (0), "basıldı" (1) ve "basıldı-bırakıldı" (2) durumları ölçülür ve bu değerler başka bloklara aktarılabilir. Karşılaştır özelliğiyle ise sensörden ölçülen değerin, var olan üç durum ile karşılaştırması yapılır ve elde edilen Doğru/Yanlış (True/False) mantık sonucu başka bloklara aktarılabilir.

Sensörlerle ilgili uygulamalara başlamadan önce "akış kontrolü (flow control)" sekmesinden "bekle (wait)" bloğu tanıtılır. Bekle bloğu, EV3 yazılımındaki en basit bloklardan biridir ve en basit akış kontrol yapılarından birini sağlar. Temel işlevi, bir programın yürütülmesini belirli bir koşul sağlanana kadar duraklatmak ve ardından programı devam ettirmektir. Kum saati ikonu ile gösterilen bekle bloğu, sensörler arasında veri akışının sağlanmasında kullanılır. Örneğin bekle bloğu bir sensörün belirli bir değere ulaşması veya bir sensörün değerinin değişmesi için bir süre beklenmesini sağlar.



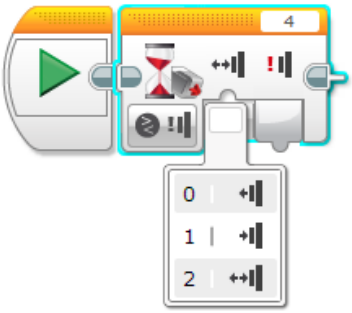
Resim 50. Bekle/Wait Bloğu

Bu aşamada, dokunma sensörüne dokunulunca harekete geçen bir robot programlanır. Bu program için Akış Kontrolü (Flow Control) sekmesinden "bekle (wait)" bloğu ve Eylem (Action) sekmesinden "direksiyon hareketi (move steering)" bloğu kullanılır. İlk önce bekle bloğu program alanına sürüklenip bırakılır. Daha sonra resimde görüldüğü gibi "Touch Sensor>Compare> State" modu seçilir.



Resim 51. Bekle/Wait Bloğu Dokunma Sensörü State Modu

Dokunma sensörünün üç durumu olduğu burada gösterilir.



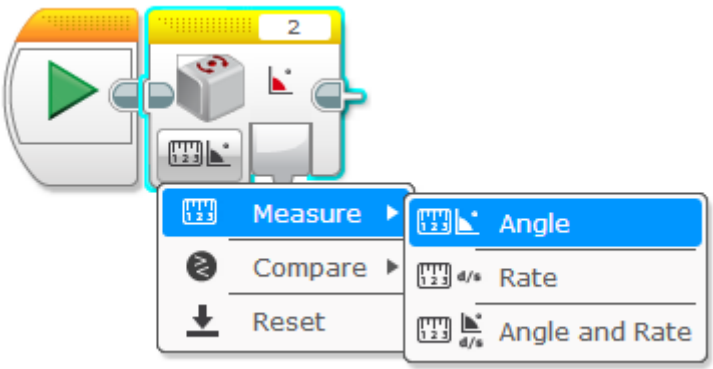
Resim 52. Dokunma Sensörü Ayarları

Blok üzerinde yer alan butona tıklandığında üç farklı fonksiyon bulunduğu görülür:
0 – Bırakıldığında
1 – Basıldığında
2 – Basılıp bırakıldığında

Bekle bloğunun dokunma sensörü butonuna tıklanır ve durumu 2 yani "basılıp bırakıldığında" olacak şeklinde ayarlanır. Daha sonra "direksiyon hareketi (move steering)" bloğu sürüklenip program alanına eklenir ve "motor 5 sn boyunca çalışsın ve sonra da dursun" olacak şekilde ayarlanır. Öğrencilere robotun dokunma sensörüne dokundukları anda harekete geçmeyeceği, ellerini çektikten sonra harekete geçeceği vurgulanır.

Resim 53. Örnek Program

Bu bölümde EV3 yazılımının sensor sekmesinde bulunan açı sensörü komutu incelenecektir. Aşağıdaki görselde görüldüğü üzere açı sensörü ikinci porta takılmıştır. Açı sensöründe "measure (ölçüm)", "compare (karşılaştır)" ve "reset (sıfırla)" olmak üzere üç mod bulunur.

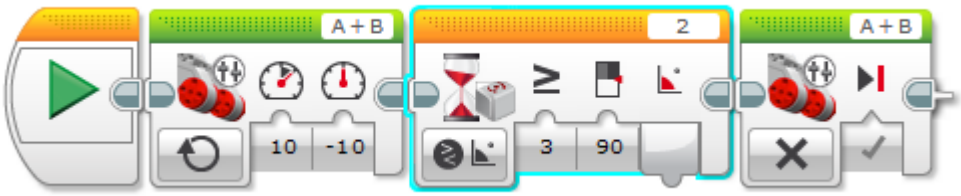


Resim 54. Açı Sensörü Ayarları

Ölçüm özelliğiyle sensörün durumu kontrol edilebilir. Açı sensörü sıfırlandığı andan itibaren robotun saat ibresi yönünde veya saat ibresinin tersi yönünde yaptığı dönme miktarını açı derece cinsinden ölçer. Ayrıca saniyedeki açı değişim oranını (rate) hesaplayabilir. Bu iki ölçüm değeri istenilirse birlikte de alınabilir (Angle and Rate). Karşılaştır özelliği ile ölçülen

açı ve oran değerleri önceden belirlenmiş bir değer ile karşılaştırılır ve elde edilen mantık değeri "Doğru/Yanlış" (True/False) olarak başka bloklara aktarılabilir. Ayrıca açı sensörünün değerini sıfırlamak gerekirse "sıfırla" modu seçilir.

Bu aşamada açı sensörünü kullanarak robotu kodlama işlemine geçilir. Amaç, olduğu yerde 90 derece dönen bir robot yapmaktır. Bu program için Eylem (Action) sekmesinden "palet hareketi (move tank)", Akış Kontrolü (Flow Control) sekmesinden de "Bekle (Wait)" blokları kullanılır. İlk önce palet hareketi (move tank) bloğu program alanına sürüklenip bırakılır. Motor, açı sensöründen gelen değerlere göre hareket edeceği için özelliği "on" olarak değiştirilir. Sonra bekle (wait) bloğu program alanına sürüklenip bırakılır. Ardından Gyro Sensor>Compare> Angle modu seçilir. Son olarak bir tane daha palet hareketi (move tank) bloğu programlama alanına sürüklenip bırakılır ve özelliği "off" olacak şekilde değiştirilerek, motorlar durdurulur. Bu aşamadan sonra uygulamaya geçilir.



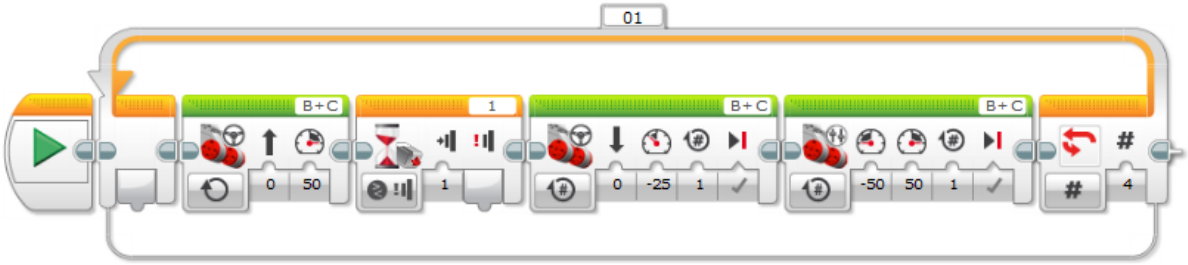
Resim 55. Örnek Program

1.2. Uygula: Engele Çarpınca Geri Dönen Robot

Öğrencilerden engele çarpınca geri dönen bir robot programlamaları istenir. Bunun için aşağıdaki adımlar takip edilir:

(i) Öğrencilerden robotun bir engele çarpıncaya kadar düz gitmesi istenir.
(ii) Robotun engele çarpınca biraz geri gitmesinin programa eklenmesi istenir.
(iii) Robotun kendi etrafında dönmesinin programa eklenmesi istenir.
(iv) Programın bu görevler dört defa gerçekleştirilecek şekilde oluşturulması istenir.

Bu arada bir işlemin belirli sayıda, sonsuza kadar veya belirli şartlar gerçekleştiğinde devam etmesi istenildiğinde Döngü (Loop) kavramının kullanıldığından tekrar bahsedilir. Bu uygulamada robot hareket ederken engele çarparsa dönüş yapar, sonra hareket etmeye devam eder, yine engele çarparsa tekrar dönüş yapar ve bu işlemi dört kez tekrarlar.



Resim 56. Örnek Program

Bu süreçte, rehber öğretmen sınıfta dolaşarak öğrencilere yardımcı olur. Yukarıdaki şekilde gösterilen programı oluşturmaları için öğrencileri yönlendirir. Fakat öğrencilerden gelen farklı ve mantıklı fikirleri de değerlendirir ve eğer uygunsa öğrencileri kendi fikirleri doğrultusunda programı oluşturmaları için cesaretlendirir.

1.3. Uygula: 360 Derece Dönen Robot

Bu uygulamanın adımları aşağıdaki gibi olabilir:

(i) Rehber öğretmen, öğrencilerden, mat üzerinde yer alan açıölçeri kullanarak robotlarının olduğu yerde 360 derece dönmesini sağlayan programı oluşturmalarını ister.

(ii) Rehber öğretmen dolaşarak öğrencilere yardımcı olur. Aşağıdaki görselde gösterilen programı oluşturmaları için öğrencileri yönlendirir. Fakat öğrencilerden gelen farklı ve mantıklı fikirleri de değerlendirir ve uygunsa öğrencileri kendi fikirleri doğrultusunda programı oluşturmaları için cesaretlendirir.

Resim 57. Örnek Program

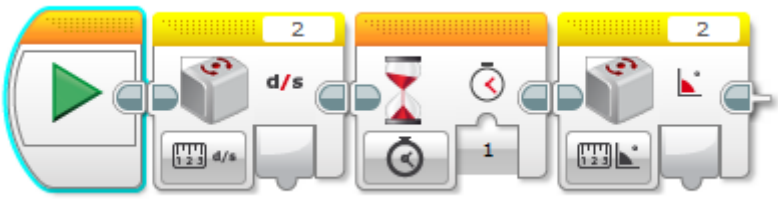
(iii) Öğrencilerden programın sonucunu değerlendirmeleri istenir. Genel olarak robotun 360 dereceden biraz daha fazla döndüğü görülür. Robotun neden tam olarak 360 derece dönmediği tartışılır. Robotun 360 dereceden daha fazla dönmesinin nedenleri şunlar olabilir:

(a) Açı sensörü hassasiyeti 360 derece için +/-3 derecedir. Yani sensör yeterince hassas değildir ve robotun dönme açısı 357 ile 363 derece arasında değişebilir.
(b) Sensör tam olarak 360 dereceyi ölçtüğü anda bu verinin aktarılması ve motorlara durma komutunun gönderilmesi zaman alır. Bu nedenle robot 360 dereceden fazla döner.
(c) Robot kendi etrafında dönerken oluşan dönme kuvveti, motorlar durdurulduktan sonra da robotun biraz daha dönmesine sebep olur.

(iv) Öğrencilerin bu sonuçlara ulaşabilmesi için rehber öğretmen tartışma ortamı oluşturur ve öğrencileri bu sonuçlara ulaştıracak sorularla tartışmayı yönlendirir. Sonrasında robotun tam

360 derece dönebilmesi için ne yapılması gerektiği tartışılır. Örneğin, robotun 360 dereceden daha az bir açı ile dönecek şekilde programlanması bir çözüm olabilir.

Bu aşamada, açı sensöründe bazı kaymalar oluyorsa öğrencilere, sensörün kalibre edilmesi gerektiği söylenir. Açı sensörü, robotla olan bağlantısı koparılıp tekrar kablosu takılarak veya port modu değiştirilerek kalibre edilir. Açı sensörünü kalibre etmek için aşağıdaki resimde sunulan program da çalıştırılabilir. Programda açı sensörü önce "oran" (rate) moduna alınır ve bir saniye sonra "derece " moduna alınır. Böylece açı sensörü sıfırlanmış (kalibre edilmiş) olur.



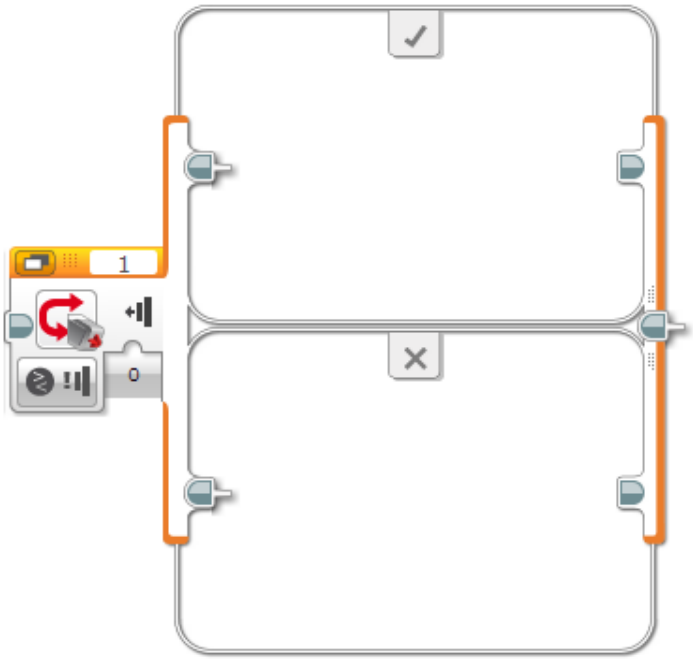
Resim 58. Açı Sensörü Kalibrasyonu

Not
Öğrencilerden robot tasarımlarına dokunma ve açı sensörlerini eklemeleri istenir. Bunun için gerekli yönergeye dokunma sensörü için aşağıdaki yollarla ulaşılabilir: i) https://le-www-live-s.legocdn.com/sc/media/lessons/mindstorms-ev3/building-instructions/ev3-touch-sensor-driving-base-4b82858ad3054e725caf23fffde42194.pdf ii)EV3 yazılımı > Lobby > Building Instructions > Building Ideas > Touch Sensor Driving Base iii) Robot setiyle birlikte gelen kitapçığa bakılabilir. Açı sensörüne aşağıdaki yollarla ulaşılabilir: i) https://le-www-live-s.legocdn.com/sc/media/lessons/mindstorms-ev3/building-instructions/ev3-gyro-sensor-driving-base-a521f8ebe355c281c006418395309e15.pdf ii)EV3 yazılımı> Lobby> Building Instructions> Building Ideas> Gyro Sensor Driving Base iii) Robot setiyle birlikte gelen kitapçığa bakılabilir.

1.4. Gözle: Anahtar (Switch) Bloğu

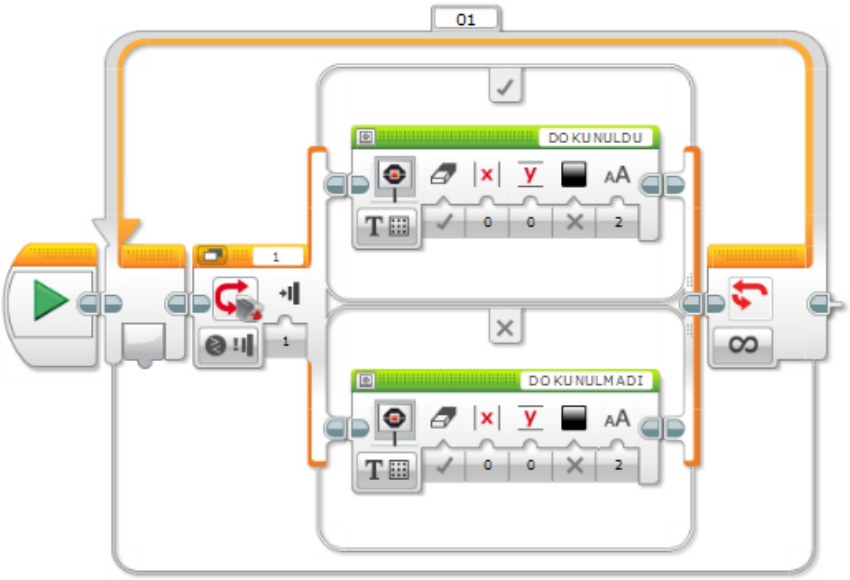
Öğrencilere, programlama aşamasında belirli işlerin çeşitli koşulların sağlanması durumunda gerçekleşmesinin istenebileceğinden bahsedilir. Örneğin hareket ederken hızlanan bir robotun belirli bir hızdan sonra ses çıkarması, ışık yayması ya da algıladığı renge göre farklı işlemler yapması gibi. EV3 yazılımının "akış kontrolü (flow control)" sekmesindeki "anahtar (switch)" bloğu bu işlemler için kullanılır. Aşağıdaki şekilde görüldüğü gibi "doğru/yanlış"

durumuna göre üstteki ya da alttaki işlemler gerçekleştirilir.



Resim 59. Koşul/Switch Bloğu

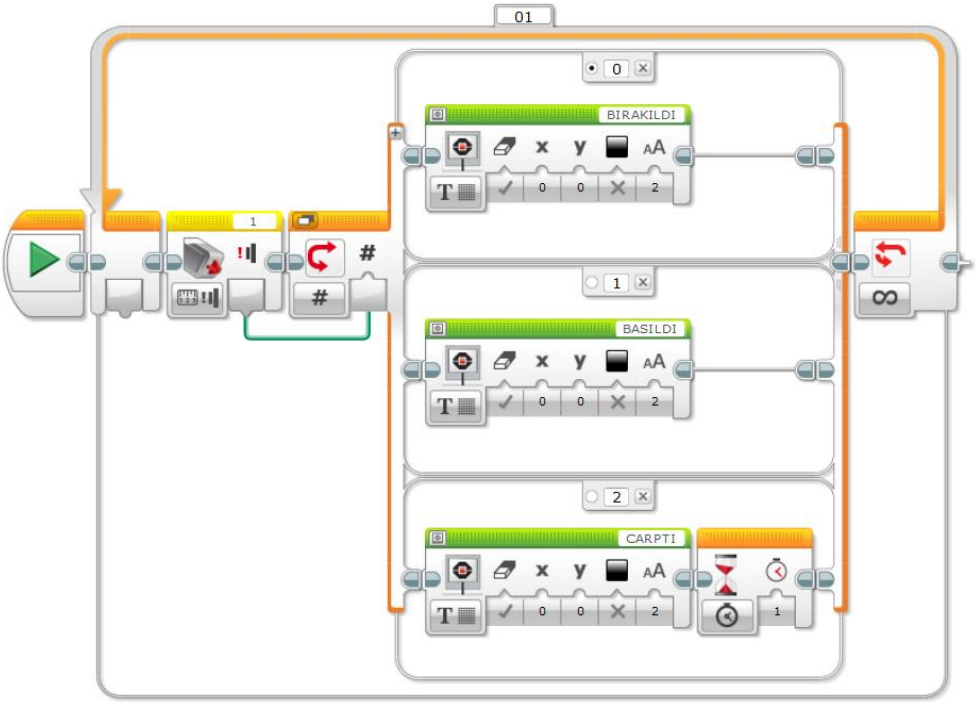
Switch bloğu kullanılarak robot programlanmaya başlanır. Amaç, dokunma sensörüne basılıp basılmadığını ekranda gösteren bir program oluşturmaktır. Bu program için Eylem (Action) sekmesinden "ekran (display)" bloğu, Akış Kontrolü (Flow Control) sekmesinden "döngü (loop)" ve "anahtar (switch)" blokları kullanılır. İlk önce döngü ve anahtar blokları program alanına sürüklenip bırakılır. Daha sonra koşulun doğru ve yanlış olma durumuna göre ekran "display" blokları "DOKUNULDU" ve "DOKUNULMADI" olarak ayarlanır. Öğrencilerin robotun dokunma sensörüne basılıp basılmadığını yazdıran durumu görmesi sağlanır.



Resim 60. Örnek Program

1.5. Uygula: Koşul Durumunu Göster

Öğrencilere dokunma sensörünün üç durumu tekrar hatırlatılır. Daha sonra ekranda "BASILDI", "BIRAKILDI" ve "ÇARPTI" ifadelerini gösteren bir program oluşturmaları istenir.



Resim 61. Örnek Program

2. TASARLA

2.1. Dokunma Sensörüne Basılınca Dönen Robot

Bu etkinlikte öğrencilere iki dokunma sensörüyle açı sensörünü tek bir robotta kullanacakları söylenir. Dokunma sensörleri robotun sağ ve sol kısımlarına yerleştirilecek şekilde tasarlanabilir. Bu etkinlikte amaç, hangi dokunma sensörüne dokunulursa robotun o yöne dönmesini ve basılı tutma süresi kaç saniye ise 10 katı dereceyle dönmesini sağlamaktır.

Tanımlama: Öğrenciler öncelikle gerekli işlemleri maddeler hâlinde yazmalıdır. Robotun nasıl tasarlanacağı, sensörlerin neler yapacağı, robotun hareket ederken nasıl davranacağı öğrenciler tarafından belirlenmelidir.

Örneğin;

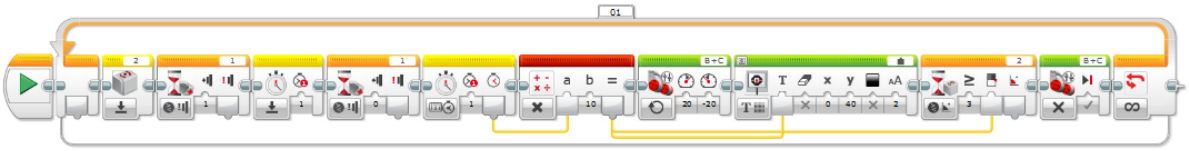
- Robot sürekli hareket edecek,
- Sensöre dokunulup dokunulmadığını kontrol edecek,
- Sensöre dokunulursa dokunulan sensör yönünde basılı tutma süresinin 10 katı dereceyle dönecek.

Fikir üretme: Bu aşamada öğrenciler tanımlamada belirlenen işlemlerin nasıl yapılabileceği ile ilgili fikir yürütmelidir. Örnek olarak öğrenciler aşağıdaki maddelere benzer fikirler üretebilir:

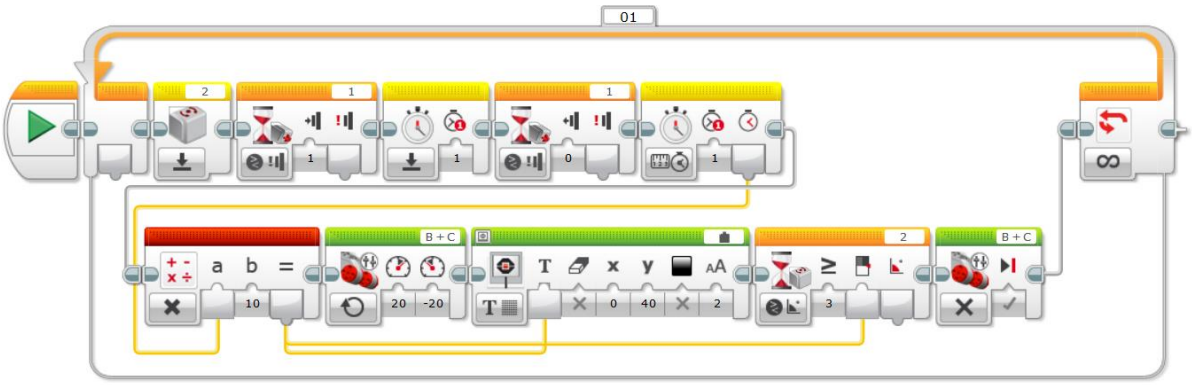
- Öncelikle robotun sensörleri hazır hâle getirilmelidir. Açı sensörü kalibre edilmeli, “wait (bekle)” bloğunun nasıl kullanılacağı tasarlanmalıdır.
- Robotu hareket ettirecek bloklara karar verilir. Örneğin “move tank” bloğu kullanılabilir.
- Dokunma sensörüne basılıp basılmadığı kontrol edilmeli, “switch” bloğu kullanılmalıdır.
- “Timer” bloğu ile ne kadar süre basılı tutulduğu hesaplanır.
- “Timer” bloğundan gelen veri 10 ile çarpılarak açı sensörüne gönderilir.
- Bu adımlar “loop” bloğu kullanılarak sürekli tekrar edilir.

3. ÜRET

Öğrenciler çözüm tasarımlarını yaptıktan sonra bilgisayar ve robot başında çalışarak istenilen görevi yerine getirir. Program rehber öğretmenlere verilecektir, örnek çözümün bir kısmı aşağıda görülebilir.



Resim 62. Örnek Program Bölüm 1



Resim 63. Örnek Program Bölüm 2

4. DEĞERLENDİR

Burada hedef, öğrencilerin öğrenme sürecinde yaşadıkları ve öğrendikleri üzerine düşünmelerini sağlamaktır. Bu sayede öğrencilerin problem çözme yetenekleri gelişecek, öğrenciler dersin konusu ve kendisi ile ilgili gözlemler yaparak yeni bilgiler öğrenme, kendilerini değerlendirme ve sonraki çalışmalarını planlama açısından fırsatlar elde edecektir. Öğrencilerden şu soruları cevaplamaları istenebilir:

- Bugün yapılan etkinlikte sizce neden motorlar için “on” durumu kullanıldı?
- Neden “switch” bloğu kullanıldı?

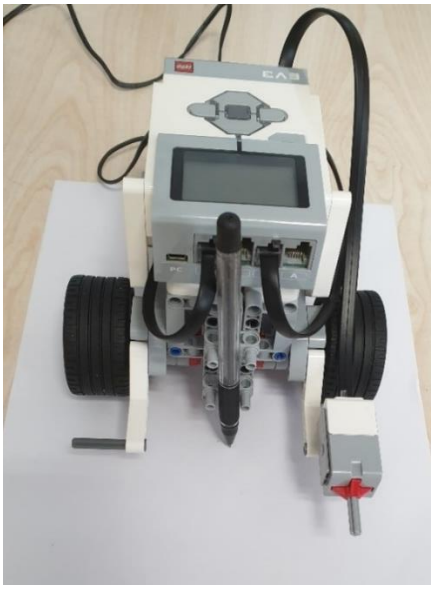
- Dokunma sensörünün üç durumu nelerdir? Bunları kıyaslayınız.
- Dokunma sensörünün günlük hayatta kullanım örnekleri nelerdir? (Örneğin arabanın kapısı açık kaldığında uyarı amacıyla ses vermesi)
- Açı sensörü ile başka neler yapılabilir, günlük hayatta hangi alanlarda kullanılabilir?

Değerlendirme, öğrencileri sıkmadan, her bir soru için verilen cevaplar tatmin edici bir düzeye ulaşıncaya kadar devam ettirilebilir.

5. İLAVE ETKİNLİK

5.1. Kalemle Zemine Çokgen Çizdirme

Bu ilave etkinlikte açı sensörü yardımıyla hareket eden bir robota kalem bağlanarak farklı şekiller çizmesi sağlanacaktır. Öncelikle robotun önüne setteki parçalar kullanılarak aşağıda yer alan resimdeki gibi bir kalem tutturulur. Robotun A4 boyutunda bir kâğıdın üzerinde hareket ederken istenilen geometrik şekli çizmesi istenir. Bu geometrik şekiller beşgen, altıgen, yedigen, sekizgen olabilir. Öğrencilere, açı sensörünü kullandıkları sırada ilgili düzgün çokgenin iç açılarının toplamını bulurken "(n-2) *180" formülünü kullanmaları gerektiği hatırlatılır.



Resim 64. Kalemin Robota Eklenmesi

5.2. Kalemle Zemine İstenilen Çokgeni Çizdirme

Etkinliklerin erken bitmesi durumunda, bir önceki etkinliğin geliştirilmesi istenebilir. Öğrencilerden robotun üzerinde bulunan düğmelerden yukarı düğmesine basıldığında üçgen, aşağı düğmesine basıldığında kare, sağ düğmesine basıldığında beşgen, sol düğmesine basıldığında altıgen çizen robotu programlamaları istenebilir. Ayrıca çizim esnasında ekrana çizilen çokgenin ismi yazılabilir.

4. Hafta: Robotlarla Mesafe Sensörü

Ön Bilgi:

- Öğrenciler temel düzeyde robotik kavramını bilir.
- Öğrenciler robot setiyle farklı robotik tasarımlar yapmıştır.
- Öğrenciler robot setini programlamak için grafik arayüzünü kullanmıştır.
- Öğrenciler robotun hareket etmesi için gerekli programlama adımlarını oluşturmuştur.
- Öğrenciler tuğlanın çeşitli sesler çıkarması için gerekli programlama adımlarını oluşturmuştur.
- Öğrenciler akıllı tuğla ekranının görüntüsünü düzenlemek için gerekli programlama adımlarını oluşturmuştur.
- Öğrenciler dokunma sensörünün kullanıldığı farklı uygulamaların programlama adımlarını oluşturmuştur.
- Öğrenciler açı sensörünün kullanıldığı farklı uygulamaların programlama adımlarını oluşturmuştur.

Haftanın Kazanımları:

- Öğrenciler, mesafe sensörünün çalışma mantığını ve kullanma yöntemlerini açıklar.
- Öğrenciler, mesafe sensörünün kullanıldığı farklı uygulamaların programlama adımlarını oluşturur.
- Öğrenciler, robotun farklı mesafelerdeki nesnelere göre davranması (hareket etmesi, ses çıkarması, hızını ayarlaması, tuğla ekranında farklı simgeler göstermesi) için gerekli programlama adımlarını oluşturur.

Haftanın Amacı:

Bu haftanın amacı, öncelikle öğrencilerin mesafe sensörünün (ultrasonic sensor) çeşitli kullanım şekillerini kavramalarını ve sensörü farklı görevler için programlarken gerekli düzenlemeleri yapabilmelerini sağlamaktır.

Kullanılacak Malzemeler:

Robot seti, bilgisayar, mesafe sensörü, mat (çalışma alanı).

Haftanın İşlenişi:

Gözle: Mesafe sensörünü kullanarak robota belirli mesafeye göre işlem yaptırma, robotu belirlenen mesafe kadar hareket ettirme, sensörü kullanarak teker yarıçapını ölçme.
Uygula: Her bir bloğun çeşitli bileşenlerini programlayarak uygulama.
Tasarla: Robotun istenilen işlemleri yapabilmesi için gerekli bileşenlerini tanımlama ve planlama.
Üret: Verilen görevleri programlama.
Değerlendir: Haftanın içeriği ile ilgili yansıtma etkinliği.

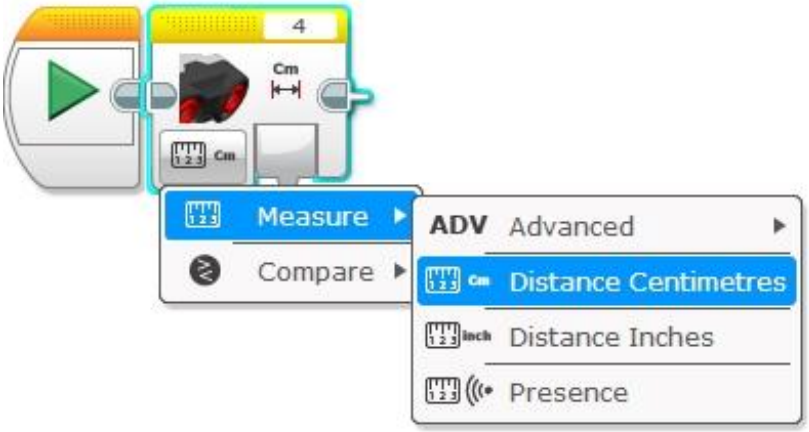
1. ADIM: GÖZLE ve UYGULA

1.1. Gözle: Mesafe Sensörü Nedir, Nasıl Kullanılır?

Mesafe sensörü (ultrasonic sensor) ileriye doğru yüksek frekanslı ses dalgaları gönderir ancak bunlar insanlar tarafından duyulamaz. Karşıda bulunan nesneye çarpan ses dalgaları geri döner ve mesafe sensörü tarafından geri alınır. Ses dalgasını gönderme ve alma arasında geçen süre kullanılarak mesafe sensörünün karşıdaki nesneye uzaklığı hesaplanır. Mesafe sensörü ile nesne arasındaki mesafenin başlangıç noktası sensör üzerindeki "gözler" değildir, mesafe resimde gösterilen noktadan başlanarak hesaplanır. Mesafe sensörü en fazla 255 cm uzaklıktaki cisimleri algılayabilir, daha uzaktaki cisimleri algılayamaz. Mesafe sensörüyle santimetre veya inç cinsinden uzaklık işlemleri yapılabilir.

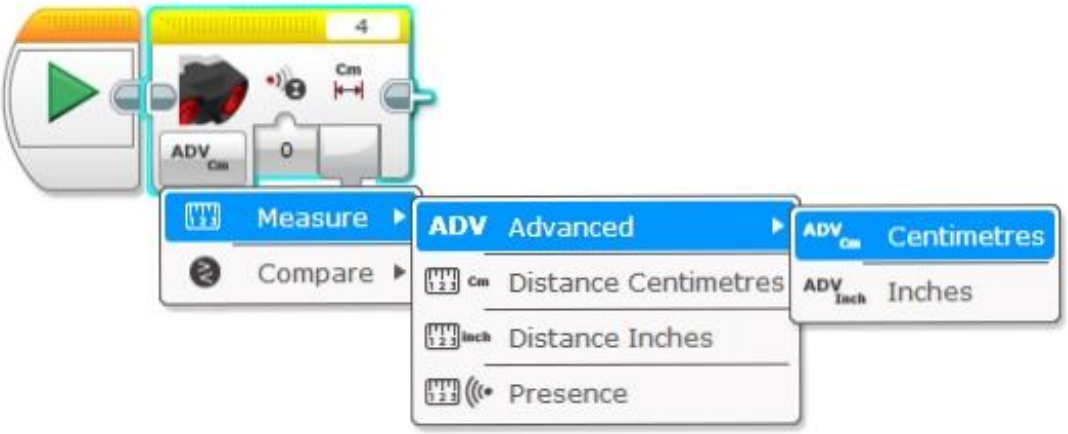
Resim 65. Mesafe Sensörü

EV3 yazılımında sensör sekmesinde "mesafe sensörü (ultrasonic sensor)" komutu bulunur. Aşağıda resimde görüldüğü üzere mesafe sensörü dördüncü porta takılır. Mesafe sensöründe temel olarak "Ölçüm (Measure)" ve "Karşılaştır (Compare)" olmak üzere iki menü bulunur.



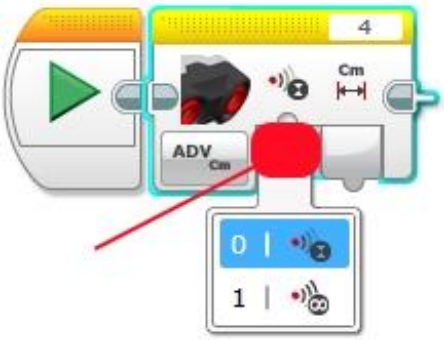
Resim 66. Mesafe Sensörü Bloğu

Ölçüm menüsü kullanılarak karşıda bulunan cisme santimetre cinsinden uzaklık (distance centimeters) ve inç cinsinden uzaklık (distance inches) belirlenebilir, ayrıca karşıda başka bir mesafe sensörünün olup olmadığı (presence) sorgulanabilir. Santimetre veya inç cinsinden hesaplamada uzaklık değeri sayısal olarak verilirken "presence" komutu ile doğru veya yanlış olmak üzere "boolean" bir değer üretilir. Karşıda başka bir mesafe sensörü varsa "doğru", yoksa "yanlış" değeri üretilir. "Gelişmiş (Advanced)" menüsü de aşağıda resimde görüldüğü gibi, santimetre ve inç cinsinden uzaklık ölçmek için kullanılır.



Resim 67. Mesafe Sensörü Ölçüm/Measure Seçenekleri

Gelişmiş (Advanced) menüsünden santimetre komutu verildiğinde aşağıdaki resimde görünen menü ortaya çıkar. Fare ile aşağıda kırmızı ile işaretlenmiş yere tıklandığında iki seçenek çıkar. Bunlar 0 (ping) ve 1'dir (continuous). 0 seçeneği ile karşıya sadece bir tane ses dalgası gönderilirken 1 seçeneği ile sürekli ses dalgası gönderilir. Bu seçenekler bazı durumlarda faydalı olabilir. Örneğin ortamda birden fazla mesafe sensörü varsa her iki sensörden gelen sürekli ses dalgaları yanlış hesaplamalara sebep olabilir. Bunu engellemek için sensörlerin sırayla birer tane dalga göndermesi sağlanabilir.



Resim 68. Advance Menüsü Seçenekleri

1.2. Gözle: Belirli Bir Mesafeye Kadar İlerleme

Bu program için Eylem (Action) sekmesinde bulunan Direksiyon Hareketi (Move Steering) ve Akış Kontrolü (Flow Control) menüsünde bulunan "Bekle (Wait)" komutları kullanılır. Direksiyon hareketi komutu robotun iki motorunun birlikte hareket ettirilmesi için kullanılır. "Bekle" komutu ise belirli bir olay gerçekleşinceye kadar belirli komutların çalıştırılmasını sağlar. Öncelikle robota mesafe sensörü takılır. Programda aşağıdaki resimde görülen komutlar oluşturulur.

Not

Mesafe sensörlü temel tasarım (ultrasonic sensor driving base) ile ilgili gerekli yönergeye aşağıdaki yollarla ulaşılabilir.

i) https://le-www-live-s.legocdn.com/sc/media/lessons/mindstorms-ev3/building-instructions/ev3-ultrasonic-sensor-driving-base-61ffdfa461aee2470b8ddbeab16e2070.pdf

ii) EV3 yazılımı > Lobby > Building Instructions > Building Ideas > Ultrasonic Sensor - Driving Base

iii) Robot setiyle birlikte gelen kitapçığa bakılabilir.

Resim 69. Örnek Program

Oluşturulan kod ile B ve C portlarına takılan büyük motor'ların 3 numaralı porta takılan mesafe sensöründen gelen uzaklık değeri 5 cm'den küçük oluncaya kadar çalıştırılıp mesafe 5 cm'den küçük olduktan sonra durması sağlanır. Motorlar 50 hızıyla hareket edip doğruca karşıya ilerler.

Bu adımda programın nasıl yazıldığı ekrana yansıtılarak öğrencilere gösterilir ve programın çalışma mantığı öğrencilere açıklanır.

Öğrencilerden aşağıdaki adımları yapmaları istenir (**Dikkat:** Öğrenciler yazdıkları kodu doğrudan çalıştırmamalıdır. Bu durum çeşitli problemlere neden olabilir. Bunun yerine öğrenciler yukarıdaki resimde görüldüğü gibi önce programı akıllı tuğlaya indirmeli ve sonra programı akıllı tuğla üzerinden çalıştırmalıdır).

(i) Yukarıdaki kod EV3 yazılımında oluşturulur ve robot çalıştırılır.

(ii) Kodun çalışması bitince nesne ile mesafe sensörü arasındaki mesafe ölçülür.

(iii) Öğrencilerden buldukları değer ile programda yazılı olan mesafe değerini (ilk örnek için 5 cm) bir kâğıda yazmaları istenir.

(iv) Öğrencilerden i, ii ve iii numaralı adımları 7 cm, 10 cm ve 15 cm için tekrarlamaları istenir.

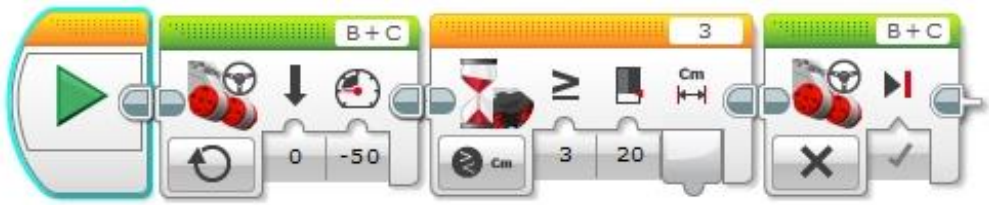
(v) Öğrencilere buldukları verilerin ne anlama geldiği sorulur.

(vi) Öğrencilere robotun tam olarak verilen değerde duramayacağı ve hata payı bulunduğu için çıkan mesafe değerinin verilen değerlerden farklı çıktığı açıklanır.

1.2. Uygula: İstenilen Mesafe Kadar Geri Gitme

(i) Öğrencilerden mesafe sensörünü kullanarak, robotlarının mesafe sensörünün önünde bulunan engelden 20 cm geriye gittikten sonra durmasını sağlayacak programı oluşturmaları istenir.

(ii) Rehber öğretmen sınıfta dolaşarak öğrencilere yardımcı olur. Aşağıdaki resimde gösterilen programı oluşturmaları için öğrencileri yönlendirir. Fakat rehber öğretmen, öğrencilerden gelen farklı ve mantıklı fikirleri de değerlendirir ve eğer uygun ise kendi fikirleri doğrultusunda programı oluşturmaları için öğrencileri cesaretlendirir.



Resim 70. Örnek Program

(iii) Öğrencilerden programı çalıştırdıktan sonra çıkan mesafeyi ölçmeleri istenir. Bu mesafeyi 20 cm'ye olabildiğince yaklaştırmaları için programda değişiklik yapmaları istenir.

1.3. Gözle: Tekerin Yarıçapını Hesaplama

Öğrencilere bir dairenin yarıçapı ile çevresi arasındaki ilişki sorulur. Öğrenciler "çevre=2π*r" formülünü biliyorsa bir sonraki adıma geçilir. Eğer bilmiyorlarsa formülün ne anlama geldiği açıklanır.

Öğrencilere tekerin yarıçapının bulunması için aşağıdaki algoritmanın kullanılabileceği anlatılır:

(d) Karşıda bulunan engel ile aradaki mesafe ölçülür,
(e) Tekerler bir tur ileri döndürülür,
(f) Mesafe tekrar ölçülür ve fark hesaplanır,
(g) Fark 2π'ye (π 3.141 alınacaktır) bölünür ve yaklaşık yarıçap değeri bulunur.

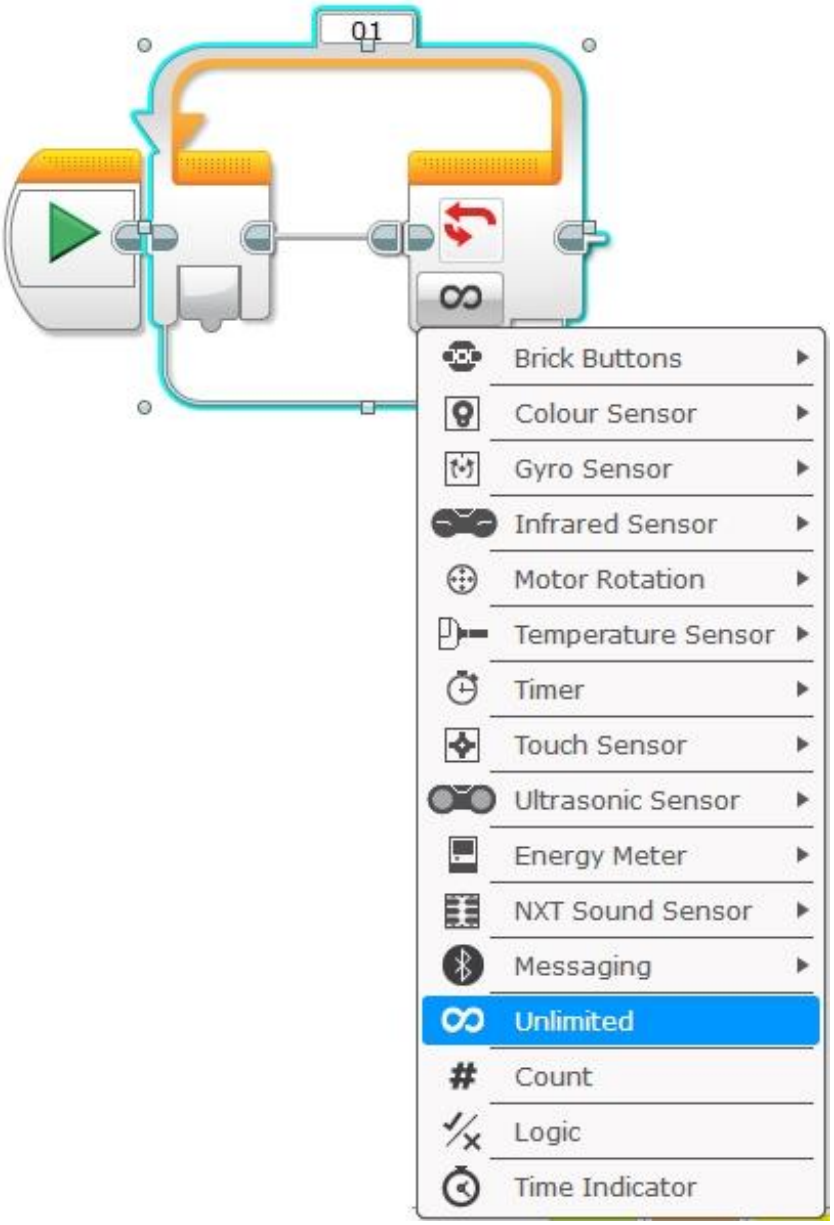
1.4. Uygula: Tekerin Yarıçapını Hesaplama

Öğrencilerden tekerin yarıçapını oluşturacakları program ile hesaplamaları istenir (**Not:** Yarıçap yaklaşık 28 milimetre olarak bulunmalıdır).

Öğrenciler yarıçapı kendileri bulamazlarsa rehber öğretmen tüm sınıf karşısında program yardımıyla yarıçapı hesaplar.

1.5. Uygula: Engele Yaklaştıkça Yavaşlayan Robot

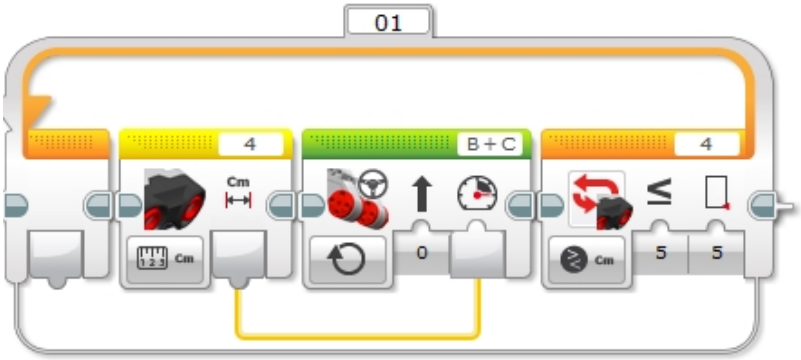
(i) Bu etkinlikte karşıda bulunan engele yaklaştıkça yavaşlayan ve engele 5 cm kaldığında duran bir robot programı yapılacaktır. Bu programın yazılması için turuncu Akış Kontrolü (Flow Control) sekmedeki "döngü (loop)" komutunun bilinmesi gerekir. Rehber öğretmen öğrencilere aşağıda resmi olan "döngü" komutunu gösterir ve seçenekleri hızlıca anlatır. Ses, metin ve grafik konularının işlendiği önceki haftalarda gösterilen "döngü komutu" burada da tekrar edilir.



Resim 71. Döngü Bloğu Seçenekleri

(i) Öğrencilere mesafe sensörü ile ölçülen uzaklık değerinin, robotun hareketini sağlayan motorların güç değeri olarak aktarıldığında, mesafe azaldıkça motorların gücünün de azalacağı ve böylece robotun engele yaklaştıkça yavaşlayacağı detaylıca anlatılır,

(ii) Aşağıdaki resimde görülen program oluşturularak çalışma mantığı öğrencilere anlatılır.



Resim 72. Örnek Program

2. TASARLA VE ÜRET

2.1. Birinci Görev: Öndeki Aracı Takip Eden Robot

Bu etkinlikte robotun önünde bulunan bir arabayı (araba olmak zorunda değil herhangi bir nesne de olabilir) takip etmesi sağlanır.

Öğrencilerden öndeki arabayı takip eden, yani öndeki araba ilerledikçe ilerleyen bir robotun nasıl programlanacağı üzerinde düşünmeleri istenir. Öğrenciler grup olarak tartışır. Gerektiğinde rehber öğretmen onlara yardımcı olabilir. Öğrencilere tam bir çözüm verilmemelidir. Gruplara çözümü kendilerinin üretmesi için zaman verilmelidir. Eğitim boyunca, tasarlama sürecinde olduğu gibi, öğrencilerin aşağıda örnek olarak verilen iki adıma benzer bir süreci gerçekleştirmesi gerekir.

Tanımlama: Öğrenciler öncelikle takip işlemi için neler gerektiğini belirlemeli ve gerekli işlemleri maddeler hâlinde yazmalıdır.

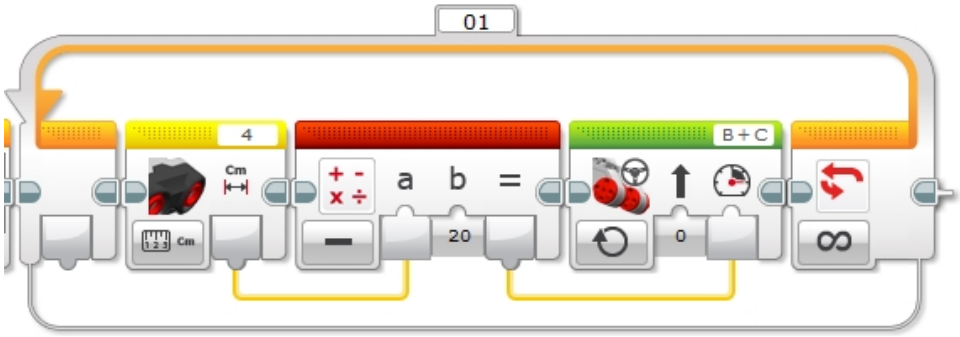
Örneğin;

- Robot öndeki araç ile arasındaki mesafeyi sürekli kontrol edecek,
- Öndeki araç hareket ederse hareket edecek,
- Öndeki araç hızlı hareket ediyorsa robot hızlanacak, yavaş hareket ediyorsa yavaşlayacak,
- Öndeki araç durunca duracak.

Fikir üretme: Bu aşamada öğrencilerin yukarıda belirlenen işlemlerin nasıl yapılabileceği ile ilgili fikir yürütmesi gerekir. Örneğin öğrenciler aşağıdaki maddelere benzer fikirler üretebilir:

- Öndeki araçla arasındaki mesafeyi sürekli kontrol etmek için mesafe sensörü ve döngü bloğu kullanılabilir.
- Öndeki araçla arasındaki mesafe 20 cm'den büyükse robot 10 hızında hareket ettirilir.
- Öndeki araçla arasındaki mesafe 40 cm'den büyükse robot 20 hızında hareket ettirilir.
- Öndeki araçla arasındaki mesafe 60 cm'den büyükse robot 50 hızında hareket ettirilir.
- Öndeki araçla arasındaki mesafe 80 cm'den büyükse robot 100 hızında hareket ettirilir.
- Öndeki araçla arasındaki mesafe 20 cm'den küçükse robot 10 hızında hareket ettirilir.
- Öndeki araç 10 cm'den yakınsa robot durur.

Temel olarak bu adımlarla robottan istenilen işlemler yapılabilir. Ama öğrenci isterse robotun öndeki araca daha yakın gitmesini veya öndeki aracın hızına daha fazla ayak uydurmasını sağlayabilir. Mesafeye bağlı olarak robotun yavaşlaması da sağlanabilir. Her grubun çözümü farklı olabilir, önemli olan bu fikirlerin gruplar tarafından ortaya konulması ve ortaya konulan fikirlerin problemin çözümünü sağlayabilmesidir. Yukarıda anlatılan algoritmadan farklı bir örnek çözüm aşağıdaki resimde verilmiştir. Rehber öğretmen öğrencilerden gelebilecek farklı görüş ve çözüm önerilerine açık olmalı ve onları bu konuda teşvik etmelidir.



Resim 73. Örnek Program

Öğrenciler çözüm için tasarımlarını yaptıktan sonra bilgisayar ve robot başında çalışarak istenilen görevi yerine getirir.

2.2. İkinci Görev: Takımlar Yarışıyor

Bu bölümde iki yarışma yapılır. Öğrenciler yarışmalara takım olarak katılır. Yarışmalardaki amaç, robotun ileride bulunan bir engele kadar ilerlemesi ve geri gelerek başlangıç noktasına vardığında durmasıdır. Başlangıç noktasından 1 metre uzağa engel konulur. Robotlar engele 10 cm mesafe kala -dönmeden- geriye doğru hareket ederek başladıkları yere geri gelirler. Robot engele 8 cm'den daha çok yaklaşırsa veya 12 cm'den daha uzak bir mesafedeyken geri gitmeye başlarsa yarışmacılar elenir. Ayrıca robot geri geldiğinde ilk başlangıç noktasını 2 cm'den fazla geçer ya da başlangıç noktasına 2 cm'den daha fazla mesafe varken durursa yarışmacılar yine elenir.

1. Yarışma: Bu yarışmada amaç başlangıç çizgisine en yakın noktada durmaktır. Programı oluşturmaya başlamadan önce grupların tasarlama adımı için yukarıda bir örneği verilen tanımlama ve fikir üretme sürecini gerçekleştirmesi gerekir. Bu yarışma için verilen süre rehber öğretmen tarafından belirlenir (Beş saniye genellikle yeterlidir). Yukarıda belirtilen kurallarla birlikte bu süreyi geçiren yarışmacılar elenmiş sayılır. Bütün grupların robotları işlemlerini bitirdiğinde başlangıç noktasına olan uzaklıkları ve işlemi tamamlama süreleri kayıt altına alınır. Başlangıç çizgisine en yakın noktada duran robot yarışmayı kazanır. Eşitlik durumunda hızlı gelen robot birinci sayılır.

2. Yarışma: Bu yarışmadaki amaç görevi en kısa sürede tamamlamaktır. Programı yazmaya başlamadan önce grupların tasarlama adımı için yine yukarıda bir örneği verilen tanımlama ve fikir üretme sürecini gerçekleştirmesi gerekir. Yarışma kurallarına uyan robotlardan görevi en hızlı tamamlayan birinci olur. Eşitlik durumunda başlangıç noktasına daha yakın olan robot birinci sayılır.

2.3. Park Sensörü

Öğrencilerden robotu araçlardaki park sensörüne benzer şekilde programlamaları istenir. Robotun belirli bir mesafeye gelince uyarı vermesi, mesafe azaldıkça daha kısa aralıklarla ses çıkarması ve mesafe arttıkça daha uzun aralıklarla ses çıkarması, tuğla ekranında mesafe ve uyarı ile paralel görseller yansıtması istenir.

Tasarla: Öğrencilerin programın adımlarını detaylı olarak planlamaları (mesafeleri belirlemeleri, belirlenen mesafelerdeki işlemleri tanımlamaları, kullanılacak sesleri ve görselleri seçmeleri, mesafeler değiştikçe robotun nasıl davranacağına karar vermeleri) ve bu ögelerin programla nasıl yapılabileceğini tasarlamaları gerekir. Programı oluşturmaya başlamadan önce grupların tasarlama adımı için yukarıda bir örneği verilen tanımlama ve fikir üretme sürecini gerçekleştirmeleri gerekir.

Üret: Bu bölümde de öğrenciler aktif rol üstlenir. Rehber öğretmen yalnızca yönlendirir ve öğrencilere takıldıkları noktalarda destek olur. Yani rehber öğretmen yalnızca ihtiyaç anında destek sağlamalıdır. Üret aşamasında, öğrencilerden bir önceki adımda tasarladıkları robot planını kullanarak probleme "algoritmik" -daha yapılandırılmış- bir çözüm önerisi geliştirmeleri istenir. Öğrenciler bilgisayar ve robot başında çalışarak gerekli yazılım çözümleri geliştirirler. Burada öğrencilerin en çok zorlanacağı konulardan biri ses, hareket, görüntü ve mesafe sensörlerinin senkronizasyonunu sağlamaktır.

3. ADIM: DEĞERLENDİR

Bu aşamada hedef, öğrencilerin öğrenme sürecinde yaşadıkları ve öğrendikleri üzerine düşünmesini sağlamaktır. Bu sayede öğrenciler problem çözme yetenekleri, dersin konusu ve kendisi ile ilgili gözlemler yaparak yeni bilgiler öğrenme, kendilerini değerlendirme ve sonraki çalışmalarını planlama açısından fırsatlar elde eder. Öğrencilerden şu soruları cevaplamaları istenebilir:

- Verilen problemi nasıl tanımlarsınız? (Problemi kendi cümleleri ile ifade etme)
- En çok hangi görevde zorlandınız? Bu zorlukların üstesinden nasıl geldiniz? (Problemin çözümü için hangi stratejileri kullandınız ve n eden bu stratejileri seçtiniz?) Yeteri kadar tartışma ortamı oluşmazsa rehber öğretmen aşağıdaki soruları kullanarak tartışma ortamı yaratmaya çalışır.
    - Öndeki aracı takip eden programda, robotun hareket etmeye başlamasını nasıl sağladınız?
    - Öndeki aracı takip eden programda, öndeki araç durunca robotun durmasını nasıl sağladınız?
    - İkinci görevde, belirlenen sürede robotu engele yaklaştırıp sonra başladığı noktaya geri getirmek için ne gibi çözümler denediniz?
- Problemi çözerken ne gibi sıkıntılar yaşadınız ve bunların üstesinden gelmek için neler yaptınız?
- Kullandığınız yöntemler bu sıkıntıları gidermede başarılı oldu mu?

- Grup arkadaşınızla fikir ayrılıkları yaşadınız mı? Bunların üstesinden gelmek için neler yaptınız?
- Grup arkadaşınızdan ne/neler öğrendiniz?

Değerlendirme, öğrencileri sıkmadan, her bir soru için verilen cevaplar tatmin edici bir düzeye ulaşıncaya kadar devam ettirilebilir.

5. Hafta: Robotlarla Renk Sensörü

Ön bilgi:

- Öğrenciler robot setiyle farklı robotik tasarımlar yapmıştır.
- Öğrenciler robotik setini programlamak için grafik arayüzünü kullanmıştır.
- Öğrenciler robotun hareket etmesi için gerekli programlama adımlarını oluşturmuştur.
- Öğrenciler tuğlanın çeşitli sesleri çıkarması için gerekli programlama adımlarını oluşturmuştur.
- Öğrenciler tuğla ekranının görüntüsünü düzenlemek için gerekli programlama adımlarını oluşturmuştur.
- Öğrenciler dokunma mesafe ve açı sensörlerini farklı amaçlar için programlama adımlarını oluşturmuştur.

Haftanın Kazanımları:

- Öğrenciler renk sensörünün çalışma mantığını ve kullanma yöntemlerini açıklar.
- Öğrenciler renk sensörünü farklı amaçlar için programlama adımlarını oluşturur.
- Öğrenciler robotun algıladığı renk, yansıyan ışık ve ortam ışığına göre davranması (hareket etmesi, ses çıkarması, tuğla ekranında farklı simgeler göstermesi) için gerekli programlama adımlarını oluşturur.

Haftanın Amacı:

Bu haftanın amacı, öncelikle öğrencilerin renk (colour) sensörünün çeşitli kullanım şekillerini (renk algılama, ortam ışığı ve yansıyan ışığını ölçme) kavrayarak sensörü çeşitli amaçlar için programlarken gerekli düzenlemeleri yapabilmesini sağlamaktır.

Kullanılacak Malzemeler:

Robot seti; bilgisayar; renk, açı, dokunma ve mesafe sensörleri; mat (çalışma alanı).

Haftanın İşlenişi:

Gözle: Renk sensörünü kullanarak renkleri, yansıyan ışığı ve ortam ışığını algılama; yansıyan ışığı kullanarak robotu hareket ettirme (izleri / yolları takip etme); robotun ortam ışığına göre tepki vermesini (ışığı açma, yavaşlama, ses çıkarma) sağlama; renk sensörünü diğer sensörlerle birlikte kullanma.

Uygula: Her bir komutun çeşitli bileşenlerini programlayarak uygulama.

Tasarla: Robotun istenilen işlemleri yapabilmesi için gerekli bileşenlerini tanımlama ve planlama.

Üret: Verilen görevleri programlama.

Değerlendir: Haftanın içeriği ile ilgili yansıtma etkinliği

1. GÖZLE VE UYGULA

1.1. Gözle: Renk Sensörü Nedir ve Nasıl Kullanılır?

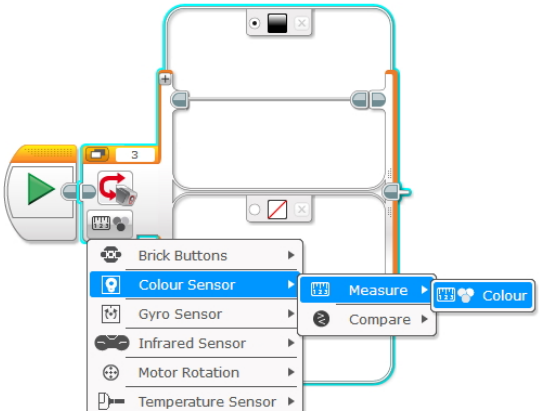
Robotlar soyut olan programları, insanların yaşadığı somut fiziksel ortamlarda çalıştırır. Dolayısıyla, fiziksel dünya ile etkileşim hâlindedirler. Fiziksel dünyada bir nesnenin rengi veya bir kaynaktan gelen ışığın şiddeti önemlidir. Örneğin, bir şoförün trafik ışıklarında ne yapması gerektiğini öğrenmesi için öncelikle renkleri algılayabilmesi gerekir. Başka bir örnek vermek gerekirse tablet, cep telefonu ve televizyon gibi cihazların ışık şiddeti uygun ayarlanmadığında bu cihazlardan gelen görüntünün algılanmasında zorluk yaşanabilir. Benzer şekilde robotlar için de renk çeşidi veya ışık şiddeti önemlidir. EV3 setinde bu iş için renk sensörü bulunur. Renk sensörü hem renk çeşidini hem de ışığın yoğunluğunu ölçmek için kullanılır. Renk sensörü "siyah" (1), "mavi" (2), "yeşil" (3), "sarı" (4), "kırmızı" (5), "beyaz" (6) ve "kahverengi" (7) renklerini algılayabilir. Bunun yanında karşısında hiçbir renk olmadığını da (0) algılayabilir. Burada renklerin yanına yazılan rakamlar, o renklerin kodlarıdır. Renk olmama durumu için "sıfır" kullanılır. Sensörün karşısında herhangi bir nesne yoksa sensör renk yok verisini iletir.

Resim 74. Renk Sensörü

Not
Renk sensörlü temel tasarıma (colour sensor down - driving base) ulaşmak için gerekli yönergeye aşağıdaki yollarla ulaşılabilir: i) https://le-www-live-s.legocdn.com/sc/media/lessons/mindstorms-ev3/building-instructions/ev3-rem-color-sensor-down-driving-base-d30ed30610c3d6647d56e17bc64cf6e2.pdf ii) Ev3 yazılımı > Lobby > Building Instructions > Building Ideas > Color Sensor Down - Driving Base iii) Robot setiyle birlikte gelen kitapçığa bakılabilir.

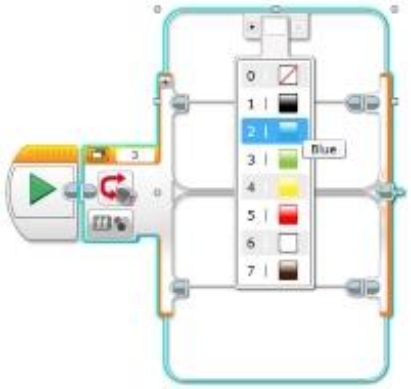
1.2. Uygula: Karşılaşılan Rengin İngilizcesini Söyleyen Program

Bu uygulamada, sabit duran bir robotun renk sensörüne gösterilen farklı cisimlerin renklerinin İngilizce adlarını söylemesini sağlayan bir program oluşturulacaktır. Bu programı oluşturmak için "loop"komutu içerisinde bir "switch" komutu kullanılmalıdır. Bu aşamada rehber öğretmen tarafından "switch" komutunun ne işe yaradığı ve renk sensörü ile nasıl kullanılabileceği kısaca anlatılır. "Switch" komutu belirli durum veya koşullarda belirli işlemlerin yapılmasını sağlamak için kullanılır. Örneğin, robotun mavi rengi gördüğünde "blue" (mavi) demesi istenebilir. Burada bir koşul vardır: "mavi olmak". Mavi olma koşulu sağlandığında akıllı tuğla "blue" diyen sesi oynatmalıdır. Aşağıda resimde bu kodun nasıl oluşturulduğu adım adım gösterilmektedir..



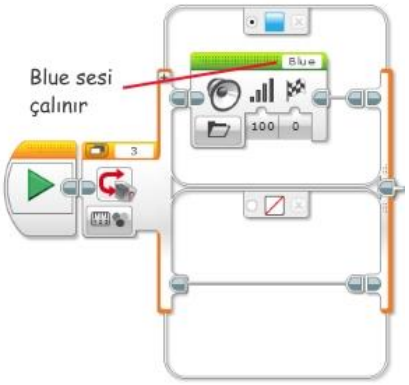
Resim 75. Adım 1

Switch içerisinden colour sensor > measure > colour seçilir. Bunun anlamı "switch" ifadesinde renk sensörünün renk karşılaştırılmasına dayanan bir veya daha fazla durumun olacağıdır.



Resim 76. Adım 2

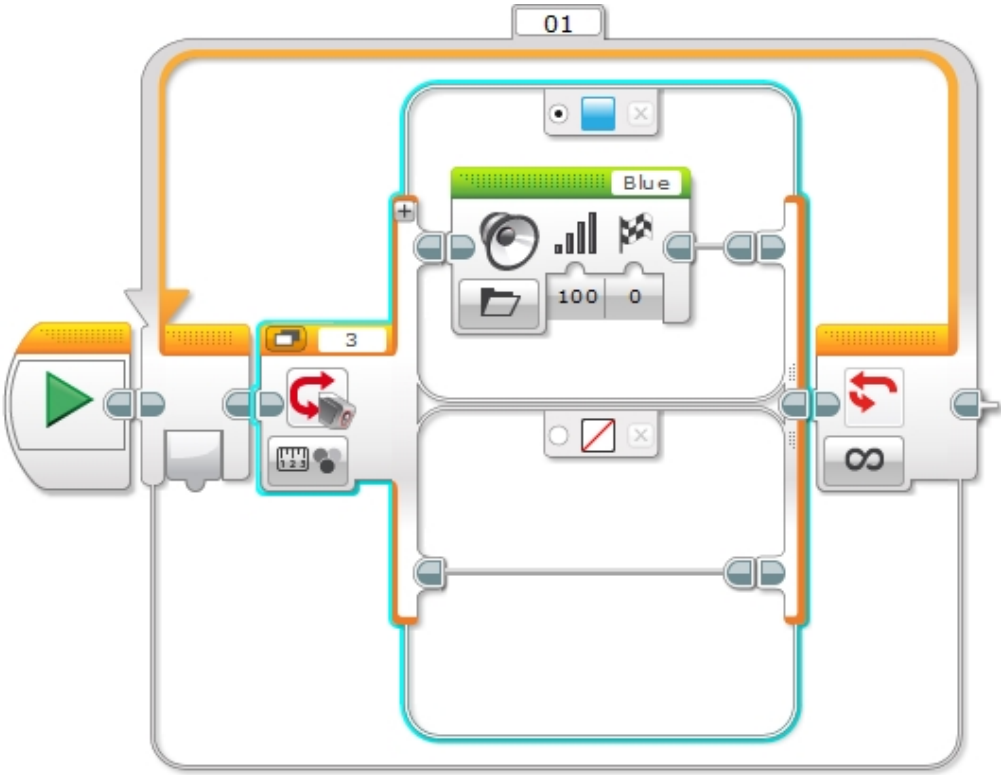
Switch renklerinde "blue" (mavi) seçilir, yani mavi koşulu oluşturulur.



Resim 77. Adım 3

Cismin mavi renkli olması durumunda "blue" diyen ses oynatılır. Mavi olmadığı durumda ise herhangi bir şey yapılmaz.

Bu kodun eksik olduğu nokta, bu işlemi sadece program çalışır çalışmaz bir kere yapacak olmasıdır. Fakat robotun mavi rengi her gördüğünde mavi demesi istenmektedir. Bu yüzden bu komutu bir döngü içerisine alıp sürekli tekrarlatmak gerekir. Bu programın son hâli aşağıdaki resimde gösterilmektedir.



Resim 78. Örnek Program

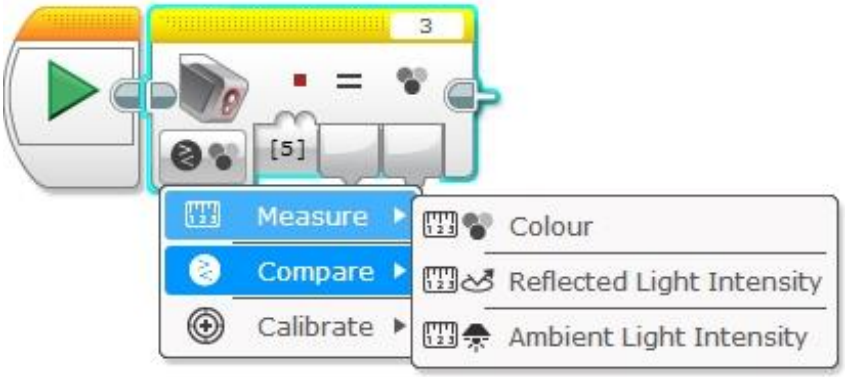
Bu aşamada, öğrencilere "anahtar (switch)" ile birden fazla seçimin yapılabileceği gösterilerek ve uygulayarak anlatılır (**Hatırlatma:** Yeni koşullar eklemek için + işaretine (add case) tıklanır). Bundan sonra öğrencilerden tanımlı yedi rengin tamamının İngilizce adını söyleyen ve herhangi bir renk olmadığında "No" (Lego Sound Files> Communication klasörü içerisinde) diyen programı oluşturmaları istenir. Rehber öğretmen öğrencilere programı yazarken yardımcı olabilir fakat tam çözümü vermemelidir. (**Rehber öğretmen için ipucu:** Programı yazmak için + işaretine (add case) tıklanarak her renk için ve renk olmama durumu için birer koşul yaratılır ve gerekli ses oynatma komutları bu koşulların içerisine yazılır.) Öğrencilerin programı farklı renkler için çalıştırıp komutları denemesi sağlanmalıdır.

1.3. Gözle: Yansıyan Işık Şiddeti

Renk sensörü ışık şiddetini de ölçebilir. Işık şiddetini ölçmek için iki mod bulunur. Bunlar yansıyan ışık şiddeti ve ortam ışığı şiddetidir. Yansıyan ışık şiddetini ölçmek için renk sensörü karşısındaki yüzeye kırmızı bir ışık gönderir ve o yüzeyden yansıyan ışığın şiddetini "0" ile "100" arasında bir değer olarak belirler. "0" en karanlık, "100" ise en aydınlık değeri ifade eder. Ölçülen değer sıfıra ne kadar yakınsa yüzeyden yansıyan ışık o kadar azdır, yani karanlıktır. Ölçülen değer 100'e ne kadar yakınsa yüzeyden yansıyan ışık o kadar fazladır, yani aydınlıktır.

EV3 programında, Sensor sekmesindeki "renk sensörü (colour sensor)" bloğu incelenerek renk sensörünün program içerisinde nasıl kullanılacağına bakılır. Renk sensöründe "Ölçüm (Measure)" ve "Karşılaştır (Compare)" olmak üzere iki seçenek bulunur. Ölçüm seçeneği, o sensörden ölçülen değeri hesaplar ve akıllı tuğlaya iletir. Örneğin, karşısında bir cisim bulunan renk sensörü, ölçüm seçeneği ile o cismin rengini (colour), o cisimden yansıyan ışığın yoğunluğunu (Reflected Light Intensity) ya da ortamda bulunan ışığın şiddetini (Ambient Light

Intensity) hesaplayıp akıllı tuğlaya iletir. Karşılaştır (Compare) seçeneğinde ise ölçülen değer önceden belirlenmiş bir değer ile karşılaştırılır. Aşağıda resimde bu seçenekler görülmektedir.



Resim 79. Renk Sensörü Bloğu Seçenekleri

Bir cisimden yansıyan ışığın kırmızı olup olmadığının saptanması gerektiğini varsayalım. Bu durumda karşılaştırma seçeneği içerisinden "renk (colour)" kullanılarak cismin renginin kırmızı olup olmadığı bulunur. Aşağıdaki resimde görüldüğü gibi, karşılaştırma seçeneğinde karşılaştırılacak değer 5, yani kırmızı olarak belirlenir. Eğer karşıdaki cisim kırmızı ise doğru değeri, kırmızı değilse yanlış değeri üretilir.

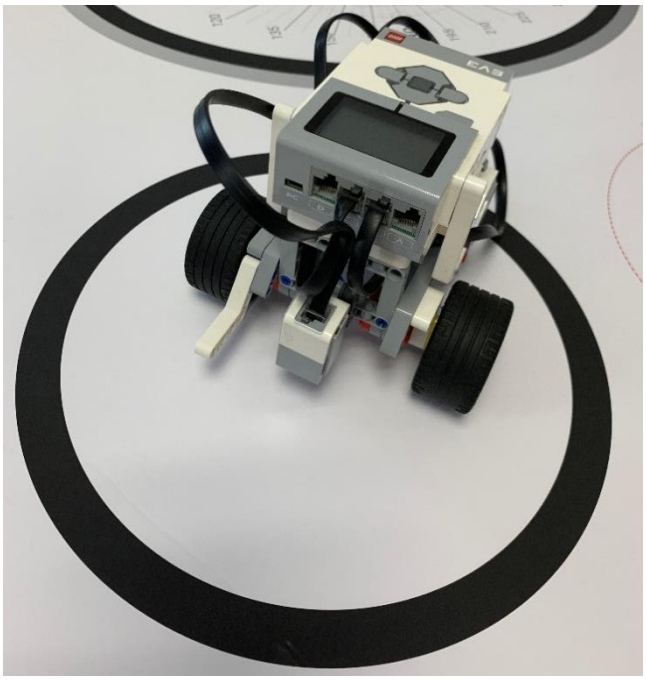
Resim 80. Adım 1

Şimdi de bir cisimden yansıyan ışığın miktarının 80'den fazla olup olmadığının saptanması gerektiğini varsayalım. Karşılaştırma seçeneği içerisinden "yansıyan ışık yoğunluğu (reflected light intensity)" kullanılarak bu işlem gerçekleştirilebilir. Aşağıdaki resimde verilen komut bu işlem için kullanılır. Görüldüğü üzere büyüktür (>), büyük eşittir (≥), küçüktür (<), küçük eşittir (≤), eşittir (=), eşit değildir (≠) gibi karşılaştırma işlemleri gerçekleştirilebilir. Karşılaştırılmak istenen değer belirlenip yazılır. Örnekte bu değerin "80" olarak belirlendiği görülebilir. Sonuçta yansıyan ışık 80'den büyükse, yani yeteri kadar aydınlıksa doğru değeri, değilse yanlış değeri üretilir.

Resim 81. Adım 2

1.4. Uygula: Dairenin İçerisinden Çıkmayan Robot

Bu etkinlikte, beyaz bir zemin üzerinde, siyah şeritle çizilmiş dairesel bir bölgede hareket edip bu bölgeden çıkmayan robot yapılır.



Resim 82. Kullanılacak Mat Bölgesi

Bu programın algoritması şöyledir:

i. Siyah şeridi görünceye kadar ilerle.
ii. Siyah şeridi görünce robotun sol tekerini 100 derece geriye çevir.
iii. i ve ii numaralı adımları sürekli tekrarla.

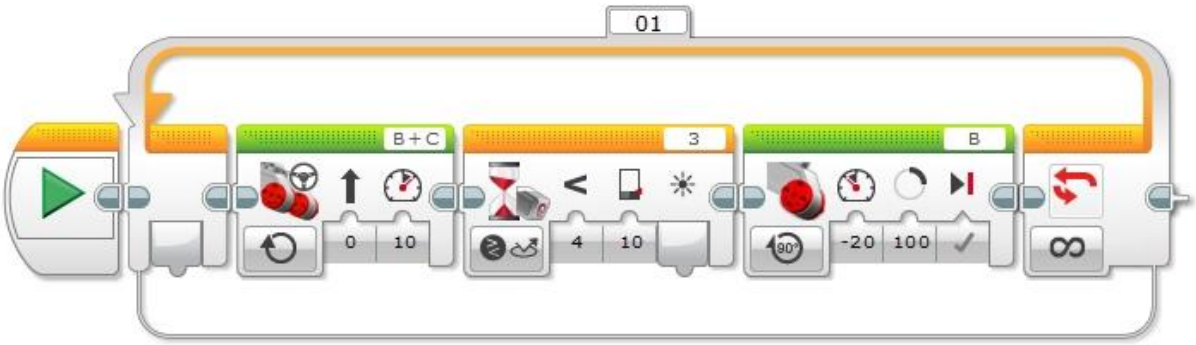
Bu program "anahtar (switch)" bloğu ve "bekle (wait)" bloğu kullanılarak iki farklı şekilde yapılır. Fakat öncelikle siyah şeride gelindiğini algılamak için kullanılması gereken eşik değerinin nasıl bulunacağından söz edilmelidir. Farklı yüzeylerde ve farklı renk tonlarında yansıyan ışık miktarı değişiklik gösterebilir. Bu yüzden her bir grup bu işlemi ayrı ayrı yapıp eşik değerini kendilerine göre belirlemelidir.

Eşik değerinin bulunması için aşağıdaki adımlar uygulanmalıdır:

i. Renk sensörü beyaz yüzey üzerine getirilir.
ii. Akıllı tuğla üzerinden veya robota bağlı bilgisayar üzerinden yansıyan ışık miktarı ölçülür.
iii. Renk sensörü siyah yüzey üzerine getirilir.
iv. Akıllı tuğla üzerinden veya robota bağlı bilgisayar üzerinden yansıyan ışık miktarı ölçülür.
v. Eşik değeri bu iki değer arasından siyah yüzey üzerindeki değere yakın olarak seçilir.

Örneğin, beyaz yüzey için "20", siyah yüzey için "7" değeri bulunmuş olsun. Bu durumda eşik değeri "10" olarak belirlenebilir. Eşik değeri daha önce de açıklandığı gibi farklı ortamlarda farklı değerler olarak bulunabilir/belirlenebilir. Bu değer kullanılarak sensörün siyah şeridin üzerine geldiği bilgisi robota iletilir. Yansıyan ışık miktarı 10'dan küçük olduğunda, robot siyah şeridin üzerinde olduğunu algılar.

Programı oluşturmak için öncelikle "bekle (wait)" bloğu kullanılır. Sol teker motorunun "B" portuna, sağ teker motorunun ise "C" portuna takıldığı varsayılsın. Yapılması gereken işlem, renk sensöründen yansıyan değer, eşik değeri olarak belirlenen 10'un altına düşünceye kadar (yani beyaz yüzeyden siyah yüzeye geçme işlemi kesinleşinceye kadar) robotu sürekli "10" hızında ilerletmektir. Eşik değerine geldikten sonra ise sol teker geriye doğru "20" hızında "100" derece döner (**Dikkat:** Burada robot "100" derece dönmez, yalnızca bir tekerlek "100" derecelik bir tur atar). İşlem bu hâliyle sadece bir kere yapılır. Fakat bu işlemin sürekli tekrarlanması gerekir. Bu yüzden de işlemlerin bir döngü içerisine alınması gerekmektedir. Sonuç olarak aşağıdaki resimde verilen örnekteki gibi bir program elde edilir.

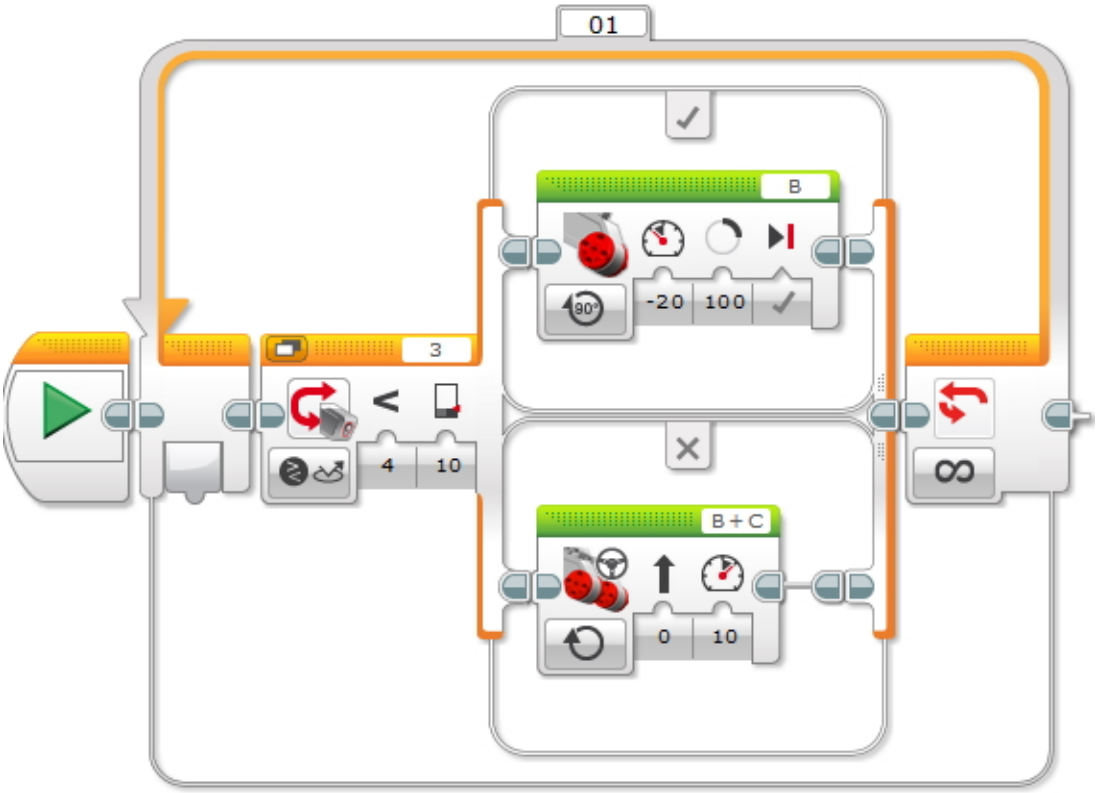


Resim 83. Örnek Program

Aynı görev, "bekle (wait)" bloğu kullanmadan "anahtar (switch)" bloğu ile yapılacak olursa mantığı şu şekilde kurgulanır:

i. Ortamdan yansıyan ışığın değeri, eşik değerinin altına düşerse sol teker ile geriye git.
ii. Ortamdan yansıyan ışığın değeri eşik değerinin üstündeyse ileriye git.
iii. i ve ii numaralı işlemleri sürekli tekrarla.

Bu program adım adım oluşturularak öğrencilere detaylıca anlatılır ve son hâli aşağıdaki resimde olduğu gibi görünür.



Resim 84. Örnek Program

1.5. Gözle: Ortam Işığı

Ortam ışığı yoğunluğu modunda renk sensörü herhangi bir ışık göndermez. Doğrudan bulunduğu ortamdan gelen ışığın yoğunluğu ile ilgilenir. Bu yoğunluk değeri yine “0” ile “100” arasındadır. Sıfır en karanlık, “100” ise en aydınlık değeri ifade eder. Ortamda bulunan ışığın şiddeti (ambient light intensity) hesaplanıp akıllı tuğlaya iletilebilir.

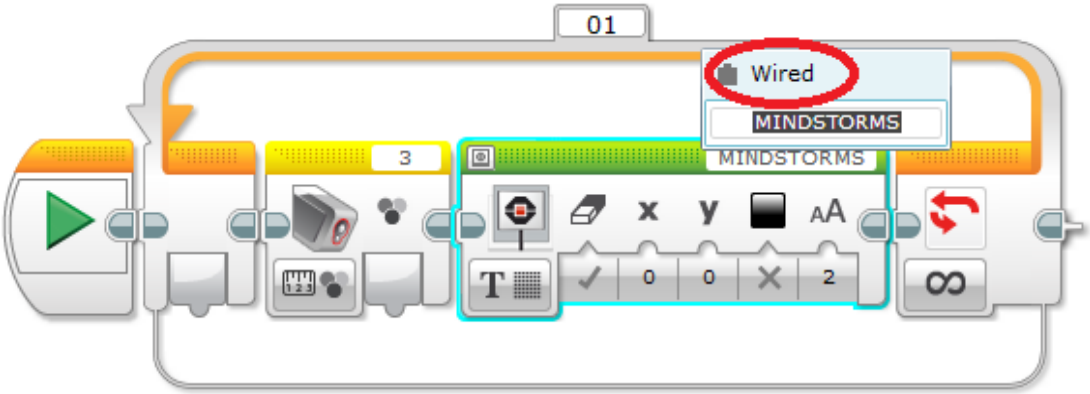
1.6. Uygula: Ortam Işığı

Bu etkinlikte renk sensörünün ölçtüğü ortam ışığı değerini her iki saniyede bir ekrana yazdıran program oluşturulur. Ortam ışığı her iki saniyede bir ölçülmek istendiği için öncelikle renk sensörü bloğu bir döngünün içine alınır. Sensörün ölçtüğü değeri ekrana yazdırmak için “ekran (display)” bloğu renk sensörüne bağlanıp aşağıdaki resimde olduğu gibi “metin (text)” seçeneği aktif hâle getirilmelidir.

Resim 85. Adım 1

Renk sensörünün ölçtüğü değeri tuğlanın ekranında yazdırmak için öncelikle display bloğundaki içerik seçeneği aşağıdaki resimde olduğu gibi “Kablolu (Wired”) olarak belirlenir.

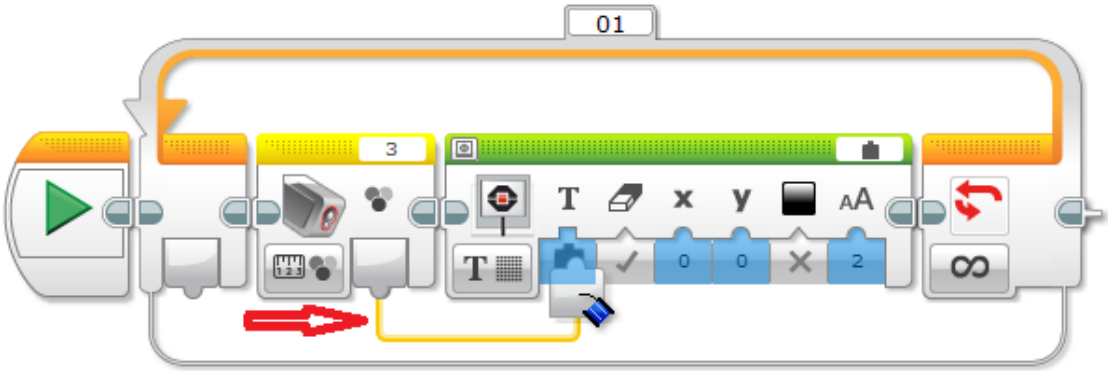
.



Resim 86. Adım 2

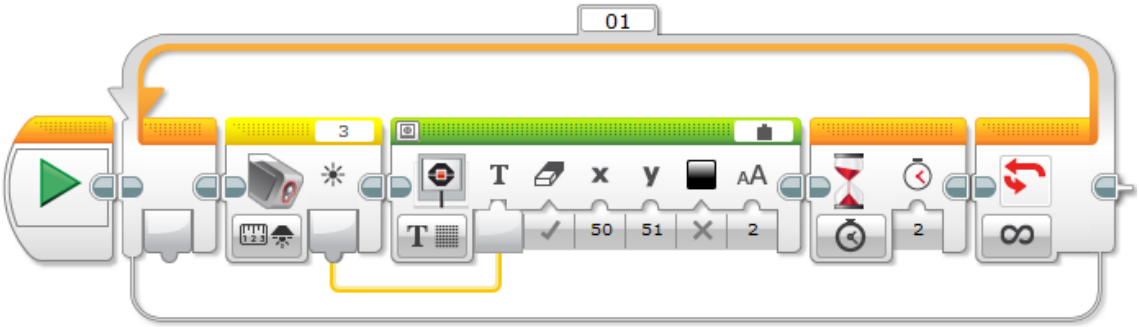
Sonra da renk sensörü bloğunun ölçtüğü değeri tuğla ekranında göstermek için renk sensörü bloğu aşağıdaki resimde olduğu gibi “ekran (display)” bloğu ile bağlanır.

.



Resim 87. Adım 3

Belirlenen değerin iki saniyede bir ölçülmesi için ise aşağıdaki resimde olduğu gibi "ekran (display)" bloğunun sonuna "bekle (wait)" bloğu eklenip bekleme süresi 2 yapılır.

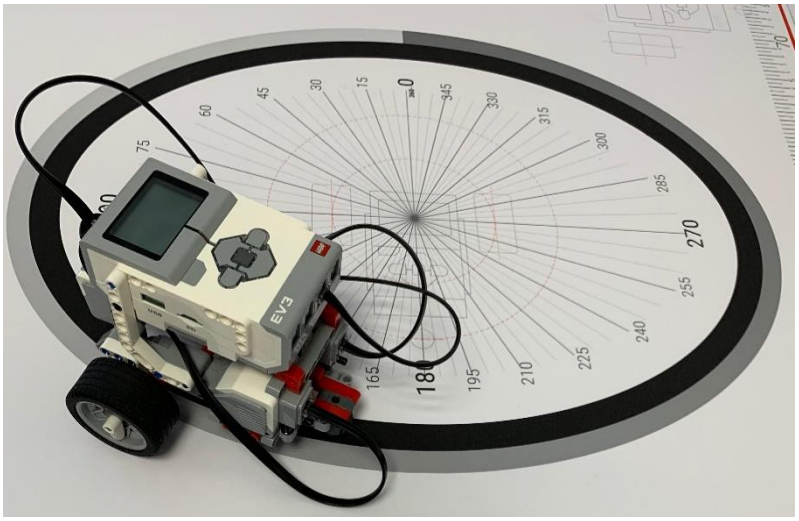


Resim 88. Adım 4

2. TASARLA VE ÜRET

2.1. Tasarla: Eliptik Bir Yörüngeyi Takip Eden Robot

Bu etkinlikte, beyaz bir zemin üzerinde, siyah bir şeritle elips şeklinde çizilmiş bir yörüngeyi sürekli takip edecek bir robotun programı hazırlanır. Program için sensörün elipsin iç tarafında olduğu ve robotun saat yönünün tersine doğru ilerlediği varsayılır. (**Dikkat:** Aksi hâlde programın çalışmayacağı vurgulanmalıdır). Tasarlama sürecinde öğrencilerin eliptik bir yörüngeyi takip eden robot için aşağıda örnek olarak verilen iki adıma benzer bir süreci gerçekleştirmesi gerekir. Bu etkinlikte mat üzerinde ilgili bölüm kullanılır.



Resim 89. Kullanılacak Mat Bölgesi

Öğrencilerden öncelikle beyaz bir zemin üzerinde, siyah bir şeritle elips şeklinde çizilmiş bir yörüngeyi sürekli takip edecek bir robotun program kodunun nasıl oluşturulacağı üzerine düşünmeleri istenir. Öğrenciler grup olarak tartışır. Rehber öğretmen onlara gerektiği noktada yardımcı olabilir fakat tam bir çözüm vermemelidir, gruplar çözümlerini kendileri üretmelidir.

Tanımlama: Öğrencilerin öncelikle istenilen işlemin neler gerektirdiğini belirlemesi ve maddeler hâlinde yazması gerekir. Örneğin:

i. Robot sürekli olarak hareket eder.
ii. Robot siyah bölgeye geldiğinde beyaz bölgeye yönlendirilir.
iii. Robot beyaz bölgeye geldiğinde siyah bölgeye yönlendirilir.
iv. Robot durduruluncaya kadar bu işleme devam eder.
v.

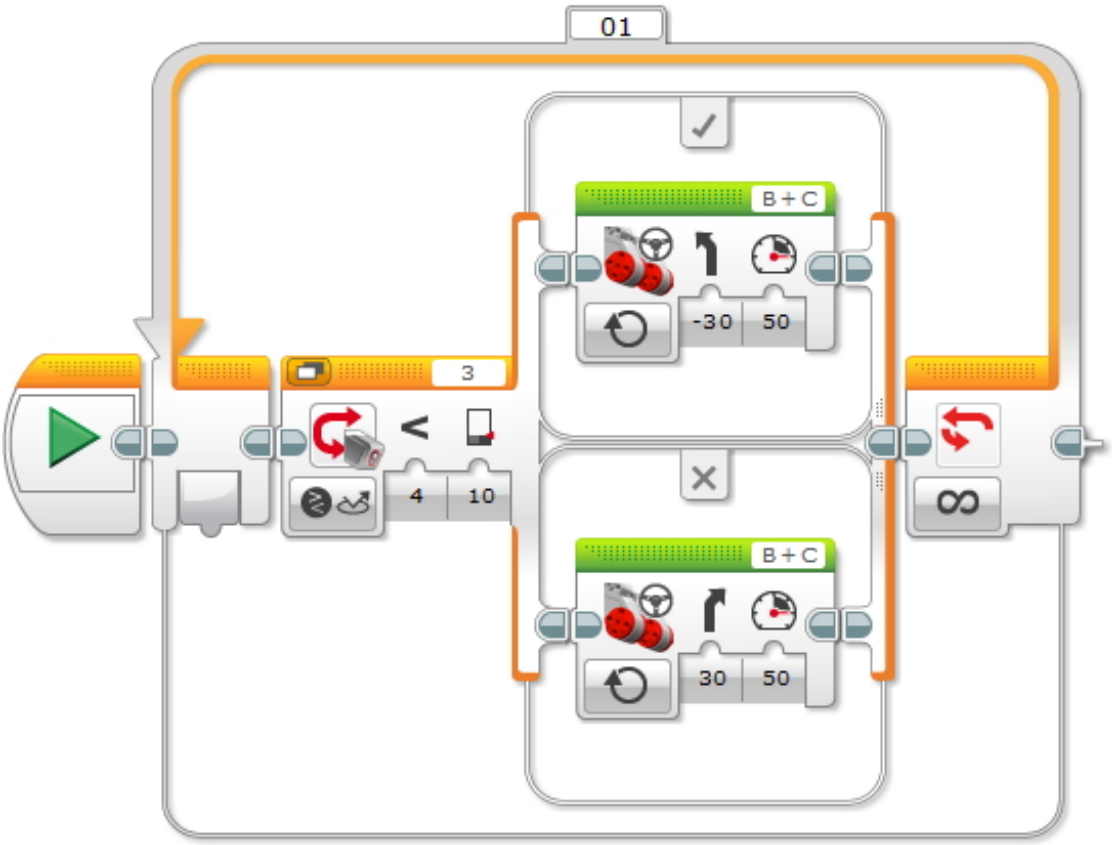
Fikir üretme: Bu aşamada öğrencilerin yukarda belirlenen işlemlerin nasıl yapılabileceği ile ilgili fikir yürütmesi gerekir. Örneğin, öğrenciler aşağıdaki maddelere benzer fikirler üretebilir.

i. Robotun siyah çizgiye geldiğinin anlaşılması için eşik değeri belirlenir (beyaz bölgeden siyah bölgeye geçtiğini anlamak için).
ii. Robot ileri doğru belirli bir hızla hareket eder.
iii. Robotun beyaz bölgeden çıkma durumu sürekli olarak kontrol edilir (renk sensöründen gelen verinin eşik değerine yakınlaşma durumu sürekli kontrol edilir).
iv. Robot siyah bölgeye geldiğinde dairenin içerisine doğru (dönüp geri gidecek şekilde) yönlendirilir.
v. ii, iii ve iv numaralı adımlar robot durduruluncaya kadar tekrarlanır.

Bu programın temel mantığı robot siyah bölgeye geldiğinde onu beyaz bölgeye yönlendirmek, beyaz bölgeye geldiğindeyse onu siyah bölgeye yönlendirmektir. Böylece robot zikzaklar çizerek ilerler. Bir önceki etkinlikte olduğu gibi, robotun siyah bölgede olduğunu anlaması için bir eşik değeri belirlenmelidir. Bunun için her bir grubun kendi programlarında kendi buldukları eşik değerini kullanması gerekir. Grupların eşik değeri bulma sorununa çözümü farklı olabilir, önemli olan bu fikirlerin gruplar tarafından ortaya konulması ve ortaya konulan fikirlerin problemin çözümünü sağlayabilecek olmasıdır.

2.1. Üret: Eliptik Bir Yörüngeyi Takip Eden Robot

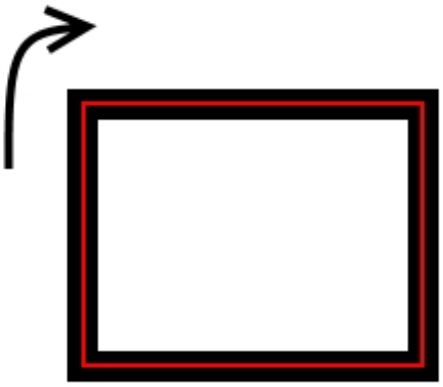
Öğrenciler çözüm tasarımlarını yaptıktan sonra bilgisayar ve robot başında çalışarak istenilen görevi yerine getirir. Öğrencilerin aşağıdaki resimde görülen programa benzer bir program hazırlamaları gerekir.



Resim 90. Örnek Program

2.2. Tasarla: Dikdörtgen Üzerinde Hareket Eden Robot

Bu etkinlikteki amaç, robotun dikdörtgen şeklindeki bir çizgiyi takip etmesi için program oluşturmaktır. Mat üzerinde olduğu gibi, burada da dikdörtgenin kenarları siyah bir şeritle çizilmeli ve arka plan rengi beyaz olmalıdır. Program için robotun dikdörtgenin kenarlarının dış kısmını (iç kısmından takip eden de yapılabilir fakat burada dış kısmından takip etmelidir) takip ederek saat yönünde ilerlediği varsayılır. Aşağıda resimde dikdörtgen örneği görülmektedir.

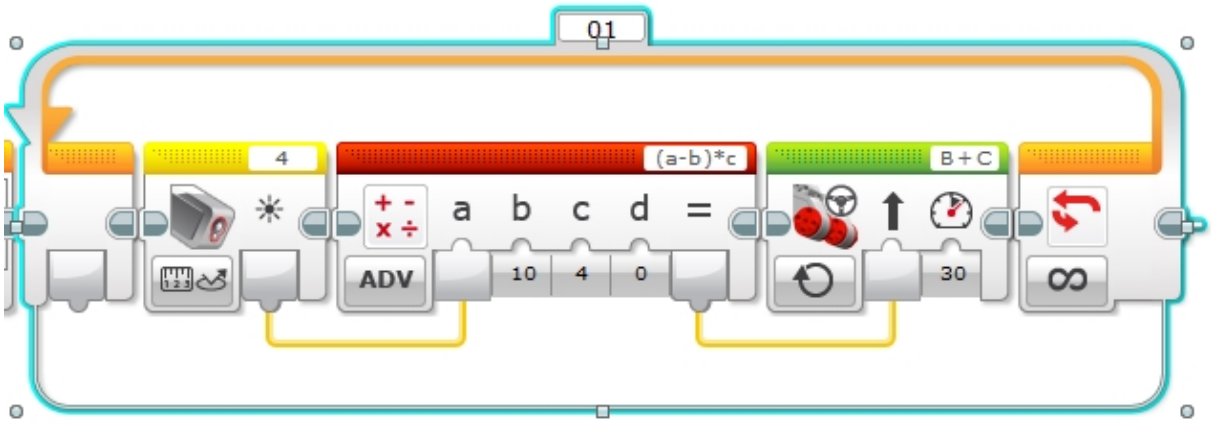


Resim 91. Dikdörtgen Takip Bölgesi

Robotun dikdörtgenin üzerine gösterim amaçlı çizilen kırmızı hattı takip etmesi beklenir. (**Dikkat:** Gerçekte dikdörtgenin üzerinde kırmızı hat bulunmamalıdır.) Robotun gösterim amaçlı çizilen kırmızı hattı bire bir takip etmesi zorunlu olmasa da bu çizgiye yakın bir şekilde hareket etmesi gerekir. Programı yazmaya başlamadan önce grupların tasarlama adımı için yukarıda bir örneği verilen tanımlama ve fikir üretme sürecini gerçekleştirmesi gerekir.

2.2. Üret: Dikdörtgen Üzerinde Hareket Eden Robot

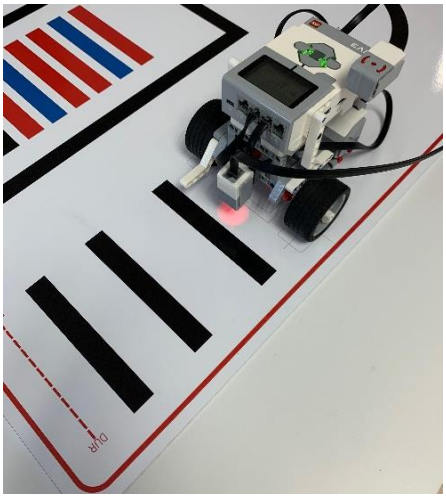
Öğrenciler çözüm tasarımlarını yaptıktan sonra bilgisayar ve robot başında çalışarak istenilen görevi yerine getirmelidir. Aşağıdaki resimde görülen çözüme benzer bir örnek hazırlanabilir. Bu çözümde yukarıdaki dikdörtgendeki kırmızı bölgenin yansıyan ışık miktarı değeri (b) "10" olarak alınmıştır. Fakat bu değer öğrencilerde farklılık gösterebilir. Öğrenciler robotu kullanarak bu değeri bulup kendi durumlarına göre değiştirmelidir. Verilen programdaki "c" değeri robotun kırmızı çizgiden uzaklaştığında yeniden kırmızı çizgiye doğru reaksiyon hızı katsayısıdır. Bu değer ne kadar fazla olursa robot sapmaya o kadar hızlı reaksiyon verir. Fakat bu değer de programcı tarafından kendi durumuna göre ayarlanmalıdır. "C" değerinin fazla veya az olması robotun çizgiyi takip etmemesine sebep olabilir. Son olarak robot dikdörtgenin dış yüzünde olmalıdır. Eğer iç yüzüne konulursa program çalışmaz.



Resim 92. Örnek Program

2.3. Tasarla: Üçüncü Çizgiyi Geçince Duran Robot

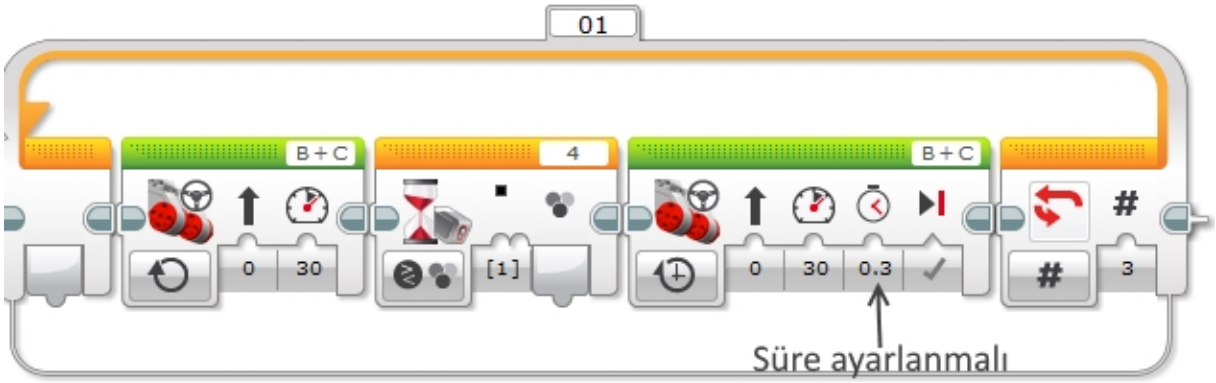
Bu etkinlikteki amaç, robotun aşağıdaki resimde gösterildiği gibi üçüncü çizgiyi kısa bir miktar geçince durması için program oluşturmaktır. Robot ilk iki çizgiyi tamamen geçtikten sonra üçüncü çizgiyi biraz geçip durmalıdır. Programı yazmaya başlamadan önce grupların tasarlama adımı için yukarıda bir örneği verilen tanımlama ve fikir üretme sürecini gerçekleştirmesi gerekir.



Resim 93. Kullanılacak Mat Bölgesi

2.3. Üret: Üçüncü Çizgiyi Geçince Duran Robot

Öğrenciler çözüm tasarımlarını yaptıktan sonra bilgisayar ve robot başında çalışarak istenilen görevi yerine getirir. Aşağıdaki resimde görülen örnek çözüme benzer bir program hazırlanabilir. Döngünün içerisindeki ikinci move steering bloğunun süresi çizgilerin kalınlığına göre ayarlanmalıdır. Örnekte verilen değer elektrik yalıtım bandı kalınlığı için geçerlidir.



Resim 94. Örnek Program

3. DEĞERLENDİR

Bu aşamada hedef, öğrencilerin öğrenme sürecinde yaşadıkları ve öğrendikleri üzerine düşünmesini sağlamaktır. Bu sayede öğrencilerin, problem çözme yetenekleri gelişecek, öğrenciler dersin konusu ve kendileri ile ilgili gözlemler yaparak öğrendikleri yeni konuları ve kendilerini değerlendirmekle beraber sonraki çalışmalarını planlamak için de fırsat bulurlar. Öğrencilerden şu soruları yanıtlamaları istenebilir:

- Verilen problemleri nasıl tanımlarsınız? (problemi kendi cümleleri ile ifade etme)
- En çok hangi görevde zorlandınız? Bu zorlukların üstesinden nasıl geldiniz? (Problemin çözümü için hangi stratejileri kullandınız ve neden bu stratejileri seçtiniz?) Yeteri kadar

tartışma ortamı oluşmazsa, rehber öğretmen aşağıdaki soruları kullanarak tartışma ortamı yaratmaya çalışır.

- Eliptik yörüngeyi takip eden robotta, robotun çizgide kalmasını nasıl sağladınız?
- Dikdörtgen üzerinde hareket eden robotta, robotun köşeyi dönmesini nasıl sağladınız?
- Üç çizgi etkinliğinde, robotun siyah alanları geçmesini nasıl sağladınız?

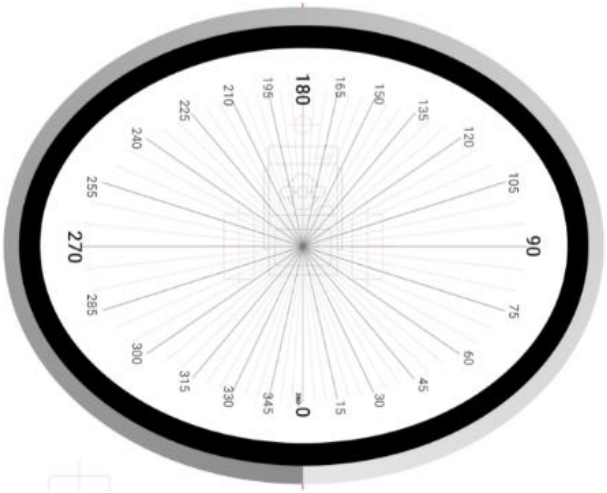
- Kullandığınız yöntemler bu sıkıntıları gidermekte başarılı oldu mu?
- Grup arkadaşınızla anlaşmazlığa düştüğünüz durumlar oldu mu ve bunların üstesinden gelmek için neler yaptınız?
- Grup arkadaşınızdan ne/neler öğrendiniz?

Değerlendirme, öğrencileri sıkmadan, her bir soru için verilen cevaplar tatmin edici bir düzeye ulaşıncaya kadar devam ettirilebilir.

4. İLAVE ETKİNLİK

4.1. Tasarla ve Üret: Eliptik Bir Yörüngeyi Takip Eden Robot

Bu etkinlikte, aşağıda gösterilen (mat üzerindeki) elipsin dış yüzeyini takip edecek bir robotun programı hazırlanır. Bu elipsin dış yüzeyinde değişen yoğunlukta gri şerit de bulunur. Öğrenciler kullanacakları değerleri belirlerken bu durumu göz önünde bulundurmalıdır. Program için sensörün elipsin üzerinde olduğu ve robotun saat yönünün tersine doğru ilerlediği varsayılır. Programı yazmaya başlamadan önce grupların tasarlama adımı için yukarıda bir örneği verilen tanımlama ve fikir üretme sürecini gerçekleştirmesi gerekir.



Resim 95. Kullanılacak Mat Bölgesi

6. Hafta: Robot ve Algoritma

Ön Bilgi:

- Öğrenciler robot kavramını bilir.
- Öğrenciler robot setiyle farklı robotik tasarımlar yapmıştır.
- Öğrenciler robot setini programlamak için grafik arayüzünü kullanmıştır.
- Öğrenciler robotun hareket etmesi için gerekli programlama adımlarını oluşturmuştur.
- Öğrenciler tuğlanın çeşitli sesleri çıkarması ve tuğla ekran görüntüsünün düzenlenmesi için gerekli programlama adımlarını oluşturmuştur.
- Öğrenciler dokunma, mesafe, açı ve renk sensörlerini farklı amaçlar için programlama adımlarını oluşturmuştur.

Haftanın Kazanımları:

- Öğrenciler değişken tanımını yapar.
- Öğrenciler değişken türlerini açıklar.
- Öğrenciler değişkenlerle sensörleri bir arada kullanır.
- Öğrenciler değişkenleri kullanarak program akışını değiştirir.
- Öğrenciler değişkenleri problem çözümünde kullanır.

Haftanın Amacı:

Bu haftanın amacı, öğrencilerin değişken kavramını öğrenerek değişken tanımı yapmasıdır. Sonraki süreçte, öğrencilerin çeşitli problemlerin çözümünde değişkenleri tanımlamak ve kullanmak için gerekli düzenlemeleri yapabilmesini sağlamaktır.

Kullanılacak Malzemeler:

Robot seti, bilgisayar, mesafe sensörü, renk sensörü, mat (çalışma alanı).

Haftanın İşlenişi:

Gözle: Değişkenlerin tanımlanması ve kullanılmasını çeşitli örnekler üzerinde inceleme.

Uygula: Değişkenlerin tanımlanmasını ve kullanılmasını çeşitli örnekler üzerinde uygulama.

Tasarla: Robotun karşısına çıkan engelin sağından geçme, çizgi takip ederken önüne çıkan engeli aşıp tekrar çizgiyi takip etme, renkleri sayma ve yerde renklerle oluşturulan şifreli sayıyı bulma işlemlerini değişkenler kullanarak gerçekleştirebilmesi için gerekli tasarımları yapma.

Üret: Verilen görevleri programlama.

Değerlendir: Haftanın içeriği ile ilgili yansıtma etkinliği.

1. GÖZLE VE UYGULA

1.1. Gözle: Değişkenler

Programcılar yazdıkları programlar çalıştırıldığında genellikle bilgisayardan bazı değerleri hatırlamasını isterler. Bilgisayar bu değerleri "RAM" olarak adlandırılan bellekte saklar. Bilgisayarın "RAM" belleğinde sakladığı bu değerlere programcılar değişken adını verir. Değişkenler aracılığı ile bilgisayarın belleğinde çeşitli değerler saklanabilir, değişkenin içerisinde saklanan değerler daha sonra istenildiğinde çeşitli işlemler için kullanılabilir ve istenilirse değiştirilebilir. Öğrencilerin bir sınavdan aldığı puanlar, boy uzunlukları ve bir sınıftaki öğrenci sayısı değişkenlere örnektir. Programcılar değişkenleri çeşitli isimlerle adlandırırlar. Örneğin, 5A sınıfındaki öğrencilerin sayısının bir değişken olarak saklanması istenirse bu değişkene "5aOgrenciSayisi" adı verilebilir. Değişken ismi olarak "5aÖğrenciSayısı" yerine "5aOgrenciSayisi" yazılmasının nedeni, "*LEGO robotlarını programlamak için kullanılan EV3 yazılımının Ö, Ş, Ğ, Ü, İ, Ç ve ı gibi Türkçe karakterlere*" izin vermemesidir. Ayrıca, *değişken isimlerinde kullanılması yasak olan karakterler (örneğin "!" karakteri) girildiğinde EV3 yazılımı uyarı verir*. Programcı değişken isimlerini bu kurallara göre dilediği gibi belirleyebilir. Fakat değişken isimleri uzun olduğunda onları okumak zorlaşabilir. Bu yüzden değişken isimlerinin kısa olması programcının işini kolaylaştırır. Örneğin, "5A" sınıfındaki öğrenci sayısının tutulacağı değişkenin ismi değişkenin işlevini anımsatacak şekilde kısaca "5aOgrSay" olarak da belirlenebilir.

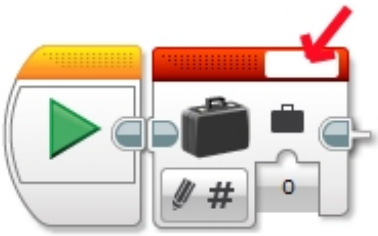
1.2. Uygula: Grup Sorusu

Bu sorular her bir gruba sorulur ve yanıtlar sınıfta tartışılır:

i. Bilgisayar oyunlarında her oyuncunun bir puanı vardır. Bilgisayar oyunlarında oyuncunun puanı değişkene örnek olabilir mi? Yanıtınızı sebepleriyle açıklayınız.
ii. Günlük yaşamınızdan iki farklı değişken örneği bulunuz. Bu değişkenlere istediğiniz bir isim veriniz. Fakat verdiğiniz ismin kısa olmasına dikkat ediniz.

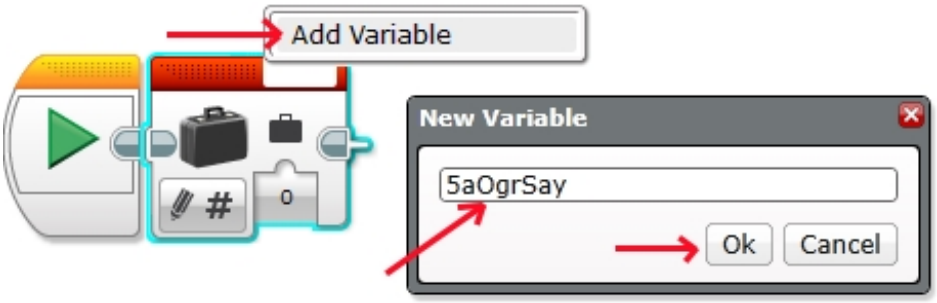
1.3. Gözle: EV3 Yazılımında Değişken Tanımlama

Değişken tanımlamak için Veri İşlemleri (Data Operations) sekmesindeki "Değişken (Variable)" bloğu kullanılır. Değişkeni oluşturup ona bir isim vermek için "Değişken (Variable)" bloğunun sağ üst köşesindeki boşluğa fare imleci ile tıklanır.



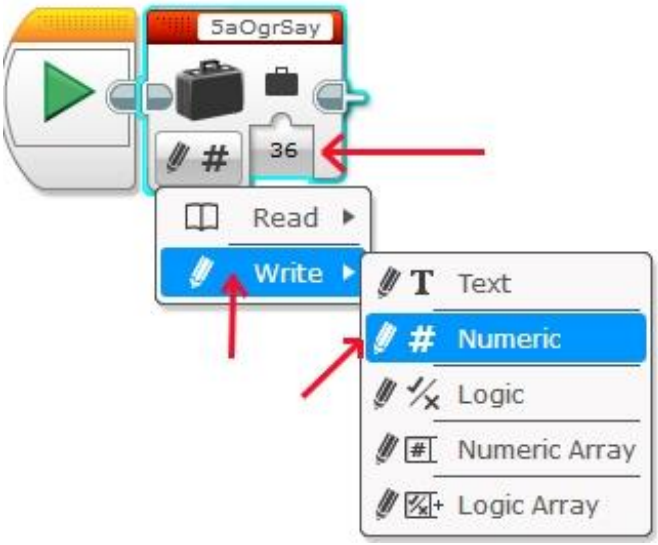
Resim 96. Adım 1

Aşağıdaki resimde de görüldüğü gibi ortaya çıkan menüden "Değişken Ekle (Add Variable)" seçilir ve açılan metin kutusunun içerisine değişkenin ismi yazılır. Son olarak "Tamam (Ok)" butonuna tıklanır.



Resim 97. Adım 2

EV3 yazılımında "metinsel" (text), "mantıksal" (logic) ve "sayısal" (numeric) değerler değişkenler içerisinde saklanabilir. Metinsel değişkenlerde saklanacak değer bir metindir. Örneğin, "Merhaba Dunya" ifadesi bir metinsel değişkendir. Mantıksal değişkenlerde ise "doğru" (true) veya "yanlış" (false) değerleri saklanabilir. Sayısal değişkenlerin içerisinde "5" ve "2.5" gibi sayısal değerler saklanır. Çeşidi ne olursa olsun bir değişkene öncelikle değer ataması yapılır. Bunun için "Değişken (Variable)" bloğunda "Yaz (Write)" modu seçilir ve çıkan menüden değişkenin tipi belirlenir. Aşağıdaki resimde görünen örnekte "5aOgrSay" değişkenine "36" değeri verildiği görülebilir.



Resim 98. Adım 3

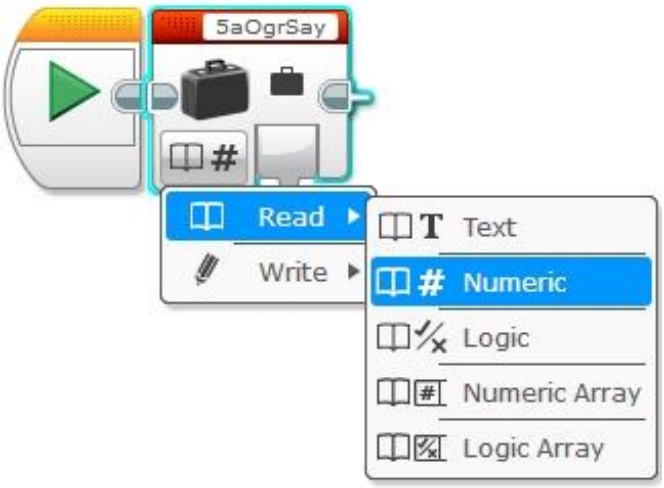
Artık bu değer programın içerisinde "5aOgrSay" ismi ile saklanır. İstenildiğinde farklı işlemler için bu değişken kullanılabilir veya bu değişkenin değeri değiştirilebilir. Değişkenin değerini değiştirmek için "Yaz (Write)" modunda değişkene yeni bir değer vermek yeterlidir (**Not**: Burada "36" değeri "45" ile değiştirilerek değişken için nasıl yeni bir değer atanabileceği öğrencilere gösterilir).

1.4. Uygula: EV3 Yazılımında Değişken Kullanma

Sayısal değişkenler ile yapılabilecek işlemlere örnek olarak "5A" sınıfındaki öğrenci sayısı ile "5B" sınıfındaki öğrenci sayısı toplanmak istenmiş olsun. "5A" sınıfında "36", "5B" sınıfında

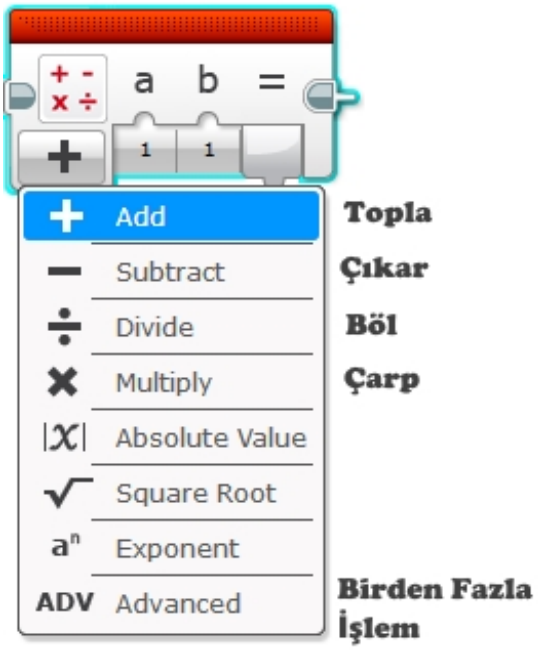
ise “28” öğrenci vardır. Öncelikle bu değişkenler oluşturulup bunlara sırasıyla “36” ve “28” değerleri atanmalıdır. Ardından bu değerler “Okuma (Read)” modunda toplama işlemine aktarılmalıdır. Aşağıdaki resimde okuma moduna nasıl gelineceği gösterilmektedir.

.



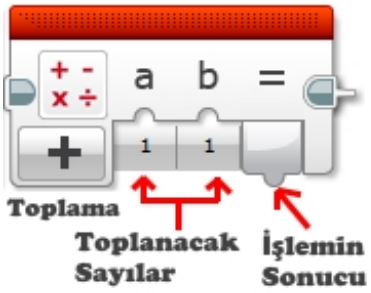
Resim 99. Okuma Modu

Toplama işlemi yapmak için resimde gösterildiği gibi Veri İşlemleri (Data Operations) sekmesindeki “matematik (math)” bloğu kullanılmalıdır. “Math” bloğu ile toplama, çıkarma, çarpma ve bölme gibi işlemler yapılabilir.



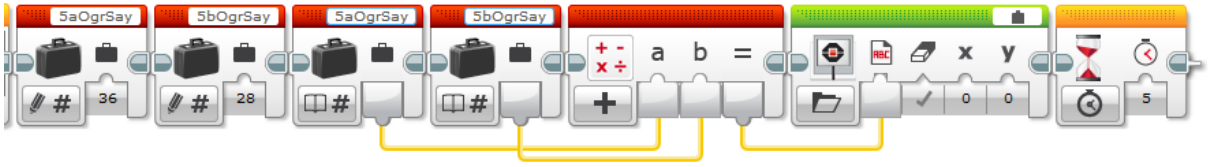
Resim 100. Matematiksel İşlemler

Öncelikle “math” bloğunda yapılacak işlem seçilir. Toplama işlemi yapılacağı için aşağıdaki resimde toplama işlemi seçildiği görülmektedir. Bu blokta işleme girecek sayılar için “a” ve “b” başlıklarında iki girdi kutucuğu ve sonuç için “=” başlığı altında çıktı kutucuğu bulunur. “a” ve “b” kutucuklarına girilen değerler toplanır ve sonuç (=) kutucuğuna aktarılır.



Resim 101. Toplama İşlemi

Aşağıdaki resimde “5A” ve “5B” sınıfındaki öğrenci sayılarını toplayıp robotun ekranına yazan program bulunmaktadır. Rehber öğretmen bu programı öğrencilere uygulamalı bir şekilde anlatmalıdır.



Resim 102. Örnek Program

1.5. Gözle ve Uygula: Karşısına Engel Çıktığında Sağından Geçen Robot

Bu etkinlikte karşısına engel çıktığında sağından geçen robot tasarlanır. Etkinlik için bir mesafe sensörü gerekir.

Rehber öğretmen tasarlama için tanımlama, fikir üretme ve uygulama adımlarını öğrencilerle birlikte gerçekleştirir.

Tanımlama: Öncelikle programın neler gerektirdiğinin belirlenmesi ve gerekli işlemlerin maddeler hâlinde ortaya konması gerekir. Örneğin:

i. Robot tasarımı araba şeklinde olmalıdır.
ii. Bu robot için bir adet mesafe sensörüne ihtiyaç vardır.
iii. Robot düz ve sabit bir şekilde ilerler.
iv. Robot önüne bir engel çıktığında (engel 35 cm’den yakında olduğunda) sağa döner.
v. Robot sağa döndükten sonra 30 cm ilerleyip sola döner.
vi. iii, iv ve v numaralı adımlar sırasıyla tekrarlanır.

Fikir Üretme: Bu aşamada yukarıda belirlenen işlemlerin nasıl yapılabileceği ile ilgili fikir yürütülür. Aşağıdaki maddelere benzer fikirler üretilebilir.

i. Robot sürekli bir hızda, düz doğrultuda ilerlemelidir.
ii. Bir değişken tanımlanmalı ve bu değişkenin değeri “0” olmalıdır.

iii. Eğer robot bir engele 35 cm'den fazla yakınlaşırsa değişkenin değeri 1 olmalıdır.
iv. Değişkenin değeri "1" ise (a) robot durmalı, (b) "90" derece sağa dönmeli, (c) tekerlerini bir tur döndürüp tekrar durmalı, (d) "90" derece sola dönmeli, (e) değişkenin değerini "0" yapmalı ve (f) sabit hızla düz doğrultuda devam etmelidir.
v. iii ve iv numaralı adımlar tekrarlanmalıdır.

Uygula: Öğrencilerden tanımladıkları ve hakkında fikir ürettikleri programı tamamlamaları ve kontrol edip eksikliklerini gidermeleri istenir.

Dikkat

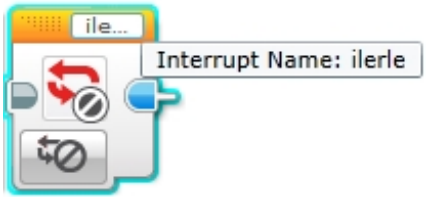
Rehber öğretmen kodun çalıştırılmasını öncelikle öğrencilere göstermelidir. Robotun çalışması için geniş bir alana ihtiyaç vardır. Bu etkinlikte dersliğin zemini kullanılabilir. Robotun karşısına çıkacak engel için herhangi bir nesne kullanılabilir veya rehber öğretmen robot ilerlerken ayağını robotun önüne engel olarak koyabilir. Robot ilerlerken birkaç kez robotun rotasına engel eklenebilir. Süreçte rehber öğretmen robotların ortamdaki herhangi bir nesneye çarpmamasına da dikkat etmelidir.

Bu programda amaç "robotun bir çizgiyi takip etmesi" değildir. *Bu programda amaç, zeminde düz bir hatta ilerleyen robotun karşısına 35 cm'den daha yakın bir engel çıktığında engelin sağından geçerek yeniden ileriye doğru hareket etmesini sağlamaktır*. Engeli aştıktan sonraki yön ile engelden önceki yön bire bir aynı olmak zorunda değildir. Robotun engel öncesi ve engel sonrası benzer yönlerde ilerlemesi yeterlidir.

Bu program paralel çalışan iki kod hâlinde tasarlanır. Paralel kodlardan bir tanesi robotun önüne engel çıkıp çıkmadığını kontrol eder. İkinci kod ise robotun önünde engel yoksa düz ilerlemesini engel olduğunda ise engeli aşmasını sağlar.

Bu programın oluşturulması için döngünün çalışmasının nasıl sonlandırılacağı bilinmelidir. Birçok programlama dilinde "break" komutu kullanılarak döngü sonlandırılır. EV3 yazılımında ise Akış Kontrolü (Flow Control) sekmesinde bulunan "döngü kesme (loop interrupt)" bloğu ile döngü sonlandırılır. Aşağıdaki resimde "döngü kesme (loop interrupt)" bloğunun ilerle adlı döngünün çalışmasını sonlandırmak için nasıl kullanıldığı görülmektedir.

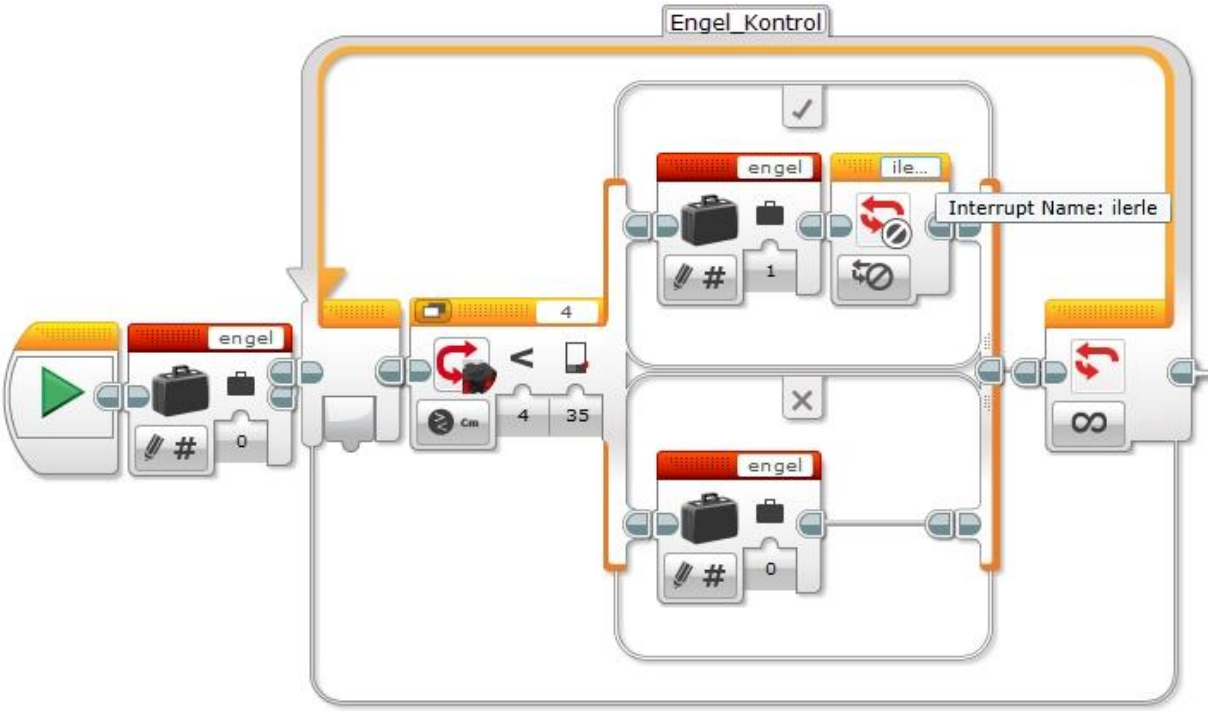
.



Resim 103. Interrupt Bloğu

Öncelikle paralel kodlardan ilki olan, engel olup olmadığını kontrol eden kod yazılabilir. Bu kod için engel isimli bir değişken oluşturulur. Robotun karşısında 35 cm'den yakın bir engel varsa engel değişkeninin değeri "1", engel yok ise engel değişkeninin değeri "0" olarak atanır. Engel ile karşılaşıldığında ilerle komutunun durdurulması gerekir. Bunun için engel değişkeninin 1'e eşit olması durumunda "döngü kesme (loop interrupt)" komutunun

çalıştırılması gerekir. Aşağıdaki resimde bu işlemler için düzenlenen kod blokları görülmektedir.

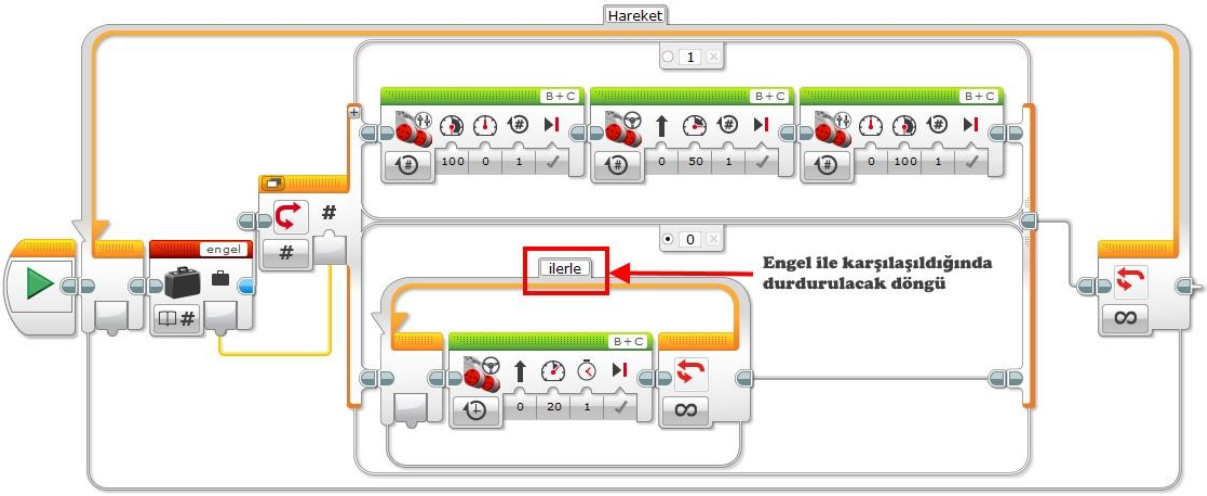


Resim 104. Birinci Kod

Paralel kodlardan ikincisinin görevi ise robotun aşağıdaki resimde gösterildiği şekilde sıralı işlemleri yapabilmesini sağlamaktır.

i. Eğer karşıda engel yoksa (engel=0) düz bir şekilde ilerlemek,
ii. Eğer karşıda engel varsa (engel=1) sağa dönüp bir teker turu ilerlemek ve
iii. Sola dönmek.

Aşağıdaki resimde bu işlemler için düzenlenen kod blokları görülmektedi.



Resim 105 İkinci Kod

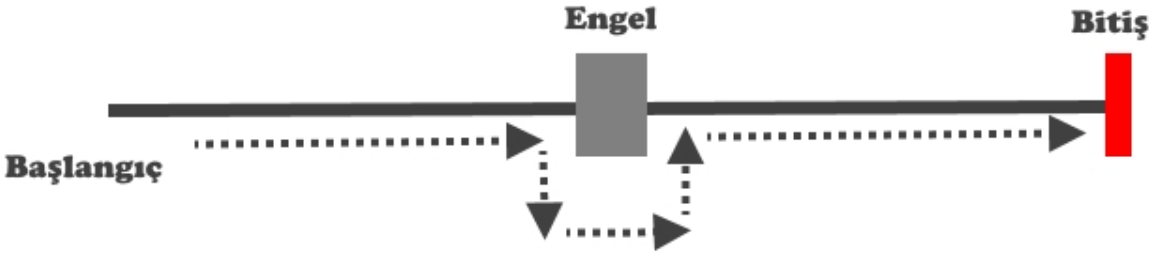
2. TASARLA VE ÜRET

2.1. Tasarla: Çizgi Takip Ederken Engel Aşan Robot

Rehber öğretmen çizgi takip ederken engel aşan robotun programının yazılmasını öğrencilerden ister. Bu programda robotun aşağıdaki sıralı işlemleri yapması sağlanır.

i. Herhangi bir engelle karşılaşmadığında çizgiyi takip etmek..
ii. Engel ile karşılaştığında çizgiden ayrılıp engelin sağından geçmek
iii. Engeli geçtikten sonra yeniden çizgi üzerine gelip çizgi üzerinden ilerlemeye devam etmek.
iv. Kırmızı renkteki bitiş çizgisine geldiğinde durmak.

Robotun yapması gerekli görevin şeması aşağıdaki resimde görülmektedir. Görev için birden fazla engel kullanılabilir.



Resim 106. Görev Şeması

Programı oluşturmaya başlamadan önce grupların tasarlama adımı için yukarıda bir örneği verilen tanımlama ve fikir üretme süreçlerini gerçekleştirmesi gerekir.

2.2. Üret: Çizgi Takip Ederken Engel Aşan Robot

Öğrencilerden tanımladıkları ve hakkında fikir ürettikleri programı tamamlamaları ve kontrol edip eksikliklerini gidermeleri istenir.

2.3. Tasarla: Renk sayacı

Dikkat: Bu program için birer adet renk sensörü ve dokunma sensörü ile renk sensörüne gösterilmek üzere farklı renklerde cisimlere ihtiyaç vardır. Renk sensörüne gösterilecek cisimler ışığı fazla yansıtırlarsa programın çalışmasını etkileyebilir. Bu yüzden mat cisimler tercih edilmelidir.

Bu programın amacı renk sensörüne gösterilen farklı renklerin sayısını ekrana yazdırmaktır. Program renk sensörüne gösterilen her yeni renk için sayaç değerini bir artırır ve bu değeri ekrana yazar. Örneğin, program ilk çalıştığında ekrana “0” yazar. Ardından sensöre kırmızı renkli bir cisim gösterildiğinde ekrandaki değer “1”, siyah renkli bir cisim gösterildiğinde “2”, mavi renkli bir cisim gösterildiğinde “3” olur fakat yeniden mavi renkli bir cisim gösterilirse ekrandaki değerde bir artış olmaz. Gösterilen her yeni renk için ekrandaki değer “1” artar. Program dokunma sensörüne basılmasıyla çalışmayı durdurur.

Programı oluşturmaya başlamadan önce grupların tasarlama adımı için yukarıda bir örneği verilen tanımlama ve fikir üretme süreçlerini gerçekleştirmesi gerekir.

2.4. Üret: Renk sayacı

Öğrencilerden tanımladıkları ve hakkında fikir ürettikleri programı tamamlamaları ve kontrol edip eksikliklerini gidermeleri istenir.

Modifiye Renk Sayacı

Bir önceki renk sayacı kendine gösterilen cisimlerin renklerini algılayıp bunları sayıyordu. Bu programın bir önceki programdan tek farkı robotun yerde bulunan ardışık renkler üzerinde ilerlerken kaç adet yeni renk bulduğunu sayması.

2.5. Tasarla: Şifreli Sayıyı Bulma Yarışması

Bu sorunun öğrenciler tarafından çözülebilmesi için öncelikle öğrencilerin ikilik sayı tabanını ve sabitleri bilmesi gerekir. Sabitlerin öğretmen tarafından öğrencilere anlatılacağı varsayıldığından burada bahsedilmeyecektir. Rehber öğretmen sabitleri kendisi anlatmalıdır. İkilik sayı tabanı şu şekilde anlatılabilir.

i. Öğrencilerden aşağıdaki örüntünün kuralını bulmaları istenir.
   1 2 4 8 16 32 64 …
ii. Örüntünün devamında hangi sayıların geleceği sorulur.
iii. Öğrencilere ikilik sayı tabanındaki sayıların 0 ve 1 rakamından oluştuğu anlatılır. Birkaç tane sayı örneği verilir.
iv. Öğrencilere ikilik bir sayının basamaklarına ayrılması aşağıdaki kutu tekniği ile anlatılır:
   1011 sayısı aşağıdaki şekilde basamaklarına ayrılır.

1	0	1	1

v. Yukarıda bulunan örüntünün elemanları sağdan sola doğru ve basamaklarına ayrılan sayı ise soldan sağa doğru olacak şekilde alt alta yazılır. Her bir sütundaki iki sayı çarpılır ve tüm sonuçlar toplanır.

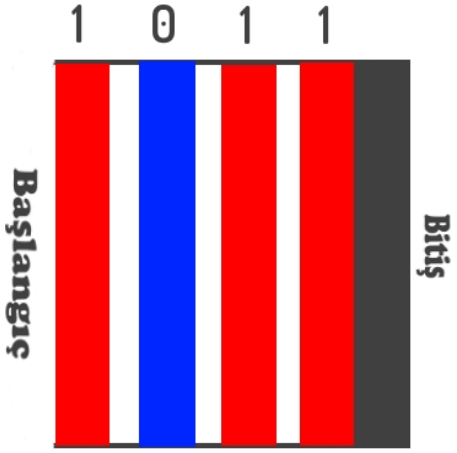
8	4	2	1
1	0	1	1

   8*1+4*0+2*1+1*1=11. Sonuç ikilik tabandaki bir sayının (1011) onluk tabandaki karşılığıdır. Bu şekilde ikilik tabandaki sayılar onluk tabana çevrilir.

vi. Öğrencilere bir iki tane örnek verilerek onluk tabana çevirmeleri istenir.

Görev Tanımı:

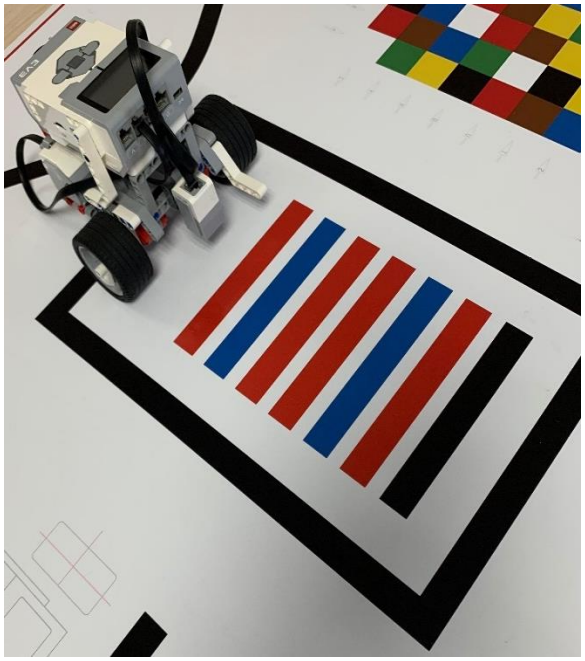
Renkler kullanılarak şifreli sayılar oluşturulur. Örneğin aşağıdaki resimde görüldüğü üzere kırmızı renk 1'e mavi renk ise 0'a karşılık gelir. Beyaz rengin sayısal bir karşılığı yoktur. Siyah renk ise şifrenin bittiğini gösterir. Kırmızı ve mavi renklere karşılık gelen sayılar sırasıyla yan yana yazılarak şifreli sayı bulunur. Aşağıdaki resimde şifreli sayı ikilik tabandaki "1011" sayısıdır.



Resim 107. Şeklin İkilik Taban Karşılığı

Aşağıdaki resimde görüldüğü üzere mat üzerinde şifreli bir sayı bulunmaktadır. Robot matta soldan sağa doğru ilerlemeli ve siyah kısma geldiğinde durarak şifreyi ekranına yazmalıdır. Programı doğru çalışan ve şifreli sayının onluk tabandaki karşılığını ilk bulan grup yarışmayı kazanır.

Programı oluşturmaya başlamadan önce grupların tasarlama adımı için yukarıda bir örneği verilen tanımlama ve fikir üretme süreçlerini gerçekleştirmesi gerekir.



Resim 108. Kullanılacak Mat Bölümü

Dikkat: Öğrenciler robotu çok hızlı hareket ettirmemelidirler. Hızlı giden robot yanlışlıkla masadan düşebilir. Öğrenciler robotun hızı konusunda uyarılmalıdır.

2.6. Üret: Şifreli Sayıyı Bulma Yarışması

Öğrencilerden tanımladıkları ve hakkında fikir ürettikleri programı tamamlamaları ve kontrol edip eksikliklerini gidermeleri istenir.

3. DEĞERLENDİR

Bu aşamada hedef, öğrencilerin öğrenme sürecinde yaşadıkları ve öğrendikleri üzerine düşünmesini sağlamaktır. Bu sayede öğrencilerin, problem çözme yetenekleri gelişecek, öğrenciler dersin konusu ve kendileri ile ilgili gözlemler yaparak öğrendikleri yeni konuları ve kendilerini değerlendirmekle beraber sonraki çalışmalarını planlamak için de fırsat bulurlar. Öğrencilerden şu soruları yanıtlamaları istenebilir:

- Verilen problemleri nasıl tanımlarsınız? (problemi kendi cümleleri ile ifade etme)
- En çok hangi görevde zorlandınız? Bu zorlukların üstesinden nasıl geldiniz? (Problemin çözümü için hangi stratejileri kullandınız ve neden bu stratejileri seçtiniz?) Yeteri kadar tartışma ortamı oluşmazsa rehber öğretmen aşağıdaki soruları kullanarak tartışma ortamı yaratmaya çalışır.
    - Çizgi takip ederken engel aşan ve tekrar çizgi takip etmeye devam eden robot uygulamasında, engeli aştıktan sonra çizgiyi tekrar takip etmek için ne kullandınız? Başka hangi yöntemlerle bu görevi yerine getirebilirdiniz?
    - Renk sayacı uygulamasında, sadece farklı renklerde (renk geçişlerinde) robotun sayacı artırmasını nasıl sağladınız?
    - Şifreli sayıyı bulma yarışmasında, değişkenleri kullanmadan beyaz renklerin sayılmamasını sağlayabilir miydiniz?
- Problemi çözerken ne gibi sıkıntılar yaşadınız ve bunların üstesinden gelmek için neler yaptınız?
- Kullandığınız yöntemler bu sıkıntıları gidermekte başarılı oldu mu?
- Grup arkadaşınızla fikir ayrılıkları yaşadınız mı? Bunların üstesinden gelmek için neler yaptınız?
- Grup arkadaşınızdan ne/neler öğrendiniz?

Değerlendirme, öğrencileri sıkmadan, her bir soru için verilen cevaplar tatmin edici bir düzeye ulaşıncaya kadar devam ettirilir.

7. Hafta: Robotlarla Paralel İşlemler

Ön Bilgi:

- Öğrenciler robot kavramını temel düzeyde bilir.
- Öğrenciler robot setiyle farklı robot tasarımları yapmıştır.
- Öğrenciler robotu programlamak için EV3 yazılımını kullanmıştır.
- Öğrenciler robotun hareket etmesi için gerekli programlama adımlarını oluşturmuştur.
- Öğrenciler farklı sensörlerin kullanıldığı programlar oluşturmuştur.

Haftanın Kazanımları:

- Öğrenciler paralel işlemleri uygular.
- Öğrenciler EV3 yazılımının deney arayüzünü kullanır.

Haftanın Amacı:

Bu haftanın amacı, öğrencilerin robotların herhangi bir veya birden fazla sensörünün algıladığı verileri bir dosya olarak kaydetmelerini ve bu verilerin zamana göre değişim grafiklerini inceleyip yorumlamalarını sağlamaktır. Ayrıca birbiriyle çelişmemek kaydıyla programların paralel olarak çalıştırılması, yani robotun farklı görevleri eş zamanlı olarak yerine getirecek şekilde programlanabilmesi amaçlanmaktadır.

Kullanılacak Malzemeler:

Robot seti, bilgisayar, mat (çalışma alanı)

Haftanın İşlenişi:

Gözle: Paralel işlemleri gerçekleştirmek için birden fazla başla bloğu kullanma, kablo ile blokları birbirlerine bağlama ve deney arayüzünde sensör verilerinin grafiklerini inceleme

Uygula: Paralel işlemlerin gerçekleştirilmesi ve deney arayüzünde verileri inceleme

Tasarla: Otonom olarak veri toplayan robot oluşturma ve verileri değerlendirme

Üret: İstenen programları gerçekleştirme

Değerlendir: Haftanın içeriği ile ilgili yansıtma etkinliği

1. GÖZLE VE UYGULA

1.1. Gözle: Paralel İşlemler

Bu haftada öğrencilere birden fazla program bloğunun eş zamanlı olarak nasıl kullanılacağı öğretilir. Bugüne kadar yaptıkları çalışmalarda program genel olarak bir kanal üzerindeki

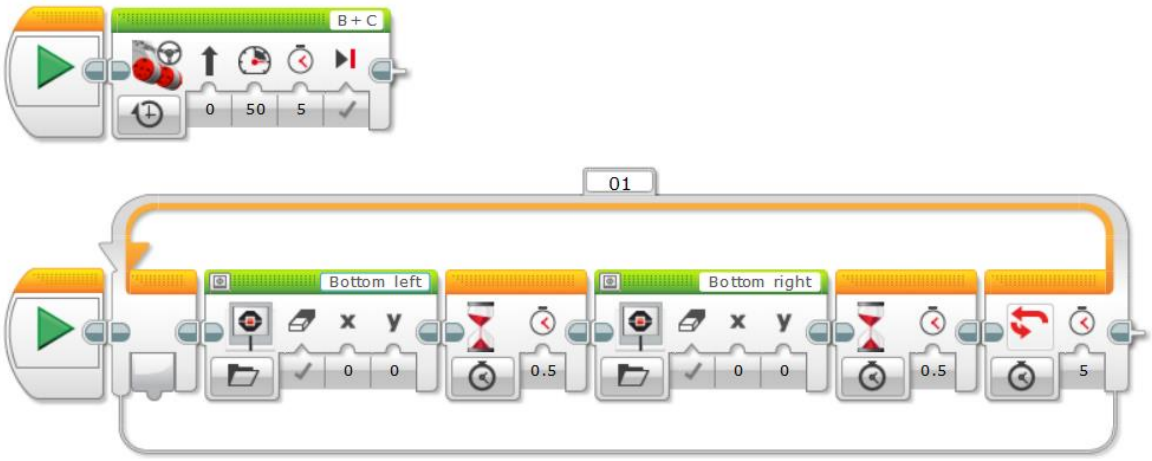
belirtilen komutları sırayla gerçekleştiriyordu. Bu haftada ise iki işlemin, örneğin robotun ileriye doğru hareket etmesi ile orta motor aracılığıyla kolun hareket etmesinin eş zamanlı olarak nasıl gerçekleştirilebileceği konusu işlenir.

Robotun iki işlemi aynı anda yapması iki şekilde gerçekleştirilebilir.

Birden Fazla Başla Bloğu Kullanmak

EV3 yazılımında programlama alanında bir adet başla bloğu bulunur; diğer programlama blokları başla bloğuna eklenerek bir sıra hâlinde program oluşturulur. Akış Kontrolü (Flow Kontrol) sekmesinde yer alan "Başla (Start)" bloğu programlama alanına sürüklenerek birden fazla program akış kanalı oluşturulabilir. Böylece program iki farklı kanalı eş zamanlı olarak çalıştırır.

Öğrencilere 5 saniye boyunca, ileri giderken sağa ve sola bakan göz ifadelerinin sürekli değiştiği (yani etrafına bakınıyormuş gibi ilerleyen) robot programı gösterilir.



Resim 109. Örnek Program

Hazırlanan programda iki adet "başla (start)" bloğu bulunur. Program başladığı anda her iki başla bloğu da çalışmaya başlar. Böylece program tek bir kanaldan değil, iki kanaldan eş zamanlı olarak ilerler. Üst kanalda B ve C motorları ile ileriye doğru 5 saniyelik hareket devam ederken alt kanalda iki farklı göz resmi 5 saniye boyunca 0,5'er saniyelik aralıklarla sürekli değişir.

Not

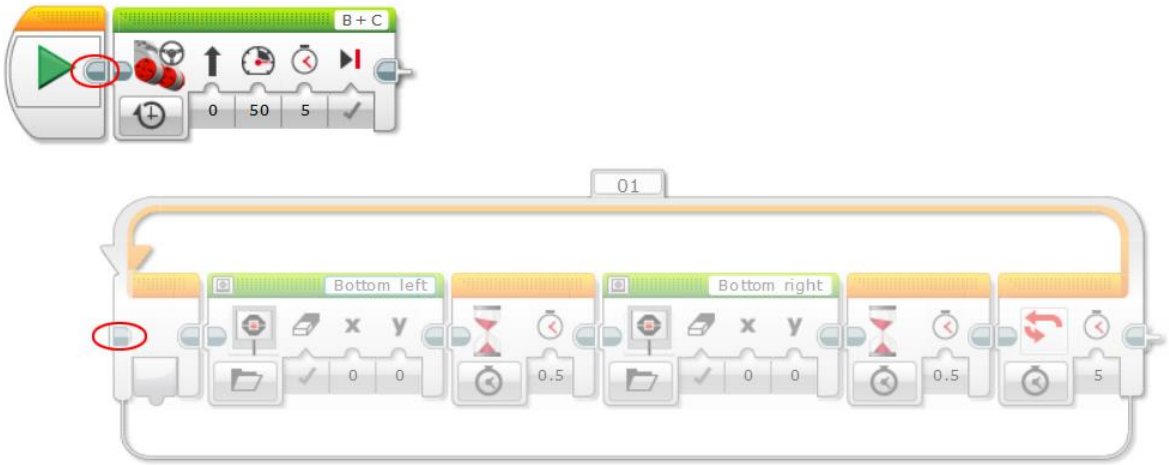
Paralel işlemlerde önemli olan nokta, kanallarda birbiriyle çelişen komutların yer almamasıdır. Örneğin, üst kanalda B ve C motorları ileri yönde hareketi sağlarken alt kanalda B ve C motorlarına geriye doğru hareket komutu verilmemelidir. Bu durumda robotun hangi kanaldaki komutu gerçekleştireceğini tahmin etmek olanaksızdır.

Blokları Kablo ile Bağlamak

Birden fazla başla bloğu kullanıldığında, programda gerçekleştirilmesi istenen paralel işlemlerin mutlaka program başladığı andan itibaren başlaması gerekir. Robotun belirli

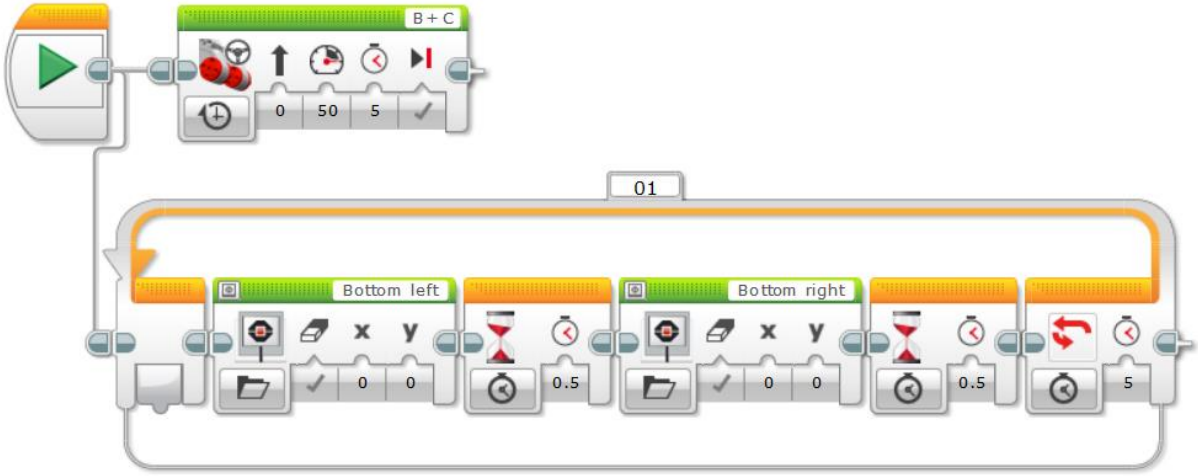
işlemleri bir kanalda gerçekleştirdikten sonra, paralel işlemleri gerçekleştirmesi istendiğinde kablo bağlantısı kullanmak gerekir.

Önceki örnekte iki farklı başla bloğu ile gerçekleştirilen robotun hareketi ve ekranda görüntünün değiştirilmesi etkinliği kablo ile de gerçekleştirilebilir.



Resim 110. Adım 1

Kablo bağlantısı yapılmak istenen program bloğunun sonunda yer alan bağlantı bölmesi, üzerine fare ile sol tıklanarak, bağlanmak istenen program bloğunun başlangıcında yer alan bağlantı bölmesine sürüklenir. Böylece aktif olmayan program blokları aktif hâle getirilir.



Resim 111. Adım 2

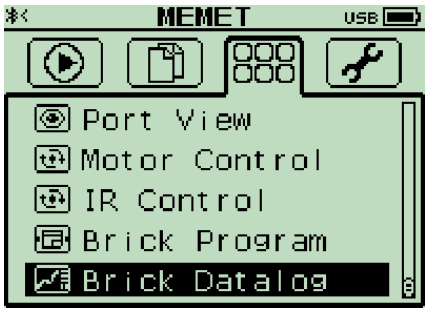
Bu yöntemle, birden fazla başla komutu kullanmadan, programın istenen noktasından itibaren paralel işlemler başlatılabilir. Yani programın ilerleyen herhangi bir noktasında robota eş zamanlı görevler verilebilir.

1.2. Uygula: Paralel İşlemler

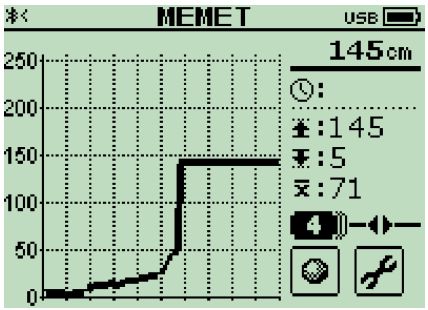
Öğrencilerden hareket hâlindeyken orta motora bağlı kolu indirerek herhangi bir nesneyi alan robot programını oluşturmaları istenir. Programda paralel işlemleri kullanmaları gerektiği, robotun nesneyi alırken durmaması ve hareketine devam etmesi gerektiği vurgulanır.

1.3. Gözle: Deney Arayüzü ve Veri Kaydı

EV3 robot setini kullanarak farklı yöntemlerle "Veri Kaydı (Data Logging)" yapılabilir. Bunlardan birincisi, akıllı tuğla üzerinde üçüncü sekmeden "Tuğla Veri Kaydı (Brick Datalog)" menüsünün seçilmesidir (Robotun Firmware versiyonuna bağlı olarak Brick Datalog menüsü olmayabilir).



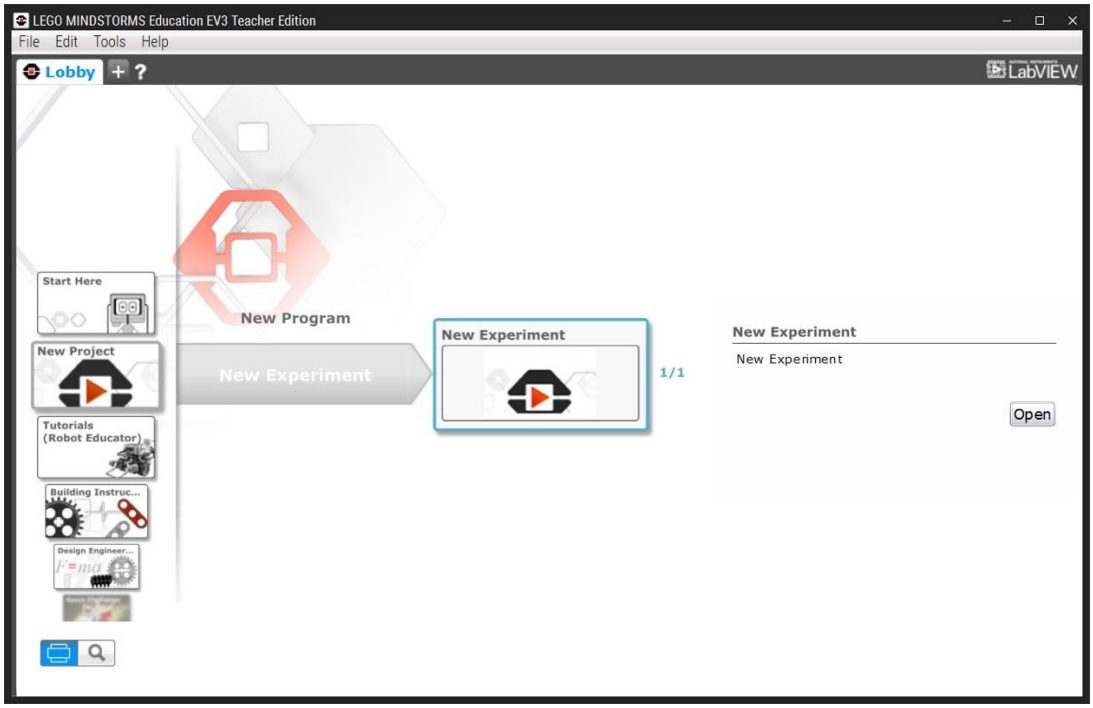
Resim 112. Veri Kaydı Seçimi



Resim 113. Veri Grafiği

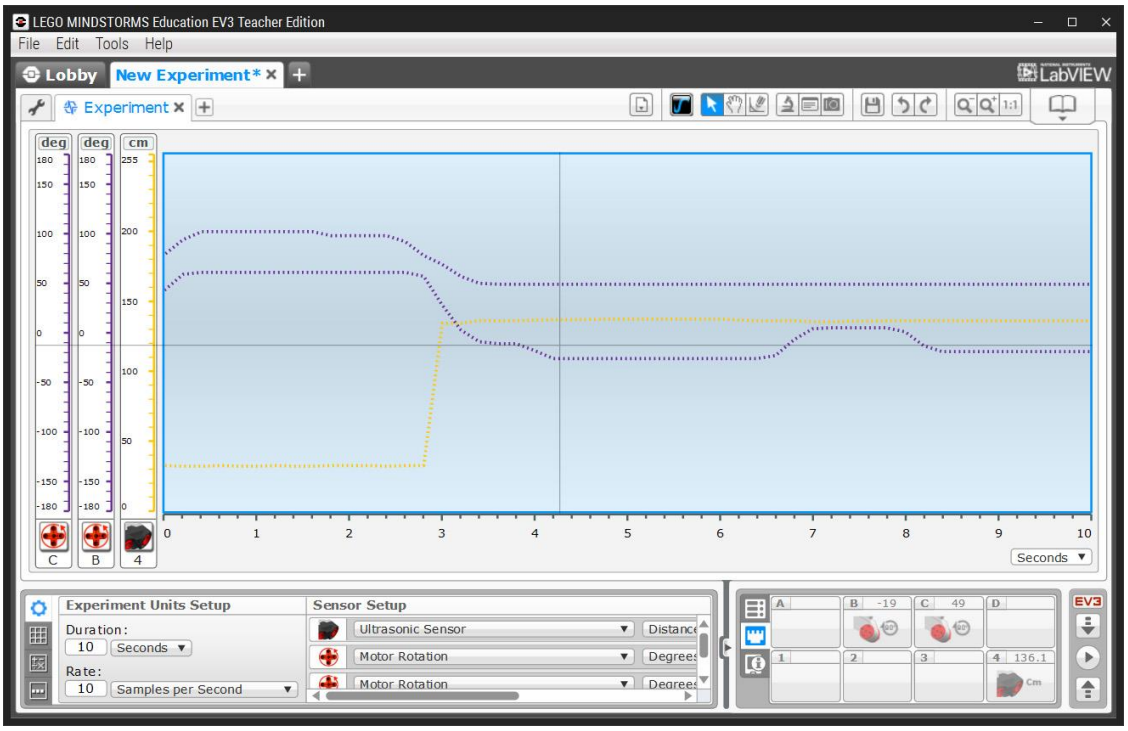
İstenen sensör veya motor portu seçilerek o porttan gelen veri grafiğinin ekranda çizildiği gözlemlenebilir.

İkinci olarak EV3 yazılımı robotun sensörleri aracılığı ile topladığı verilerin grafiğini çizdirerek üzerinde çalışmaya olanak veren deney arayüzüne sahiptir. Deney arayüzü, Lobby ekranında **New Project > New Experiment > Open** yolu izlenerek açılabilir.



Resim 114. Deney Arayüzü

Deney arayüzü açıldığında, eğer robot bilgisayara USB veya bluetooth ile bağlı ise deney ekranında eş zamanlı olarak robotun sensörlerinden alınan verilerin grafiğinin çizildiği görülecektir.



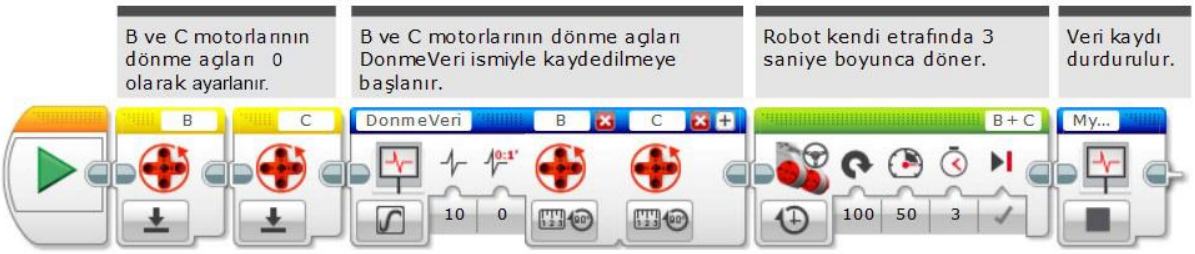
Resim 115. İlgili Veri Grafiği

Öncelikli olarak robotun sadece B ve C motorları verilerini göstermesini sağlayınız, bu amaçla "Sensör Kurulumu (Sensor Setup)" bölümünden diğer sensör verilerini X işaretini tıklayarak kapatınız. Robotun hareketsizken ve bir motorun elle yavaşça döndürülmesiyle oluşan grafikleri inceleyiniz ve öğrencilerle tartışınız. Öğrencilerin motorun dönme açısını ve zaman grafiğini anlamasını sağlayınız. Ayrıca benzer bir etkinliği motor verilerini kapatıp sadece mesafe sensörü verisini açarak yapınız. Robotun bir engele yaklaşırken ve engelden uzaklaşırken oluşan grafiği öğrencilerle tartışınız.

Son olarak robotun bağımsız olarak istenen sensör verilerini akıllı tuğla içinde bir dosyaya kaydetmesinin sağlanması ve bu dosyanın daha sonra deney arayüzünde incelenmek üzere açılabilmesi de mümkündür. Bu amaçla öncelikle robotun istenen sensör verilerini toplamasını ve bu verileri bir dosyaya kaydetmesini sağlayan programın hazırlanması gerekir. EV3 yazılımı programlama arayüzünde, mavi renkli "İleri Düzey (Advanced)" sekmesinde yer alan "Veri Kaydı (Data Logging)" bloğu ile istenen sensör verilerinin kaydedilmesi sağlanır.

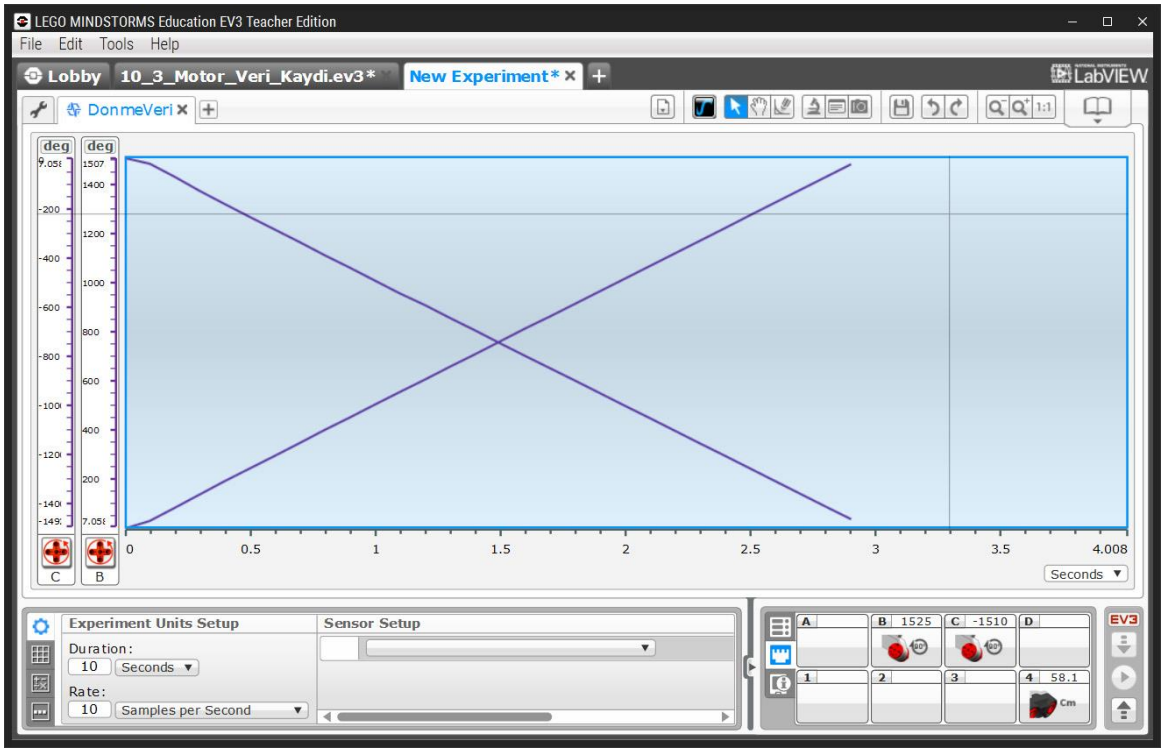
1.4. Uygula: Motor Hareket Verisinin İncelenmesi

Öğrencilerden kendi etrafında dönen robotun B ve C motorlarının dönme verilerini kaydedecek programı hazırlamaları ve programı çalıştırarak verileri robotun akıllı tuğlasına kaydetmeleri istenir. Sonrasında EV3 yazılımı deney arayüzünde verilerin grafiğini inceleyip tartışmaları istenir. Bu amaçla aşağıdaki gibi bir program ile motor verilerinin tuğla içinde DonmeVeri dosyasına kaydedilmesi sağlanır.



Resim 116. Örnek Program

Öğrencilerden, robotu bilgisayara bağlayarak deney arayüzünde, “Araçlar (Tools)” menüsünden, “Veri Kaydı Dosya Yöneticisi (Data Log File Maneger)” açılarak çalıştırılan program klasörüne kaydedilmiş DonmeVeri dosyasını açmaları ve tartışmaları istenir. Aşağıda DonmeVeri dosyası deney arayüzünde açıldığında oluşan grafik görülmektedir.

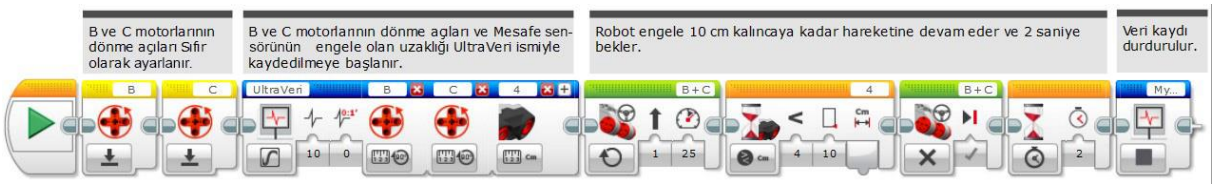


Resim 117. Veri Grafiği

.

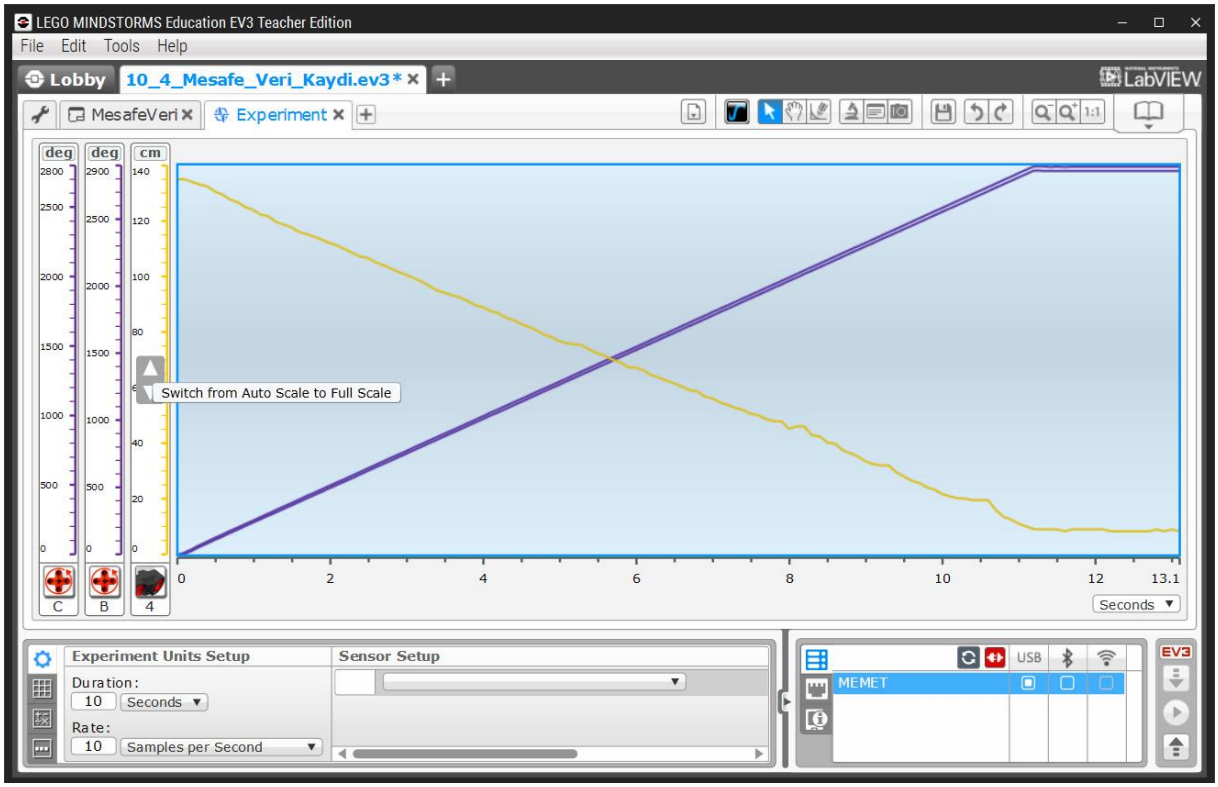
1.5. Uygula: Motor ve Mesafe Sensörü Verilerinin Karşılaştırılması

Bu etkinlikteki amaç, robot bir engele yaklaşırken mesafe sensörü ve motorlardan alınan verilerin deney arayüzünde incelenmesidir. Öğrencilerden robotlarını bir engele 10 cm kalıncaya kadar düz gidecek şekilde programlamaları istenir. Robotun bu hareketi esnasında motorlarının dönme açısını ve mesafe sensörünün ölçtüğü mesafeyi veri dosyası olarak kaydetmesi gerektiği belirtilir. Verinin daha rahat incelenebilmesi için öğrencilerden motorların gücünü 10-25 olarak ayarlamaları istenir. Deney arayüzünde bu verilerin grafiğini çizdirerek tartışmaları istenir. Örnek program aşağıda sunulmuştur.



Resim 118. Örnek Program

UltraVeri dosyası EV3 deney arayüzünde açıldığında aşağıdaki gibi bir grafik elde edilir. Bu grafik öğrencilerle tartışılır.



Resim 119. Veri Grafiği

Not

Robot bilgisayara bağlı olduğu sürece robota takılı olan sensörlerin canlı verileri de deney arayüzünde görülür. Daha sade bir arayüzde çalışmak için alt kısımda bulunan Sensör Kurulumu (Sensor Setup) bölümünde canlı olarak gösterilen sensörler kapatılabilir. Ayrıca grafiği çizilen sensör verilerini ekrana sığdırabilmek için fare sensör sütunu üzerine tıklanarak otomatik sığdırma (auto scale) veya tam sığdırma (full scale) görünümleri arasında geçiş yapılabilir.

2. TASARLA

2.1. Tasarla: Renk Tahmini Oyunu

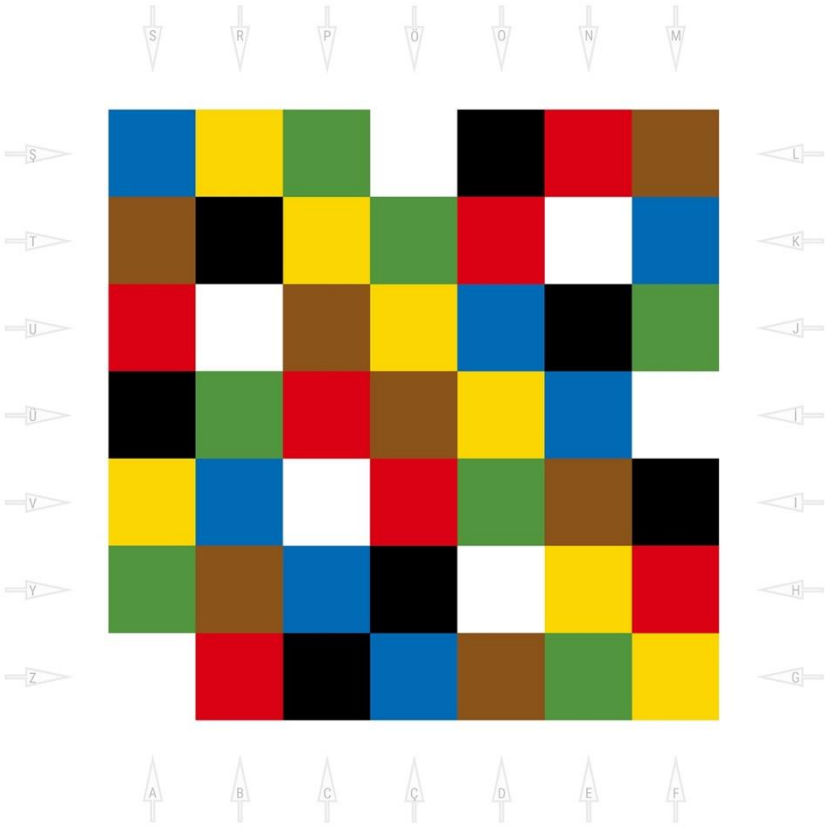
Bu etkinlikte öğrencilere, robotlarını veri toplama sürecinde görmeyecekleri, robotlarını bilinmeyen bir noktaya veri toplamak amacıyla gönderecekleri ve toplanan verileri analiz ederek robotlarının hangi renkte bir zeminde gittiğini tahmin etmelerinin isteneceği söylenir.

Not

Renk sensörlü temel tasarıma (colour sensor down - driving base) erişmek için gerekli yönergeye aşağıdaki yollarla ulaşılabilir:

i. https://le-www-live-s.legocdn.com/sc/media/lessons/mindstorms-ev3/building-instructions/ev3-rem-color-sensor-down-driving-base-d30ed30610c3d6647d56e17bc64cf6e2.pdf
ii. EV3 yazılımı > Lobby > Building Instructions > Building Ideas > Color Sensor Down - Driving Base
iii. Robot seti ile birlikte gelen kitapçığa bakılabilir.

Gruptaki öğrencilerden biri, robotunu 23 cm düz gidecek ve ilerlerken de zeminden yansıyan ışık miktarına ait verileri toplayacak şekilde programlar. Grup arkadaşı, programlanan robotu mat üzerinde yer alan renkler damasındaki herhangi bir rotada (A'dan Z'ye isimlendirilmiştir) düz gidecek şekilde çalıştırarak veri kaydının yapılmasını sağlar. Robotu çalıştıran öğrenci renkler damasının 1 cm dışındaki beyaz alandan ölçümü başlatmalı ve robot beyaz alana çıkınca ölçüm sonlanmalıdır (renkler damasının bir kenarı 21 cm uzunluğundadır). Veri toplama sürecinde programı hazırlayan öğrenci robotun hangi rotadan ilerlediğini görmemelidir. Sonraki aşamada, veri dosyası kaydedilen robot, programı hazırlayan öğrenciye verilir ve öğrenci veri dosyasını deney arayüzünde açıp analiz ederek robotun hangi rotada ilerlediğini tahmin eder. Yansıyan ışık miktarından rengi tahmin etmek güç olabilir, öğrenciler beyaz ve siyah alanlardan çıkarımda bulunabilir ve takıldıkları noktalarda robot üzerinde üçüncü sekmede bulunan Giriş Görünümü (Port View) menüsünden mat üzerindeki renklerin yansıma miktarını ölçebilirler. Öğrenciler rolleri değişerek etkinliği tekrar ederler.



Resim 120. İlgili Mat Bölgesi

Tanımlama: Öğrencilerin problemi tanımlamaları gerekir. Öğrenciler, problemi çözmelerine yardımcı olacak şu soruların cevaplarını kendi aralarında tartışırlar:

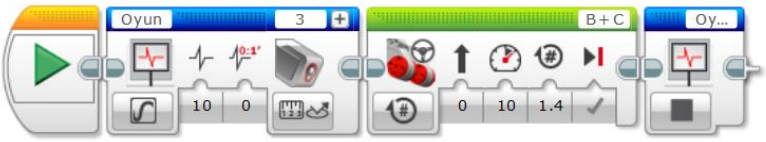
- Robot zeminden yansıyan ışık miktarını nasıl algılayıp kaydedebilir?
- Yansıyan ışık miktarı renkleri ayırt etmek amacıyla kullanılabilir mi?
- Işığı en fazla yansıtan renk hangisidir? Işığı en az yansıtan renk hangisidir?
- Grafik nasıl yorumlanır?

Fikir üretme: Bu aşamada öğrencilerden robotun yukarıdaki işlemleri nasıl gerçekleştirebileceği ile ilgili fikir yürütmeleri beklenir.

3. ÜRET

Birinci öğrenci gerekli programı hazırlayıp robota yükledikten sonra (öğrencinin robotun ilerlediği rotayı görmemesi sağlanarak), ikinci öğrenci renkler daması üzerinde robotu çalıştırıp veri toplama sürecini gerçekleştirir ve robotu birinci öğrenciye teslim eder. Birinci öğrenci

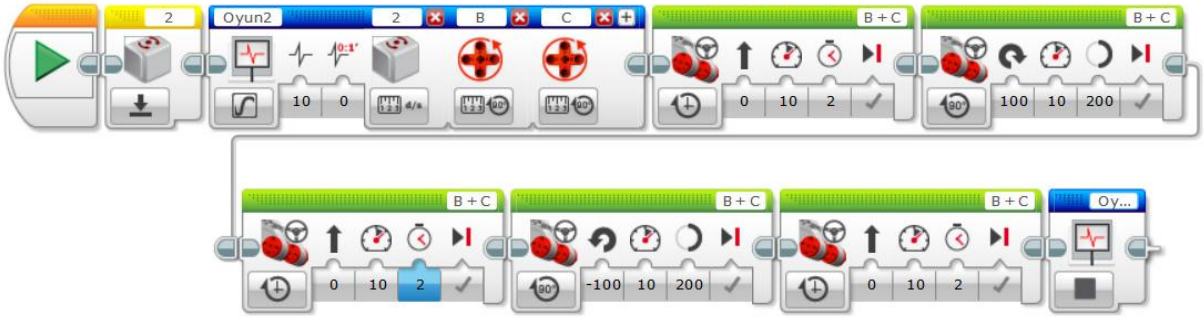
verileri inceleyerek robotun geçtiği rotayı tahmin etmeye çalışır. Sonrasında öğrenciler rolleri değişirler. Örnek program aşağıda sunulmuştur.



Resim 121. Örnek Program

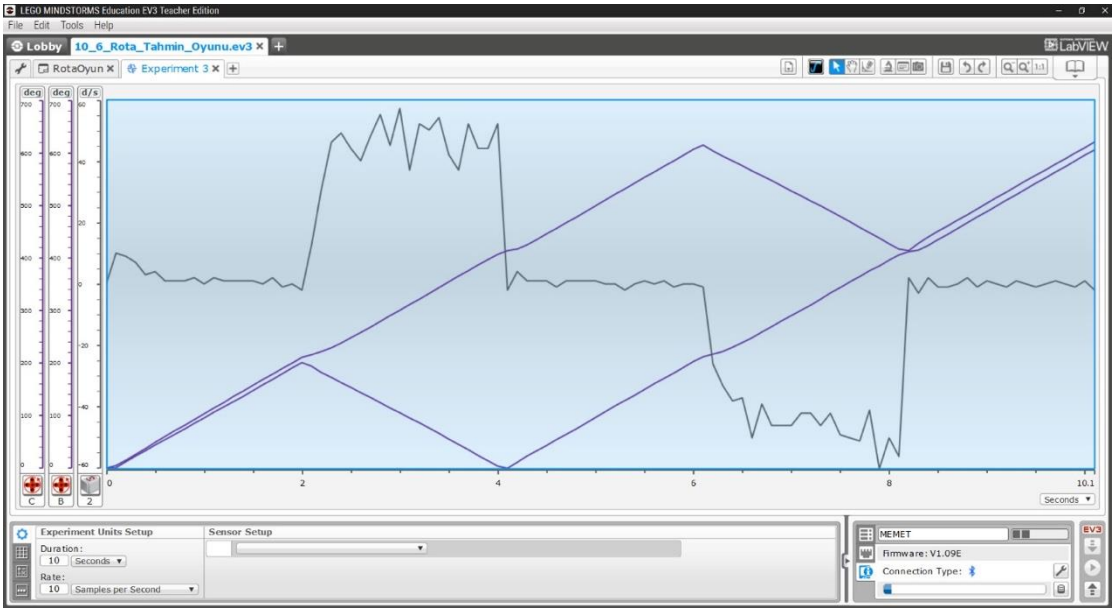
3.1. Motor ve Açı Sensör Verilerini Kullanarak Rota Oluşturma

Bu etkinlikte rehber öğretmen, öğrencilere açı sensörü, B motoru ve C motorundan kaydedilen veri setini hazır olarak verir ve öğrencilerden veri setini inceleyerek robotun izlediği yolu bir kâğıda çizmelerini ister.



Resim 122. Örnek Program

Rehber öğretmen, yukarıdaki programa benzer bir programı kendi bilgisayarında hazırlar. Öğrencilerin program bloklarını ve robotun çalışmasını görmelerine izin vermez. Program çalıştırılınca robot 2 saniye süreyle ileri gider, sonrasında 90 derece sağa döner. Tekrar 2 saniyelik düz bir hareketin ardından bu defa 90 derece sola döner ve 2 saniye daha düz gider. Bu esnada robotun B, C ve açı sensörü verileri Oyun2 adlı dosyaya kaydedilir. Rehber öğretmen tüm bu süreci öğrencilerin göremeyeceği şekilde gerçekleştirir. Sonrasında tüm gruplarla yalnızca Oyun2 veri dosyası paylaşılır. Öğrenciler veri dosyasını deney arayüzünde açıp veri setini inceleyerek robotun izlediği yolu kâğıda çizerler. Deney arayüzünde açılan dosyada aşağıdaki gibi bir grafik oluşur.



Resim 123. Veri Grafiği

Not

Etkinlik daha fazla hareket bloğu ile çeşitlendirilebilir. Rehber öğretmen eğitim öncesinde veri dosyasını hazırlayabilir, etkinlik esnasında da öğrencilerden beklenenleri açıklayıp dosyayı paylaşabilir. Etkinlik öncesinde birden çok veri dosyası hazırlanmalıdır.

4. DEĞERLENDİR

Bu aşamada hedef, öğrencilerin öğrenme sürecinde yaşadıkları ve öğrendikleri üzerine düşünmelerini sağlamaktır. Bu sayede öğrenciler süreç içinde deneyimlediklerini ve öğrendiklerini gözden geçirerek kendilerini değerlendirme ve tanıma fırsatı elde edeceklerdir. Öğrencilerden şu soruları yanıtlamaları istenebilir:

- Robotun birden fazla işlemi eş zamanlı gerçekleştirebildiği görevlere örnekler veriniz.
- Robot paralel işlemler yapabilmesi için nasıl programlanmalı ve nelere dikkat edilmelidir.
- Robotun verileri toplayıp uzmanlara gönderdiği durumlara günlük hayattan örnekler veriniz.
- EV3 yazılımının deney arayüzünde neler yapıldığını özetleyiniz, önemini tartışınız.
- Bilim insanlarının çalışmaları ile bugün yaptığınız etkinlikleri kıyaslayınız.
- Bugün neler öğrendiniz?
- Grup arkadaşınızdan ne/neler öğrendiniz?

Değerlendirme, öğrencileri sıkmadan, her soru için verilen cevaplar tatmin edici bir düzeye ulaşıncaya kadar devam ettirilir.

5. İLAVE ETKİNLİK

5.1. Diğer Grubun Hareketini Tahmin Edip Uygulanan Programı Oluşturma

Bu etkinlik iki grubun karşılıklı etkileşimde olacağı şekilde yapılır. Gruplar aynı anda hareket ve dönme gerektiren bir program yazar, çalıştırır ve buradaki verileri (açı ve hareket) robota kaydeder. Daha sonra bu verileri diğer grupla paylaşır. Kaydı alan grup kaydı oluşturulan robotun hareketini tahmin eder ve aynı hareketleri gerçekleştirecek programı hazırlar.

Not

Bu etkinliğin amacına ulaşması için veri kaydı yapılırken ve veri kaydını oluşturan program yazılırken grupların birbirinin yaptığı işlemleri görmemesi gerekir.

8. Hafta: Kendi Bloğunu Oluşturma - Bluetooth

Ön bilgi:

- Öğrenciler robot kavramını bilir.
- Öğrenciler robot setiyle farklı robot tasarımları yapmıştır.
- Öğrenciler robot setini programlamak için grafik arayüzünü kullanmıştır.
- Öğrenciler robotun hareket etmesi için gerekli programlama adımlarını oluşturmuştur.
- Öğrenciler sensörlerin kullanıldığı programlar oluşturmuştur.
- Öğrenciler wire (kablo) ile bloklar arası veri iletimini kullanmıştır.
- Öğrenciler EV3 yazılımında veriler üzerinde matematik ve mantık işlemleri gerçekleştirir.
- Öğrenciler farklı sensörleri birlikte veya paralel işlemlerde kullanmak için programlama adımlarını oluşturmuştur.
- Öğrenciler programlama adımlarında değişkenler kullanmayı, ileri seviye matematik ve mantık işlemleri kullanmayı, problemi alt parçalara bölüp fonksiyonlar oluşturmayı deneyimlemiştir.

Haftanın Kazanımları:

- Öğrenciler robotun belirlenen işlemleri yapabilmesi için, verilen problemi daha küçük parçalara ayırabilir.
- Öğrenciler her bir küçük program parçası için yapılması gerekenleri tanımlayabilir.
- Öğrenciler her bir program parçasının çözümü için fonksiyonlar oluşturabilir.
- Öğrenciler küçük problemleri çözmek için geliştirilen fonksiyonların verilen problemi çözmek için birlikte (senkronize) çalışması amacıyla gerekli programlama adımlarını oluşturabilir.
- Öğrenciler ortaya çıkan programın verimli ve etkili şekilde çalışması için gerekli önlemleri belirleyip uygulayabilir.
- Öğrenciler EV3 tuğlanın Bluetooth özelliğini kullanır.
- Öğrenciler EV3 tuğlalar arasında mesaj gönderip alabilir.

Haftanın Amacı:

Bu haftanın amacı, öğrencilerin bir problemin çözümünde problemi küçük parçalara bölerek çözüm üretmelerini sağlamaktır. Ayrıca EV3 yazılımının MyBlock özelliğini kullanarak fonksiyon oluşturmaları ve EV3 akıllı tuğlada bulunan Bluetooth özelliğini kullanarak iki akıllı tuğla arasında bağlantı gerçekleştirmeleri amaçlanmaktadır. Ek olarak, bağlı olan iki akıllı tuğla arasında veri alışverişini gerçekleştirerek bir robotun diğer robot ile yönlendirilmesini sağlamaları da hedeflenmektedir.

Kullanılacak Malzemeler:

Robot seti, bilgisayar, mesafe sensörü, renk sensörü, mat (çalışma alanı)

Haftanın İşlenişi:

Gözle: Bir problemi daha küçük alt problemlere dönüştürme ve bluetooth bağlantısı kullanarak tuğlalar arasında mesajlaşmayı sağlama

Uygula: Alt problemlerin program kodlarını oluşturma ve bu kodları fonksiyona dönüştürerek esas problemin çözüm sürecinde birlikte kullanma. Bluetooth Connection (Bluetooth Bağlantısı) ve Messaging (Mesajlaşma) bloklarını kullanma

Tasarla: Belirlenen bir labirent için robot ve program tasarımını yapma. Ayrıca, birbirlerine bluetooth bağlantısı ile bağlı iki robottan birinin gerçekleştirdiği hareketlerin, diğer robot tarafından taklit edilmesi için gerekli programı tanımlama ve planlama

Üret: İstenen robotu oluşturma ve oluşturulan robotun istenen işlemi yapması için gerekli programları oluşturma

Değerlendir: Yansıtma etkinliği

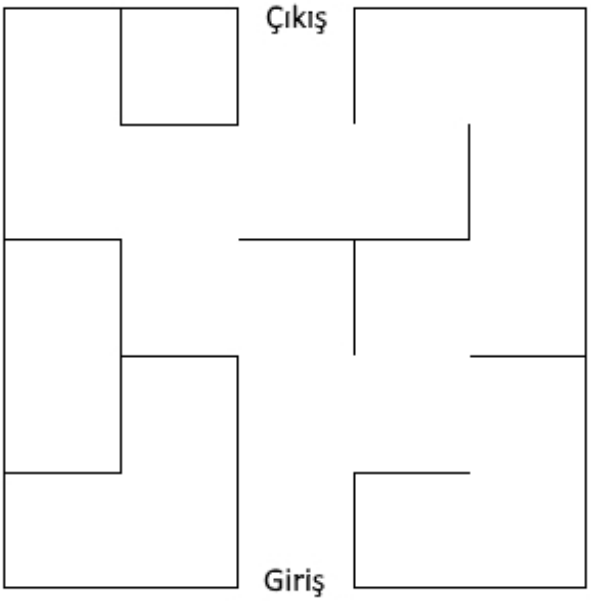
1. GÖZLE VE UYGULA

1.1. Büyük Problemler Küçük Problemlerden Oluşabilir

Programcılar çoğu zaman büyük problemlerle karşı karşıya kalır. Böyle durumlarda, büyük bir problemi olduğu gibi çözmek yerine daha küçük parçalara ayırarak küçük problemler için çözüm üretmek ve çözümleri birleştirerek büyük problemin çözümüne ulaşmak iyi bir yaklaşım olabilir. Bu yöntem çeşitli avantajlar sağlayabilir. Bu avantajlardan en önemlisi zor bir problemin basitleştirilmesidir. Problemi bütün olarak çözmektense onu oluşturan her bir parçayla ayrı ayrı uğraşmak daha kolay olabilir. Bunun yanında, programcı büyük bir program oluşturmaya çabalamak yerine küçük programlarla uğraşacağından kod yazma ve hata ayıklama kolaylaşacaktır. Büyük bir programı takip etmek veya ondaki hatayı aramaktansa program parçacıklarının içindeki hatayı aramak daha kolay olacaktır. Ayrıca yazılan program parçaları, belirli bir işlem yapan birimler olarak düşünülebilir ve başka programların içinde yeniden kullanılabilir. Böylece bir programcı tarafından yazılan kod, kolaylıkla o programcı veya başka programcılar tarafından farklı projelerde yeniden kullanılabilir.

1.2. Labirent Çözme

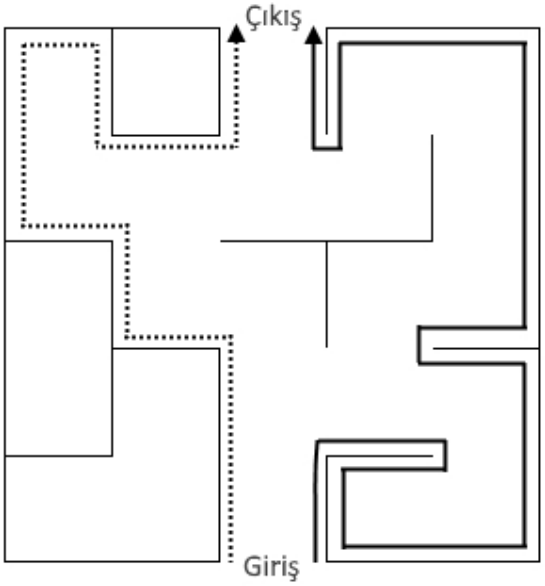
Rehber öğretmen dersin başında öğrencilere yapılacak etkinliklerde iki adet mesafe sensörü kullanılacağı için ikişerli gruplar (4 öğrenci) hâlinde çalışacaklarını söyler. Aşağıda örnek bir labirent gösterilmiştir (**Dikkat:** Bu labirent öğretim amaçlıdır. Etkinlikte kullanılacak labirent 2.1. Labirent Yarışması başlığı altında sunulmuştur).



Resim 124. Örnek Labirent

Labirent problemlerinde esas olan, labirentten çıkmak için doğru yolu bulmaktır. Labirent problemlerinin çözümü için birçok algoritma bulunmaktadır. Sağ duvar algoritması en basitlerinden biridir. Girişten itibaren sürekli sağ duvar takip edilerek ilerlenirse sonunda çıkışa ulaşılabilir. Buna benzer şekilde girişten itibaren sol duvar takip edilerek de çıkışa ulaşılabilir. Aşağıda düz çizgi ile gösterilen çözüm sağ duvar algoritmasının sonucu iken noktalı çizgi ile

gösterilen çözüm sol duvar algoritmasının çözümüdür. Aslında bu iki algoritma aynıdır. Sadece takip edilecek duvarın yönü farklı seçilerek iki farklı çözüm oluşturulabilir.



Resim 125. Sağ/Sol Duvar Algoritması

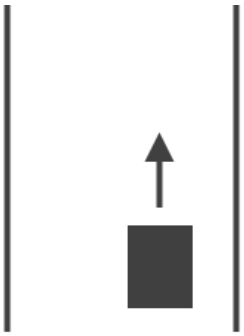
Sağ duvar algoritması her türlü labirent problemini çözmeyebilir. Eğer labirentin giriş veya çıkış noktası, labirentin kenarlarında değilse ya da labirentte hareket ederken başlanan yere geri dönme ihtimali varsa (döngü varsa) sağ duvar algoritması labirent problemini çözmede başarılı olmayabilir.

Sağ duvar algoritması çıkış için en kısa yolu bulamayabilir. Bu açıdan değerlendirildiğinde verimi düşük bir algoritmadır. Fakat bu dersin kapsamında sağ duvar algoritmasını göstermek uygun olacaktır.

1.3. Görevler

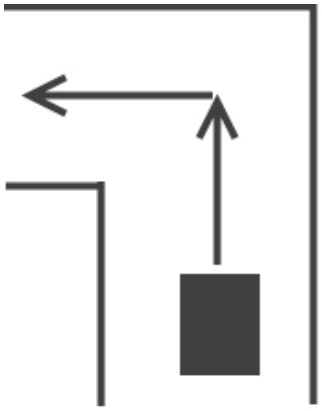
Sağ duvar algoritması için kullanılacak görevlerin belirlenmesi gerekir.

i. Aşağıda gösterildiği gibi sağ tarafında duvar varken robot düz bir şekilde ilerler. Bunu yaparken ne sağdaki duvara çok yaklaşmalıdır ne de ondan fazlaca uzaklaşmalıdır.



Resim 126. Robotun Düz İlerlemesi

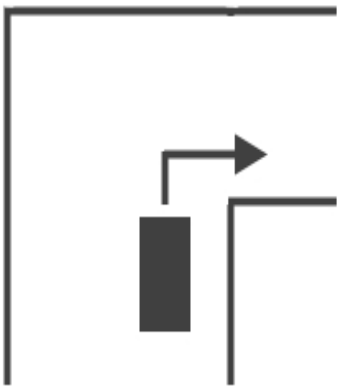
ii. Aşağıdaki şekilde görüldüğü üzere hem karşısında hem de sağında duvar varsa robot sola dönmelidir.



Resim 127. Robotun Sola Dönmesi

iii. Duvarın sağ tarafında boşluk olduğu durumlarda robot sağa dönmelidir. Aşağıdaki resimlerde bu durumlar görülebilir.

Resim 128. Sağa Dönme Durum 1



Resim 129. Sağa Dönme Durum 2

Resim 130. Sağa Dönme Durum 3

1.4. Robot Gereksinimleri

Bu etkinlikte robotun ileriye ve sağ tarafına bakacak şekilde iki tane mesafe sensörü olması gerekir. İleriye bakan sensör karşı duvarı algılamak için kullanılır. Sağa bakan sensör ise robotun sağ duvarı takip etmesi ve sağ duvarda bir açıklık olup olmadığını belirlemesi için kullanılır.

Resim 131. Robot Tasarımı

1.5. Alt Problemler

Sağ duvar algoritmasını kullanarak labirentten çıkma probleminin alt problemleri belirlenmelidir. Alt problemler aşağıdaki şekildedir:

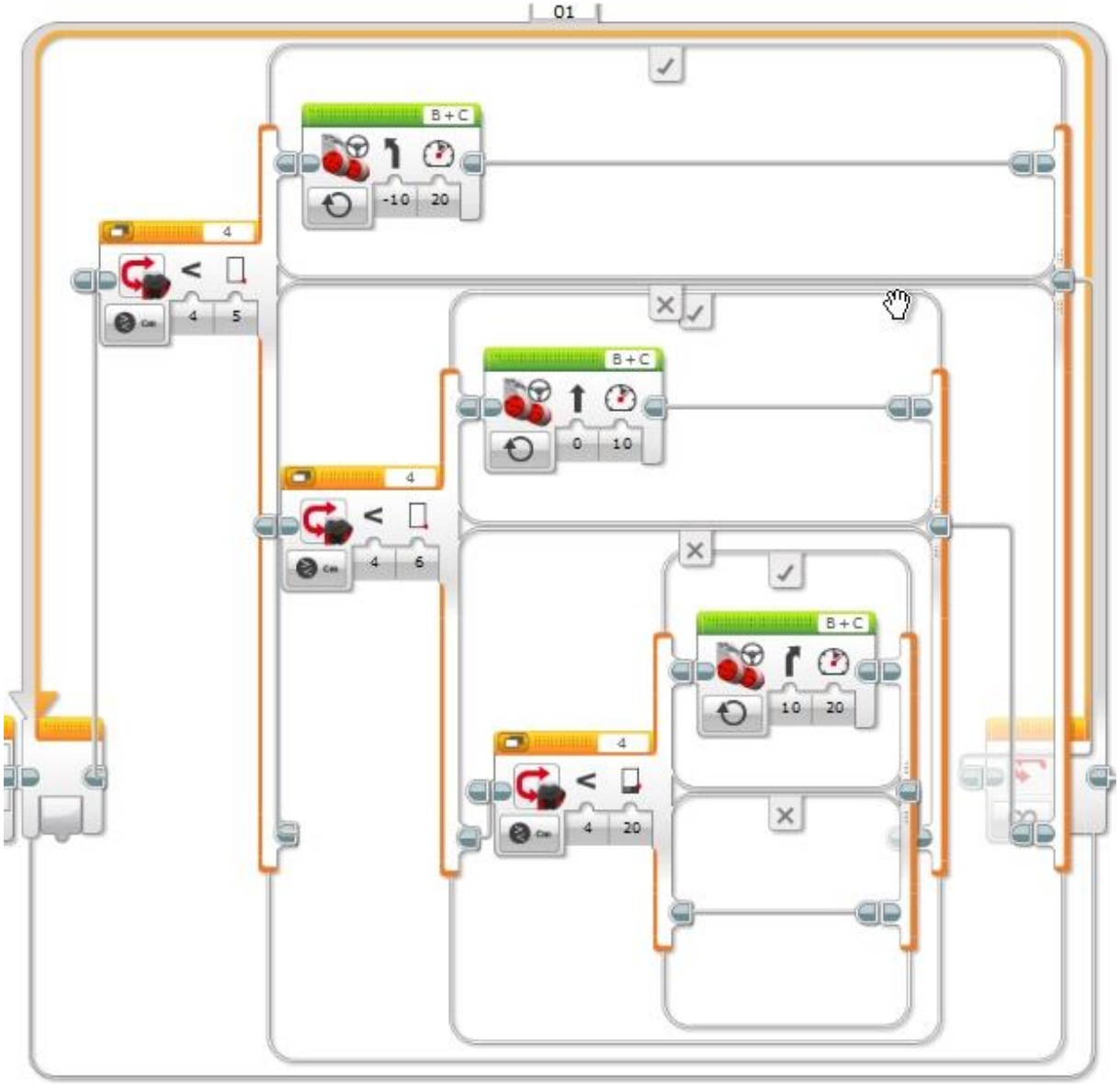
i. Robot sağ duvara ne çok yakın ne de çok uzak olacak şekilde ileriye doğru gitmelidir (duvar takip),
ii. Karşıda bir duvar varsa ve sağ duvarda açıklık yoksa robot sola dönmelidir (sola dönme),
iii. Sağ duvarda bir açıklık varsa robot sağa dönmelidir (sağa dönme),
iv. i, ii ve iii numaralı adımlardaki işlemler sürekli tekrarlanmalıdır (tekrarlama).

1.6. Uygula: Duvar Takip

Bu görevde robotun, sağındaki duvarı yaklaşık olarak 6 cm mesafede takip etmesi gerekir. 6 cm'lik mesafenin sürekli sabit kalması zorunlu değildir. Bu bazen 9, bazen 3 olabilir. Yani belirli bir aralıkta olmak kaydıyla robot sağdaki duvarı takip etmelidir. Bu görev için 4 numaralı porta takılmış olan (sağa bakan) mesafe sensörü kullanılır. Duvar takip algoritması şu şekilde özetlenebilir:

i. Robot 5 cm'den daha fazla duvara yaklaştıysa sola doğru 20 hızında 10 derece hareket ettirilir.
ii. Robotun duvara olan mesafesi 5 ve 6 cm arasındaysa robot 10 hızında ileri doğru gider.
iii. Robotun duvara olan mesafesi 6 cm'den fazla 20 cm'den az ise robot 20 hızında sağa doğru 10 derece hareket ettirilir.

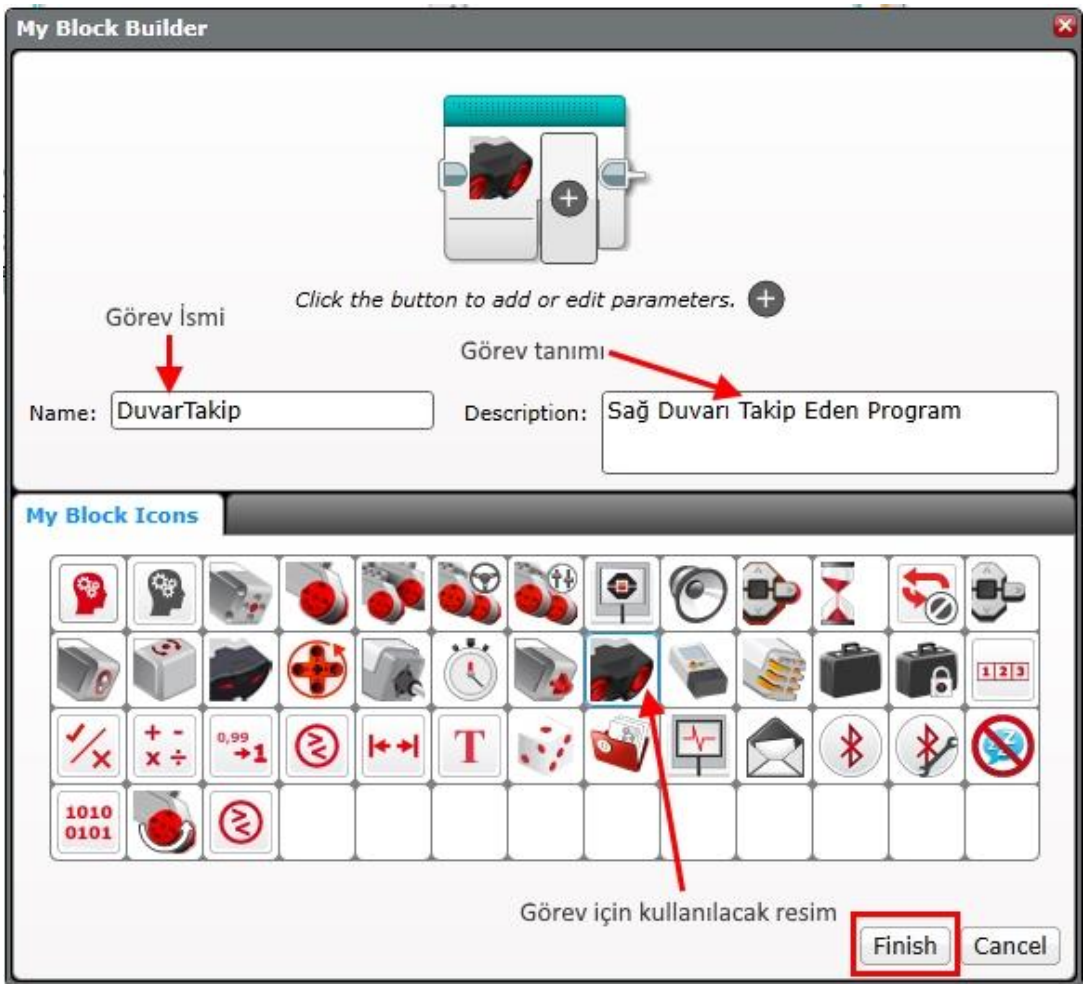
Görevin kodu aşağıdaki gibidir. Bu kod ile robotun sağda bulunan duvarı takip etmesi sağlanır. Robot sağ duvara yakınsa (mesafe < 5 cm) sola doğru yönlendirilir. Sağ duvar ile arasındaki mesafe yeterli ise (5 cm < mesafe < 6 cm) düz gitmesi sağlanır. Eğer sağ duvardan uzaklaşmışsa robot sağa doğru yönlendirilir (6 cm < mesafe < 20 cm). Burada kod içinde kullanılan 20 cm sınırının açıklanması yerinde olacaktır. Duvarda bir açıklığın bulunduğunu belirlemek için eşik değeri 20 cm olarak belirlenmiştir. 20 cm'ye kadar olan açıklıklarda duvarda bir boşluk olmadığı kabul edilir. Aşağıda bulunan kod herhangi bir duvar kullanılarak çalıştırılır ve öğrencilerle birlikte sonuçlar gözlemlenir.



Resim 132. Duvar Takip Programı

Bu kod bir blok hâline getirilmelidir. Döngüdeki kodların tamamı seçilir. Bu aşamada öğrenciler döngünün neden bloğa dâhil olmadığını merak edebilir. Döngüye daha sonra sola ve sağa dönme komutları da konulacağı için döngünün bloğun içerisine dâhil edilmediği öğrencilere açıklanır.

Araçlar (Tools) menüsünden My Block Builder alt menüsü seçilir. Açılan pencereden görevin ismi ve tanımı (yazılması zorunlu değildir) yazılır. Ardından görev bloğunun üzerinde görünecek olan görev resmi seçilir ve Bitir (Finish) butonuna tıklanarak blok oluşturulur. Oluşturulan bloğa istenildiğinde "Bloklarım (My Blocks)" sekmesinden ulaşılabilir.



Resim 133. Duvar Takip Bloğu

1.7. Uygula: Sola Dönme

Bu görevde robot (sağ duvarda bir açıklık yokken) karşısında bir duvar gördüğünde sola döner. Bunun için 3 numaralı porta bağlanan (ileriye bakan) mesafe sensörü kullanılır. Bu görev şu şekilde özetlenir:

Karşısında bir engel algıladığında:

i. Robot durur,
ii. Robot (dönüşte duvara çarpmamak için) bir miktar geri gelir,
iii. Robot sola döner.

Bu görevin kodu aşağıdaki gibidir. Bu kod çalıştırılarak öğrencilerle birlikte test edilir.

Dikkat: Bu görev, döngüde duvar takip programının sağına konulacaktır. Bu nedenle bir döngü içinde gösterilmemiştir. Kod döngüye yazılarak test edilmelidir.

Resim 134. Sola Dönme Programı

Göreve SolaDon ismi verilir ve Araçlar (Tools) menüsünden My Block Builder kullanılarak blok hâline getirilir.

1.8. Uygula: Sağa Dönme

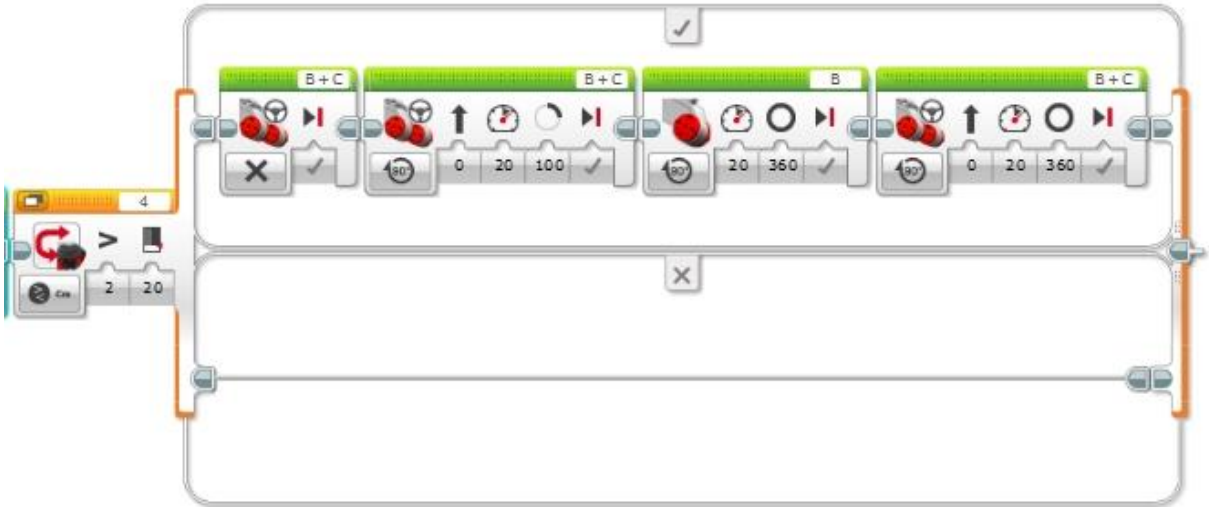
Bu görevde robot sağ duvarda bir boşluk algıladığında sağa döner. Bunun için sağa bakan ve 4 numaralı porta bağlı olan mesafe sensörü kullanılır. Bu görev şu şekilde özetlenir:

Sağ tarafta bir boşluk algıladığında:

(i) Robot durur,
(ii) Robot (dönerken duvara çarpmamak için) bir miktar ilerler,
(iii) Robot sağa döner,
(iv) Robot (yeni koridora girmesini sağlamak için) bir miktar ilerler.

Bu görevin kodu aşağıdaki gibidir. Rehber öğretmen bu kodu çalıştırarak öğrencilerle birlikte test eder.

Dikkat: Bu görev ana döngüye yerleştirilir. Bu nedenle bir döngü içinde gösterilmemiştir. Kod bir döngüye yazılarak test edilmelidir.

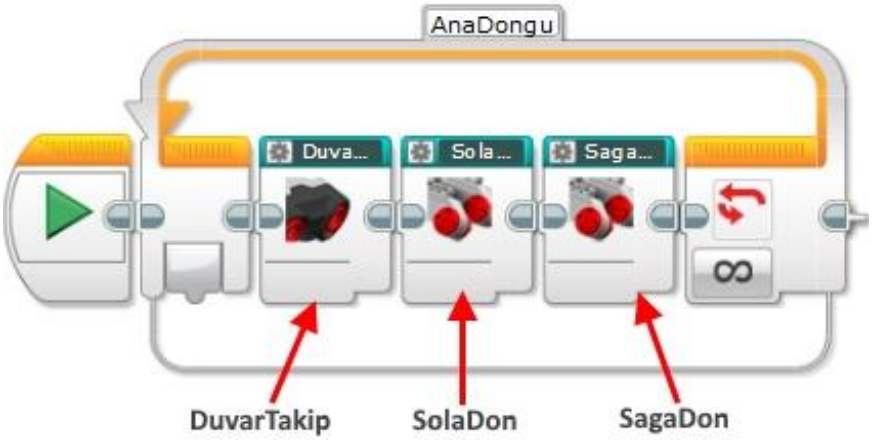


Resim 135. Sağa Dönme Programı

Göreve SagaDon ismi verilir ve Araçlar (Tools) menüsünden My Block Builder kullanılarak blok hâline getirilir.

1.9. Uygula: Görevlerin Birleştirilmesi

Problem üç alt probleme bölünerek "duvarı takip et", "sola dön" ve "sağa dön" olmak üzere üç farklı görev tanımlanmış; bu görevlerin kodu yazılarak blok hâline getirilmiştir. Oluşturulan alt çözümler, yani bloklar birleştirilince ana problem de çözülmüş olacaktır. Bu aşamada söz konusu problem için çözüm oldukça basittir ve kodu aşağıdaki şekildedir.



Resim 136. Blokların Birleştirilmesi

1.10. Gözle: Bluetooth

Öğrencilere *bluetooth* bağlantısı hakkında bilgilerinin olup olmadığı sorulur. Öğrencilerin bluetooth hakkındaki bilgi ve görüşleri dinlendikten sonra, öğrencilere *bluetooth* bağlantısı kullanarak robotlarının birbiriyle haberleşmelerini sağlayacakları belirtilir. Yapılacak etkinliklerde iki adet robot gerektiği için ikişerli gruplar (4 öğrenci) hâlinde çalışacakları ifade edilir.

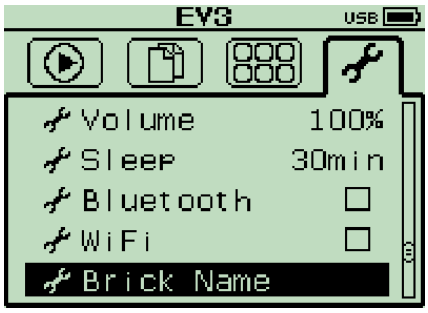
Rehber öğretmen, *bluetooth* teknolojisinin verilerin kısa mesafede kablosuz olarak aktarılmasını sağlayan bir teknoloji olduğunu belirtir. Açık alanda 10 metreye kadar çekim

alanına sahip olduğunu ifade eder. 10 metre, veri aktarımı ve haberleşme için oldukça kısa bir mesafe olsa da günümüzde *bluetooth* teknolojisinin kullanıldığı birçok alan bulunduğunu ifade eder ve öğrencilerden *bluetooth*un kullanım alanlarına örnekler vermelerini ister. Aşağıda birkaç örnek verilmiştir.

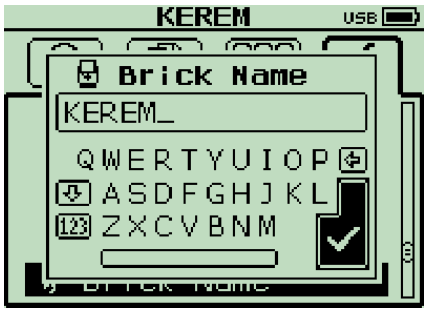
- Bluetooth kulaklık
- Bluetooth fare ve klavye
- Akıllı kol saatleri
- Bluetooth hoparlör
- Bluetooth otomobil sistemleri

Bluetooth ile cihazlar arasında haberleşmenin sağlanabilmesi için, öncelikle iki cihazın birbiriyle eşleştirilmesi gerekir. Eşleştirme öncesinde her iki cihazın *bluetooth* özellikleri aktif hâle getirilir. İki cihazın birbiriyle eşleştirilmesi esnasında girilen parola ile bağlantının güvenliği sağlanır.

Etkinliklerde robotların birbiriyle eşleştirilmesi esnasında herhangi bir sorun yaşanmaması için öncelikle tüm gruplardan robotlarına isim vermeleri istenir. Öğrenciler robotlarının isimlerini akıllı tuğla üzerindeki düğmeleri kullanarak ayarlar (settings) menüsündeki Tuğla İsmi (Brick Name) alt menüsünden değiştirebilirler. Ekranda açılan klavye üzerinde akıllı tuğla düğmeleriyle istenen karakterin üzerine gelinerek tuğlanın ismi değiştirilir.

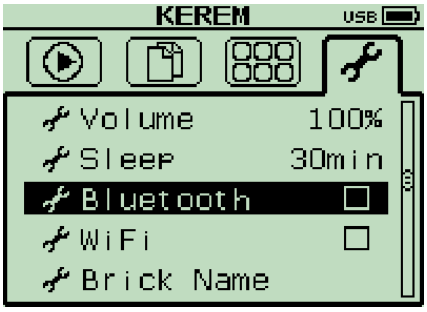


Resim 137. İsim Değiştirme Adım 1

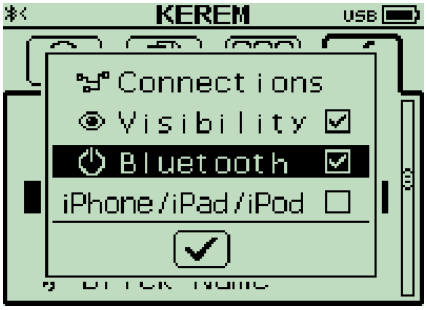


Resim 138. İsim Değiştirme Adım 2

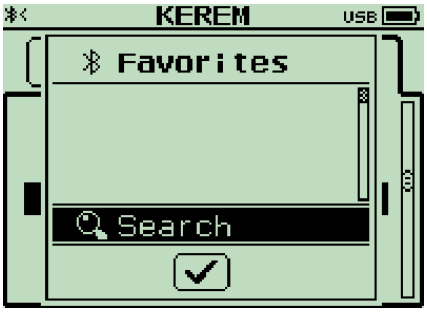
Robotların birbiriyle iletişimi geçebilmesi için *bluetooth* bağlantılarının aktif hâle getirilmesi ve birbiriyle eşleştirilmesi gerekmektedir. Robotları eşleştirmek için aşağıdaki adımlar takip edilir.



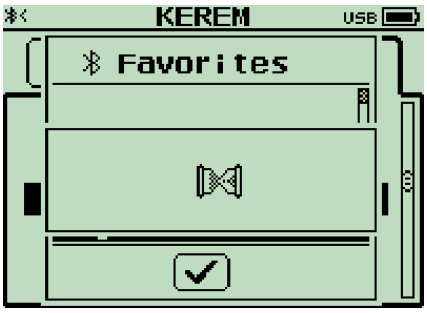
Ayarlar menüsünden Bluetooth seçeneğini seçilir.



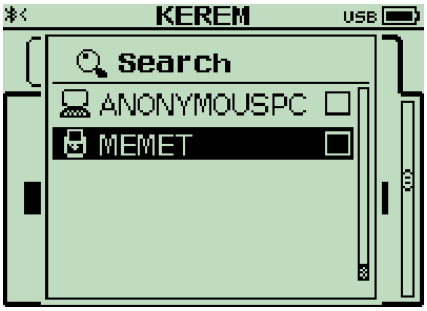
Açılan menüde Bluetooth seçeneği aktif hâle getirilir. Görülebilirlik (Visibilitiy) seçeneği de aktif hâle getirilir. Sonra Bağlantılar (Connections) seçeneğine gidilir.



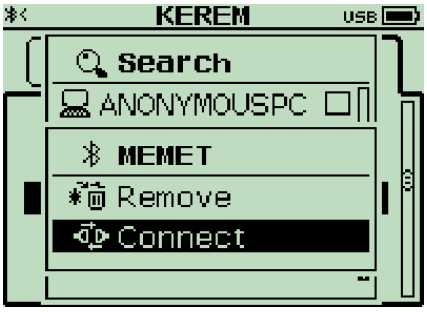
Ara (Search) seçilerek yakında bulunan diğer *bluetooth* cihazların listelenmesi beklenir.



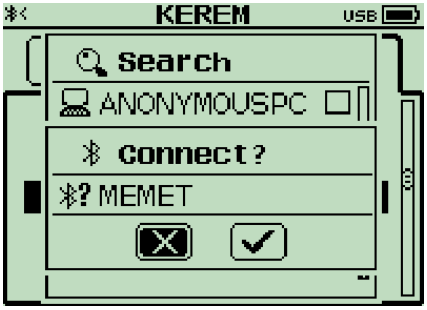
Arama süreci birkaç dakika sürebilir.



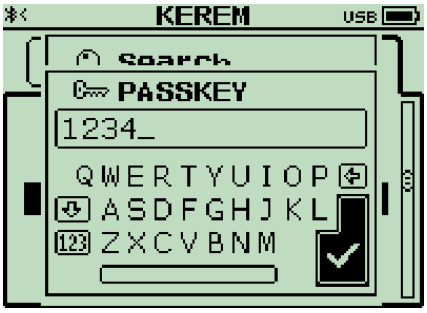
Bu süreçte robotun çevresinde bulunan cihazların isimleri sıralanır. Robotun eşleştirilmek istendiği cihaz, ismi üzerine gelinerek seçilir.



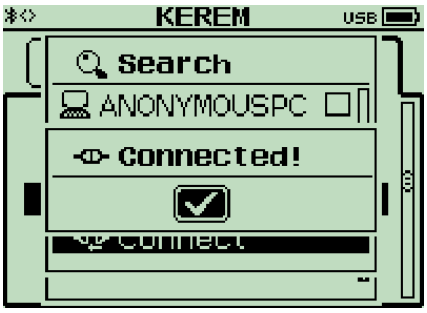
Seçilen cihaza bağlanmak için Bağlan (Connect) seçeneği seçilir.



Bağlanma isteği doğrulanır.



Akıllı tuğla üzerindeki düğmeler kullanılarak bir şifre belirlenir (1234 şifresi kullanılabilir) ve Giriş (Enter) tuşu işaretlenir. Eşleştirilmek istenen robotun ekranında da şifre onayı belirecektir, aynı şifre yeniden yazılarak Giriş (Enter) tuşu işaretlenir.

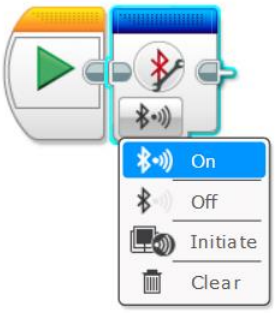


Ekranda Bağlandı (Connected!) yazısı ve bağlantı sesi duyulduğunda bağlantı işlemi tamamlanmış demektir.

Not

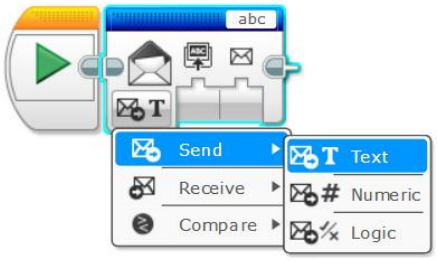
Aynı ortamda çok sayıda *bluetooth* cihazının bulunması, bağlanma süresini artırmakta ve bağlantı sorunlarına sebep olabilmektedir. Grupların, robotlarını ayrı bir ortamda eşleştirmeleri tavsiye edilir.

1.11. Gözle: Bluetooth Bağlantısı ve Mesajlaşma Blokları



Resim 139. Bluetooth Connection/Bağlantı Bloğu

Bluetooth Bağlantısı (Bluetooth Connection) bloğu EV3 yazılımında mavi İleri Düzey (Advanced) sekmesinde yer alır. Bluetooth Connection bloğunun dört modu bulunur. Bu blok ile akıllı tuğlanın *bluetooth*u açılabilir (On) veya açık olan *bluetooth* kapatılabilir (Off). Robot "Başlatma (Initiate)" moduyla belirtilen başka bir robot ile (bloğun sağ alt köşesinden, eşleştirilmek istenen robotun ismi yazılarak) eşleştirilebilir. "Temizle (Clear)" modu ise eşleştirilen robotla kurulan *bluetooth* bağlantısının sonlandırılması için kullanılır.



Resim 140. Bluetooth Messaging/Mesajlaşma Bloğu Seçenekleri

Öncesinde eşleştirilmiş iki akıllı tuğlanın birbiriyle haberleşmesini sağlamak üzere mavi sekmede yer alan Mesajlaşma (Messaging) bloğu kullanılır. Mesajlaşma bloğu kullanılarak mesaj gönderilebilir (Send), alınabilir (Receive) ve alınan mesaj var olan bir değer ile karşılaştırılabilir (Compare). Ayrıca gönderilen veya alınan mesaj Metin (Text), Sayısal (Numeric) ve Mantık (Logic) türlerinde olabilir.

Bir akıllı tuğladan diğerine mesaj gönderebilmek için üç parametrenin belirlenmiş olması gerekir. Bunlar:

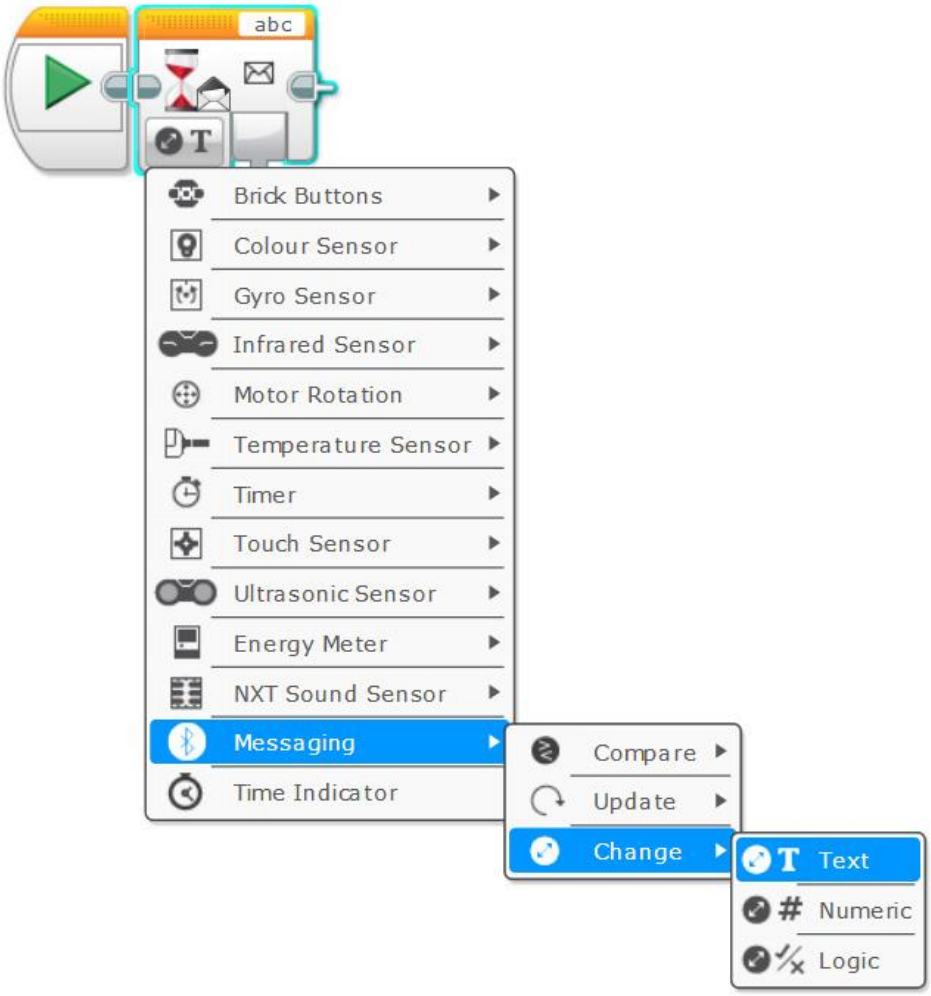
- **Mesaj başlığı:** Mesajlaşma bloğunun sağ üst köşesinde yer alır.
- **Mesajın iletileceği akıllı tuğlanın ismi:** Bluetooth bloğu kullanılarak eşleştirilen diğer akıllı tuğlanın ismi.
- **İletilecek mesajın içeriği:** Aktarılacak mesajın türüne bağlı olarak metin, sayı veya mantık (Doğru/Yanlış) değeri olabilir. Elle değer girilebileceği gibi kablo ile başka bir bloktan da aktarılmış olabilir.

Benzer şekilde, eşleştirilen tuğlanın gönderilen mesajı alması için yine Mesajlaşma (Messaging) bloğu kullanılır fakat Alma (Receive) modunun seçilmiş olması gerekmektedir.

Mesajın alınabilmesi için Gönder (Send) bloğunda belirtilen mesaj başlığı ve mesaj veri türünün (Text, Numeric, Logic) iki robotta da aynı olarak belirlenmesi gerekir.

Ayrıca, Mesajlaşma bloğu ile alınan mesajın içeriğinin var olan bir değer ile kıyaslanması da Karşılaştır (Compare) modu kullanılarak yapılabilir. Karşılaştırma işleminin sonucunda doğru veya yanlış olmak üzere Mantık (Logic) veri türünde bir sonuç elde edilir. Bu sonuç kablo ile bir diğer bloğa aktarılabilir. Özellikle Anahtar (Switch) bloğuna aktarılacak ise bu aşamada karşılaştır (compare) modunu kullanmak anlamlı olabilir.

Turuncu sekmede yer alan "bekle (wait)" bloğunun herhangi bir *bluetooth* mesaj gelinceye kadar programı bekletmek amacıyla kullanılabileceği de unutulmamalıdır.



Resim 141. Wait Bloğu Bluetooh İşlemleri

1.12. Uygula: Merhaba Mesajı

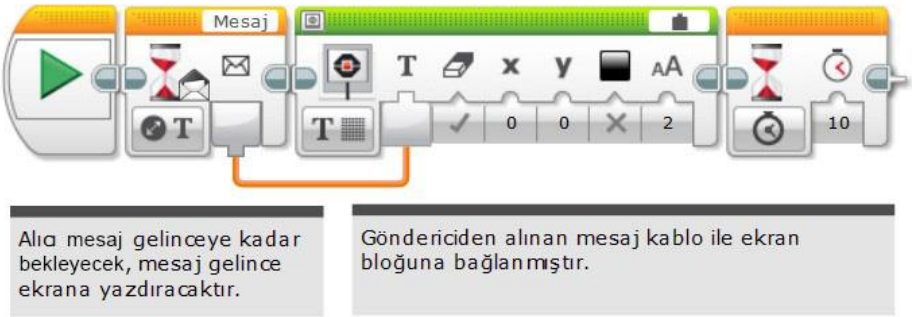
Öğrencilere iki grubun birleşerek birlikte çalışacakları belirtilir, böylece her yeni grupta iki adet akıllı tuğla olacaktır. Öğrencilerden her bir gruba ait iki robotu *bluetooth* ile birbirine bağlamaları istenir. Sonrasında bir robottan gönderilecek mesajın diğer robotun ekranında yazdırılmasını sağlayacak programları oluşturmaları istenir.

Öğrencilerin deneme yanılma yöntemini kullanmalarına imkân tanınır. Programlarında oluşacak sorunları tartışmaları ve çözüm önerileri üretmeleri desteklenir. Özellikle robotların eş zamanlı haberleşmelerini nasıl sağlayacakları konusunda çözümler üretmeleri istenir.

Bir robot mesajı göndereceği, diğer robot da alacağı için iki robotta farklı program çalıştırılır. Gün içinde öğrencilerin, görevleri dönüşümlü olarak gerçekleştirmeleri istenir.

Örnek gönderici ve alıcı programları aşağıdadır.

Resim 142. Örnek Gönderici Program

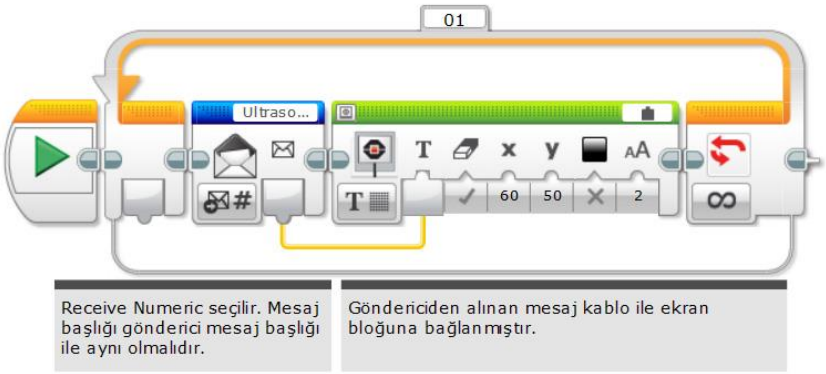


Resim 143. Örnek Alıcı Program

1.13. Uygula: Sensör Verilerinin Diğer Robota Aktarılması

Öğrencilerden, bir robotun algıladığı sensör verilerinin diğer robotun ekranında görünmesini sağlayan programı yapmaları istenir. Bu amaçla ultrasonik sensörün ölçtüğü mesafenin diğer robotun ekranında görünmesi istenir. Robot bir engele yaklaştıkça diğer robotun ekranındaki değerlerin de değişmesi gerekir. Yani ölçüm sonucu ekrana yalnızca bir defa yazdırılmamalı, sürekli olarak güncellenmelidir. Örnek program aşağıda verilmiştir.

Resim 144. Örnek Gönderici Program



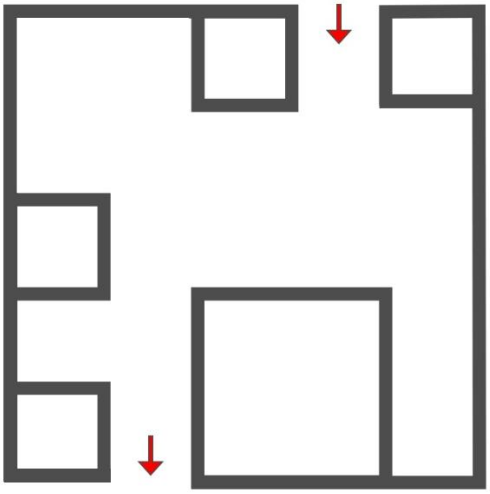
Resim 145. Örnek Alıcı Program

Öğrencilerden iki farklı sensöre ait verilerin, eşleştirilen robota nasıl aktarılabileceğini tartışmaları istenir. Yeterli vaktin olması durumunda hem mesafe sensörünün ölçtüğü uzaklık verilerin hem de renk sensörünün ölçütüğü yansıyan ışık verilerinin diğer robotun ekranına yazdırılması istenebilir (**İpucu**: Her bir sensör için farklı mesaj başlığı kullanılır).

2. TASARLA VE ÜRET

2.1. Labirent Yarışması

Derste kullanılacak labirent aşağıdadır. **Labirentin koridorlarının genişliği 30 cm'dir**. Rehber öğretmen labirenti oluşturulurken merkezlerde bulunan 20 cm x 30 cm boyutundaki mukavvalar ve rehber öğretmen tarafından üç boyutlu yazıcıda bastırılacak birleştirme aparatları kullanılacaktır. Birleştirme aparatlarının çizimi haftanın belgeleri içinde bulunmaktadır. Birleştirme aparatları kullanılarak mukavvalar birleştirilir ve labirent oluşturulur. **Rehber öğretmen dersten önce gerekli hazırlıkları yapmalıdır.**



Resim 146. Labirent

Yarışmanın kuralları:

- Yarışma için her öğrenci grubuna 45 dakikalık süre ve labirenti iki kere kullanma hakkı verilir.
- Öğrencilerin yaptıkları iki denemeden az süreli olanı grubun puanı olarak değerlendirilir.
- Öğrenciler ilk kullanımın ardından kodlarında veya tasarımlarında değişiklikler yapabilirler. Fakat labirentin ikinci defa kullanımı son haklarıdır ve bundan sonra değişiklik yapamazlar.
- En düşük puana sahip olan grup, yani labirenti en kısa sürede çözen grup yarışmayı kazanır.
- Robot labirenti çözerken duvara çarpmamalıdır. Robotu duvara çarpan grup labirent kullanma hakkını kaybeder. Eğer o grup ikinci labirent kullanma hakkında da robotu duvara çarptırırsa yarışmadan elenir.
- Gruplar denemeler yapmak için öğretmenin sağladığı labirent dışında sınıfta bulunan nesneleri kullanabilir. Örneğin sınıf duvarını kullanarak duvar takip kodunu test edebilirler.

2.2. Tasarla

Fiziksel olarak robotu oluşturma ve programlama adımlarını yazma sürecinden önce tanımlama ve fikir üretme sürecinin gerçekleştirilmesi önemlidir.

- Tanımlama Süreci: Öğrencilerin robotun ve programın neler gerektirdiğini belirlemesi/ortaya koyması gerekir. Öğrencilerin robot ve program için ayrı ayrı işlemleri maddeler hâlinde yazması beklenir.
- Fikir Üretme Süreci: Bu aşamada, öğrencilerin tanımlama sürecinde belirlenen işlemlerin nasıl yapılabileceği ile ilgili fikir yürütmesi gerekmektedir. Öğrencilerin robotun oluşturulması ve program kodlarının yazılması sürecinde çözüm için sundukları fikirleri maddeler hâlinde belirtmeleri önemlidir.

2.3. Üret

Öğrencilerin önce, tanımladıkları ve hakkında fikir ürettikleri robotu oluşturmaları gerekir. Robotun istenen şekilde ve sorunsuz hareket edip etmediği test edilmeli, sorunlar varsa gözden geçirilip düzeltilmelidir. Daha sonra program tamamlanmalı ve kontrol edilip eksiklikleri giderilmelidir.

Dikkat: Örnek çözüme rehber öğretmenlere verilen dosyalardan ulaşılabilir. Verilen çözüm, problemi çözmek için tek yol değildir. Öğrenciler birçok şekilde ilerleyerek kendi kodlarını geliştirebilir.

2.4 Labirent Çözümünün İyileştirilmesi

Yazılan kodların iyi çalışıp çalışmadığını öğrenciler ve rehber öğretmen gözlemlemelidir. Robot bazen labirentten çıksa da belirli durumlarda labirentten çıkamamaktadır ya da bazı görevleri yerine getirirken bazılarını yerine getirememektedir. Bunun nedeni alt problemler için geliştirilen çözümlerin tam olarak yeterli olmamasıdır. Bu çözümlerden bazıları iyileştirilebilir. Öğrenciler iki şekilde iyileştirme yapabilirler:

(i) Yazılan kodlardan bazıları yeterli hassasiyete sahip değildir. Bu kodlar geliştirilebilir.
(ii) Robot fiziksel olarak fazla yer kaplamaktadır. Robot başka bir tasarımla küçültülebilir.

İpuçları (Bunları öğrencilere başta söylemek uygun değildir, öncelikle kendilerinin keşfetmesine veya sorarak öğrenmelerine izin/ mkân verilmelidir):

- **İpucu 1**: Robot duvara çok yaklaştığında ve duvardan çok uzaklaştığında ani tepki veremediği için rotasından fazlaca sapar.
- **İpucu 2**: Robot ile duvar arasındaki mesafe 3 cm'nin altına düştüğünde mesafe sensöründen gelen değer birden 255'e çıkar. Bu durum, mesafe sensörünün yakın değerler için yeterince iyi olmadığını gösterir. Fakat bu durum dikkate alınarak yazılan kod iyileştirilebilir.
- **İpucu 3**: Robot tasarımında akıllı tuğla yatay durumda kullanılır. Akıllı tuğla dikey hâle getirilerek robotun daha az yer kaplaması sağlanabilir..

2.5. Tasarla: Taklitçi Robot

Bu etkinlikte öğrencilerden bir robot bağımsız olarak hareket ederken diğer robotun aynı hareketi gerçekleştirmesini sağlayacak programı hazırlamaları istenir (örneğin, önceki haftalarda gerçekleştirilen, öndeki aracı takip eden veya çizgiyi takip eden robot etkinlikleri).

Tasarla: Öğrencilerden gün içinde öğrendikleri "*bluetooth* bağlantısı ile robotların haberleşmeleri" yöntemini kullanarak, bir robotun yaptığı hareketin aynısını yapan taklitçi bir robot tasarlamaları ve programlamaları istenir. Birinci robotun tamamen bağımsız olduğu, taklitçi robotun ise birinci robota bağlı olduğu belirtilir. Birinci robotun hareketi önceden belirlenmeyecektir, robot ortama uygun herhangi bir hareketi yapacaktır (birinci robot elle de

hareket ettirilebilir). Sadece ileri geri değil, sağ sola dönme hareketleri de taklit edilmelidir (Örneğin, çizgi izleyen robot etkinliğindeki robotun hareketi).

Tanımlama: Öğrenciler öncelikli olarak problemi tanımlamalıdırlar. Problemi çözmelerine yardımcı olacak şu soruların cevaplarını kendi aralarında tartışırlar:

- Taklitçi robotun taklit etmesi istenen nedir?
- Taklitçi robot, birinci robottan hangi verileri almalıdır?
- *Bluetooth* üzerinden kaç farklı veri türü paylaşılacaktır?
- Sağa veya sola dönme hareketleri nasıl taklit edilebilir?

Fikir Üretme: Bu aşamada, öğrencilerin yukarıda belirlenen işlemleri robotun nasıl gerçekleştirebileceği ile ilgili fikir yürütmesi gerekir. Öğrenciler aşağıda belirtilen çözüm önerilerine benzer fikirler üretebilirler.

- *Bluetooth* üzerinden hareketi sağlayan B ve C motorlarının verileri paylaşılmalıdır.
- Sağa veya sola dönüşler için B ve C motorlarının verilerinin bağımsız olarak paylaşılması gerekir.
- Hareketin eş zamanlı olması için döngü kullanılması gerekir.
- Taklitçi robotun B ve C motor açıları sürekli olarak, birinci robotun B ve C motorunu açısına eşitlenmelidir.

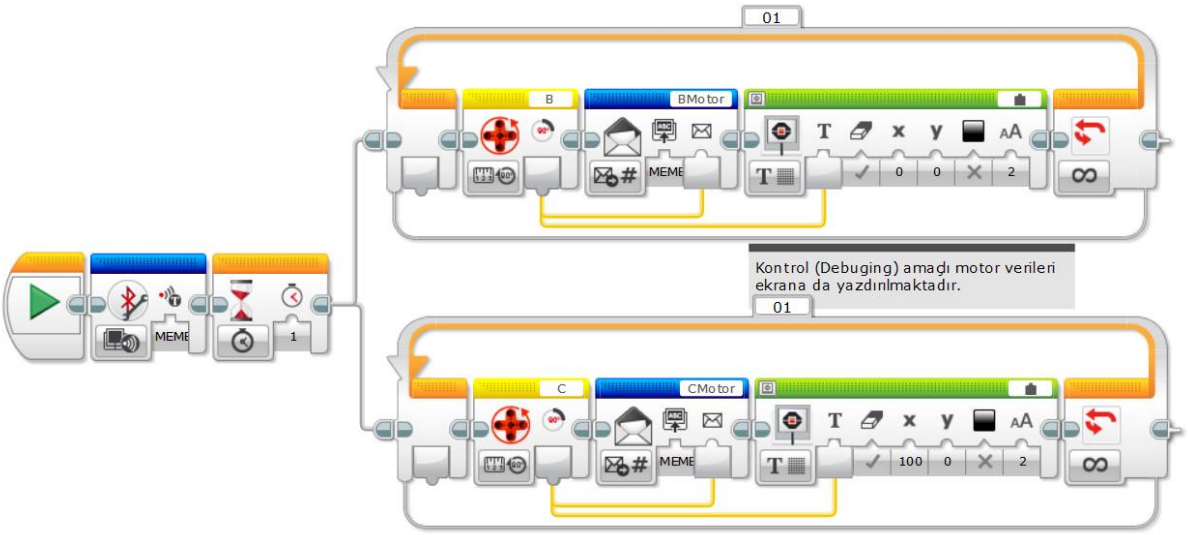
B ve C motorlarının başka hangi verileri paylaşılabilir?

Not

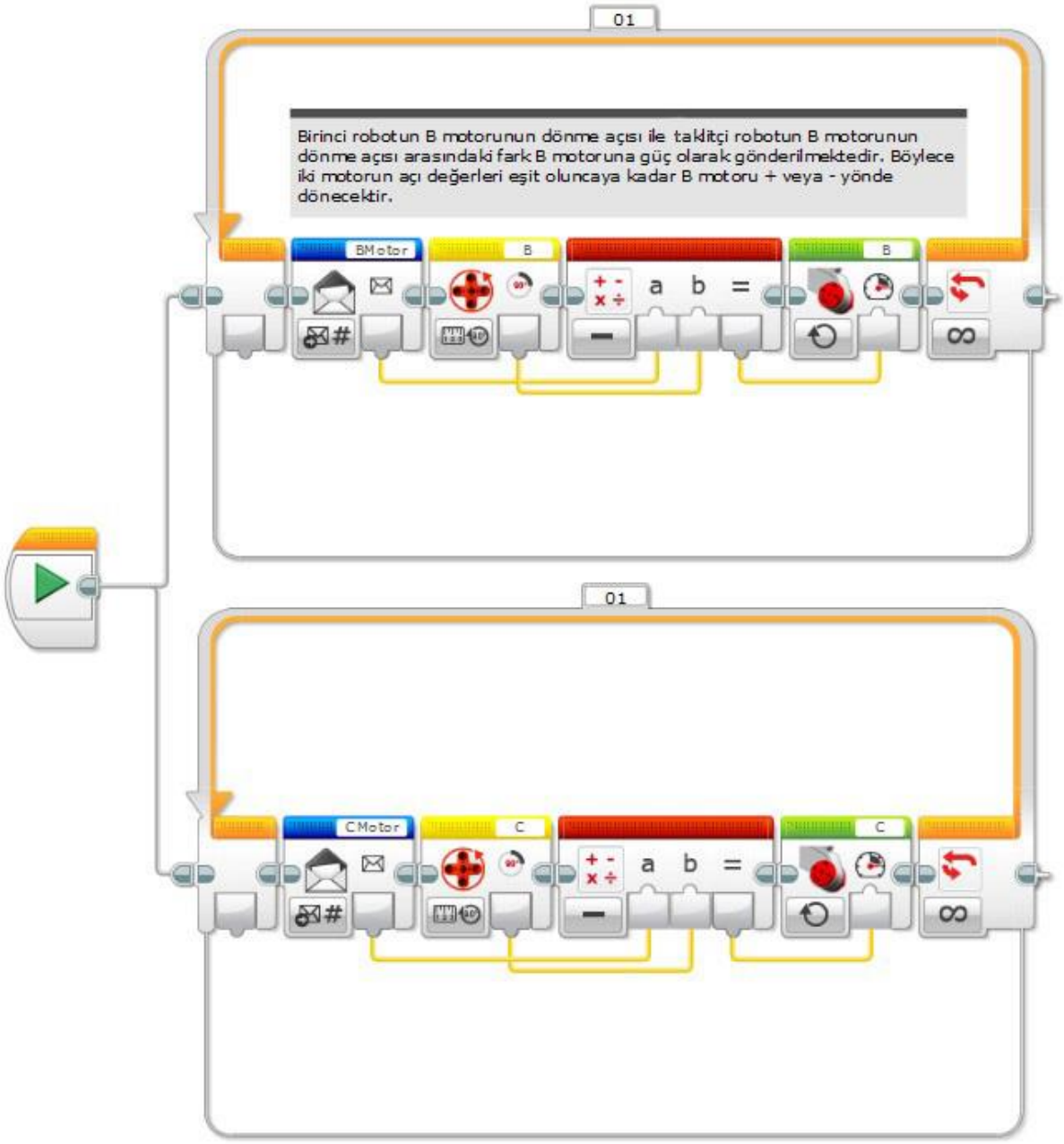
Öğrenciler bu süreçte, sadece B ve C motor verilerinin iletilmesinin haricinde, mesafe, renk veya dönme sensörü verilerinin iletilmesi ile taklitçi robotun hareketini belirleme yolunu seçebilirler. Problemin çözümünde tek yöntemin olmadığı unutulmamalıdır. Mantıklı çözüm önerileri desteklenmelidir.

2.6. Üret: Taklitçi Robot

Öğrenciler çözüm tasarımlarını yaptıktan sonra bilgisayar ve robot başında çalışarak istenen görevi yerine getirirler. Birinci ve taklitçi robot için örnek programlar aşağıda sunulmuştur.



Resim 147. Birinci Robot Örnek Program



Resim 148. Taklitçi Robot Örnek Program

Not

Benzer program, birinci robotun B ve C motorlarından ölçülen Mevcut Güç (Current Power) değerinin, taklitçi robotun ilgili motora doğrudan aktarılmasıyla da gerçekleştirilebilir.

3. DEĞERLENDİR

Bu aşamada hedef, öğrencilerin öğrenme sürecinde yaşadıkları ve öğrendikleri üzerine düşünmelerini sağlamaktır. Bu sayede öğrenciler süreç içinde deneyimlediklerini ve öğrendiklerini gözden geçirerek kendilerini değerlendirme ve tanıma fırsatı elde edeceklerdir. Öğrencilerden şu soruları yanıtlamaları istenebilir:

- Verilen problemleri tanımlayınız (problemi kendi cümleleri ile ifade etme).
- En çok hangi görevde zorlandınız? Bu zorlukların üstesinden nasıl geldiniz? (Problemin çözümü için hangi stratejileri kullandınız ve neden bu stratejileri seçtiniz?) Öğrenciler aşağıdaki problemleri bildirmezlerse rehber öğretmen bu problemleri nasıl çözdüklerini sorabilir.
    - Duvar takip etme görevinde zorluklar yaşadınız mı?
    - Sola ve sağa dönme görevinde zorluklar yaşadınız mı?
    - Birleştirme görevinde zorluklar yaşadınız mı?
    - *Bluetooth* görevlerinde karşılaştığınız problemler nelerdir?
- Problemleri çözerken ne gibi sıkıntılar yaşadınız ve bunların üstesinden gelmek için neler yaptınız?
- Kullandığınız yöntemler, bu sıkıntıları gidermekte başarılı oldu mu?
- Grup arkadaşınızla anlaşmazlığa düştüğünüz durumlar oldu mu ve bunların üstesinden gelmek için neler yaptınız?
- Grup arkadaşınızdan neler öğrendiniz?
- Bu hafta öğrendiğiniz konuları günlük hayatta nerelerde uygulayabilirsiniz?

Değerlendirme, öğrencileri sıkmadan, her soru için verilen cevaplar tatmin edici bir düzeye ulaşıncaya kadar devam ettirilir.

Proje Hazırlıyorum

Bu hafta itibarıyla proje çalışmalarına başlanması önerilmektedir. Projenin “hazırlık” süreci için planlamalar yapılmalıdır. Detaylar için EK’ler incelenebilir.

9. Hafta: MicroPython ile Hareket, Ses, Ekran Görüntüsü

Ön Bilgi:

- Öğrenciler robot kavramını bilir.
- Öğrenciler robot setiyle farklı robot tasarımları yapmıştır.
- Öğrenciler robot setini programlamak için grafik arayüzünü kullanmıştır.
- Öğrenciler EV3 yazılımında, robotun hareket etmesi için gerekli programlama adımlarını oluşturmuştur.
- Öğrenciler EV3 yazılımında, sensörlerin kullanıldığı programlar yapmıştır.
- Öğrenciler EV3 yazılımında, veriler üzerinde matematik ve mantık işlemlerini gerçekleştirmiştir.

Haftanın Kazanımları:

- VS Code editörünü kullanarak MicroPython kodları ile program yazar.
- MicroPython ile hareket, ses, ekran görüntüsü kodlarını kullanarak program yazar.
- MicroPython ile döngüleri kullanır.
- MicroPython'da hareket, ses, ekran görüntüsü ve döngüleri birlikte kullanarak problemleri çözer.

Haftanın Amacı:

Bu haftanın amacı öğrencilere MicroPython ile hareket, ses, ekran görüntüsü ve döngü kodlarını öğretmektir. Sonraki süreçte, öğrencilerin çeşitli problemlerin çözümünde bu kodları kullanmasını sağlamaktır.

Kullanılacak Malzemeler:

Robot seti, bilgisayar, gerekli yazılımlar, mat (çalışma alanı)

Haftanın İşlenişi:

Gözle: MicroPython ile hareket, ses, ekran görüntüsü ve döngü kodlarını inceleme

Uygula: MicroPython ile hareket, ses, ekran görüntüsü ve döngü kodlarını çeşitli örnekler üzerinde uygulama

Tasarla: MicroPython ile hareket, ses, ekran görüntüsü ve döngü kodlarınının bir problemin çözümünde kullanılabilmesi için tasarımını yapma

Üret: MicroPython ile hareket, ses, ekran görüntüsü ve döngü kodlarının bir problemin çözümü için oluşturulan tasarımını ürün hâline getirme

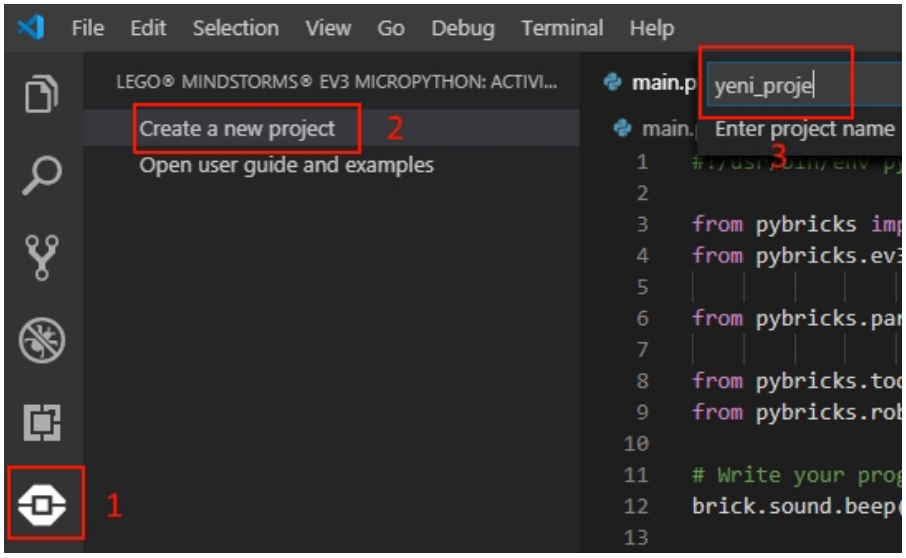
Değerlendir: Konu değerlendirme etkinliği

Proje: Projeye başlama etkinliği

1. GÖZLE VE UYGULA

1.1. Gözle: İlk Programımı Çalıştırıyorum

Bu etkinlikteki amaç MicroPython ile yazılan ilk programı çalıştırmaktır. VS Code editöründe yeni bir proje açmak için öncelikle LEGO sembolüne tıklanır. Ardından açılan pencereden "Create a new project" üzerine tıklanır. Üçüncü adımda, açılan metin kutusuna projenin ismi yazılır ve ENTER tuşuna tıklanarak proje bilgisayarda istenilen yere kaydedilir. Bu adımlar aşağıda gösterilmiştir.

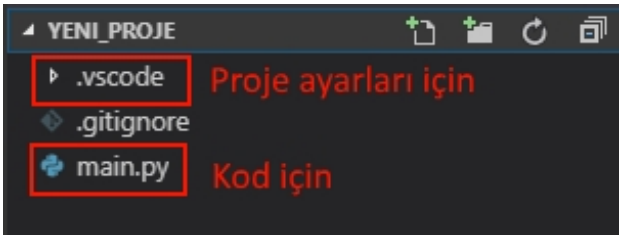


Resim 149. Proje Oluşturma Adımları

Not

MicroPython ile EV3 Kullanımı isimli dokümanda, kurulum için gerekli yazılımların yönergesi bulunur.

VS Code editöründe YENİ_PROJE isminde bir sekme açılır. Bu sekme içerisindeki linklerden kod yazılacak ana dosyaya (main.py) ve proje ile ilgili ayarların bulunduğu dosyalara (.vscode) ulaşılabilir. Aşağıdaki görselde bu iki link gösterilmiştir.



Resim 150. Proje Klasör ve Dosyaları

Kod yazılacak dosya olan main.py'ye ulaşmak için link tıklanır. Aşağıdaki görselde görüldüğü üzere kod yazılacak kısma VS Code editörü otomatik olarak bazı kodları ekler. Bunlardan "from" ile başlayanlarının görevi, EV3 robotunu programlamak için kullanılacak kütüphane dosyalarını yüklemektir. Bu kodların altına ise öğrenciler kendi kodlarını yazabilir. VS Code

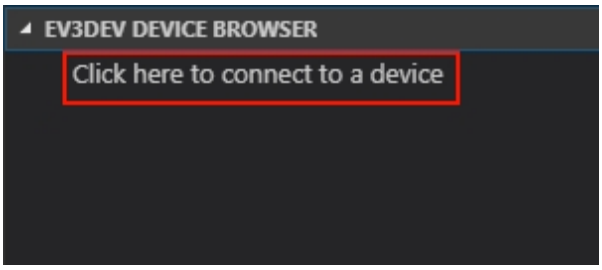
bu kısma otomatik olarak brick.sound.beep() komutunu ekler. Bu kodun görevi "bip" sesi çıkarmaktır. Hemen bu kodun üzerinde "# Write your program here" ifadesi bulunur. Başında # olan ifadeler komut olarak kabul edilmez ve çalıştırılmaz. Programcılar # sembolünü kullanarak kendileri için notlar alır, yani bunlar yorum satırlarıdır.

```
main.py
#!/usr/bin/env pybricks-micropython
                EV3 robotuna komut verebilmek için gerekli kütüphaneler
from pybricks import ev3brick as brick
from pybricks.ev3devices import (Motor, TouchSensor, ColorSensor,
                                 InfraredSensor, UltrasonicSensor, GyroSensor)
from pybricks.parameters import (Port, Stop, Direction, Button, Color,
                                 SoundFile, ImageFile, Align)
from pybricks.tools import print, wait, StopWatch
from pybricks.robotics import DriveBase

# Write your program here
brick.sound.beep()          Kod yazılacak kısım
```

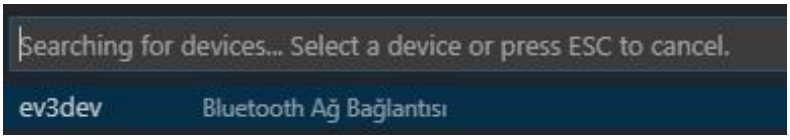
Resim 151. Örnek Program

Burada yazılan kod çalıştırıldığında robottan "bip" sesi duyulur. Fakat bu kodu çalıştırmak için öncelikle EV3 robotuna bağlanmak gerekir (**Dikkat**: Bu işlem öncesinde EV3 robotu bilgisayara USB kablo veya Bluetooth aracılığı ile bağlanmış olmalıdır). Aşağıdaki görselde gösterildiği gibi VS Code editörünün sol alt köşesinde bulunan EV3DEV DEVICE BROWSER sekmesinde bulunan "Click here to connect to a device" üzerine tıklanır.



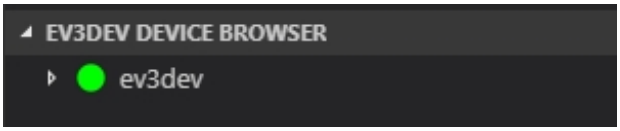
Resim 152. Bağlantı Adım 1

Açılan pencereden bağlanılmak istenilen cihaz seçilir ve bağlantı yapılır. Bu işlem aşağıda gösterilmiştir.



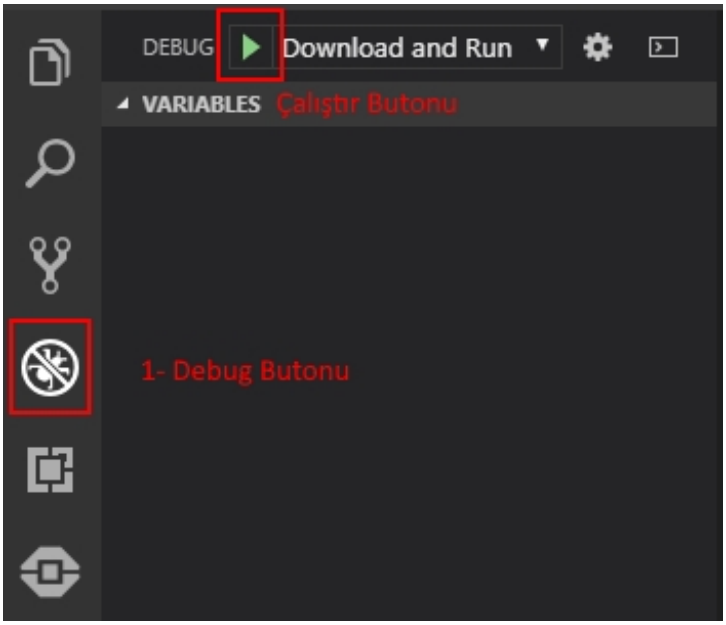
Resim 153. Bağlantı Adım 2

Bağlantı gerçekleştikten sonra EV3DEV DEVICE BROWSER sekmesinde robotun ismi görünür ve önünde bulunan daire şeklindeki işaret yeşil renge döner. Eğer bu dairenin rengi yeşil değilse bağlantı gerçekleşmemiş demektir. Bağlantısı gerçekleştirilmiş bir robot için EV3DEV DEVICE BROWSER sekmesi aşağıdadır.



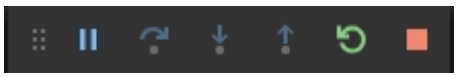
Resim 154. Bağlantı Adım 3

Program artık çalıştırılabilir. Bunun için önce VS Code editöründeki DEBUG butonu, sonra çalıştır butonuna tıklanır. Bu butona tıklanınca program robota indirilir ve çalıştırılır. Aşağıdaki görselde çalıştırma işleminin nasıl gerçekleştirileceği gösterilmektedir. Buna alternatif olarak programı çalıştırmak için F5 tuşuna basılabilir. F5 tuşuna basılır basılmaz program robota indirilip çalıştırılır.



Resim 155. Programın Yüklenip Çalıştırılması

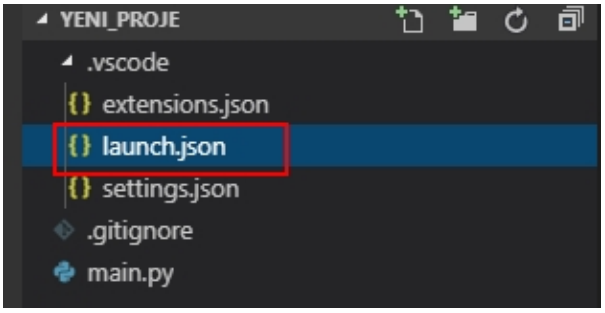
Program çalıştırılınca bilgisayar ekranında aşağıdaki pencere görünür. Bu pencereden çalıştırılan program duraklatılabilir (mavi renkli), yeniden çalıştırılabilir (yeşil renkli) ve sonlandırılabilir (kırmızı renkli).



Resim 156. Çalıştırma Seçenekleri

1.2. Gözle: Programı Robottan Çalıştırmak

Yukarıda bahsedildiği şekilde programı çalıştırmak her zaman uygun olmayabilir. Bazen program çalışır çalışmaz robot hızlı hareketler yapıp bulunduğu platformdan düşebilir. Bunu önlemek için program önce robota yüklenir ardından robot üzerindeki menü kullanılarak robot çalıştırılır. Bunun için proje sekmesinin altındaki .vscode linkinden ulaşılabilen launch.json dosyası açılır.



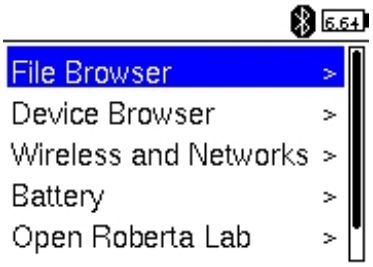
Resim 157. Ayar Dosyası Bağlantısı

Bu dosyada program ile ilgili ayarlar yapılabilir. Aşağıdaki görselde gösterildiği gibi bu dosyada "program" ile başlayan satırın önüne "//" yazılır.

```
"version": "0.2.0",
"configurations": [
    {
        "name": "Download and Run",
        "type": "ev3devBrowser",
        "request": "launch",
        //"program": "/home/robot/${workspaceRootFolderName}/main.py"
    }
]
```

Resim 158. Ayar Dosyası Değişikliği

Bunun ardından kodun yazılı olduğu main.py dosyasına geçilir ve F5 tuşuna basılır. Böylece program robota indirilmiş fakat çalıştırılmamış olur (**Dikkat**: VS Code indirme esnasında "Failed to run file" hatası verir. Bu hata göz ardı edilmelidir). Programı çalıştırmak için robotun File Browser menüsü seçilir.



Resim 159. Robottan Programın Çalıştırılması

Açılan menüden projenin ismi seçilir ve içerisindeki main.py dosyası seçilerek çalıştırılır.

1.3. Gözle: İleri Hareket

Bu kısımdaki amaç robotu ileri doğru hareket ettirmektir (**Dikkat**: Burada robotun Driving Base olduğu varsayılır). Kodu yazmak için main.py dosyası açılmalıdır. Robotu hareket ettirmek için öncelikle robotun sağ ve sol motorlarının tanıtılması gerekir. Bunun için aşağıdaki komutlar yazılır.

```
motorB=Motor(Port.B)
motorC=Motor(Port.C)
```

Not

Temel tasarım (Driving Base) yönergesine aşağıdaki yollarla da ulaşılabilir:
i) https://le-www-live-s.legocdn.com/sc/media/lessons/mindstorms-ev3/building-instructions/ev3-rem-driving-base-79bebfc16bd491186ea9c9069842155e.pdf
ii) Ev3 yazılımı > Lobby > Building Instructions > Building Ideas > Driving Base
iii) Robot setiyle birlikte gelen kitapçığa bakılır.

B portuna (Port.B) takılı motora motorB ismi verilir. Benzer şekilde C portuna takılmış motora ise motorC ismi verilir. Bu isimler zorunlu değildir. İsimlendirme kuralları dâhilinde istenilen isimler verilebilir. Motorlar Temel Tasarım (Driving Base)'deki gibi takılmıştır. Bu durumda Driving Base tanıtılmalıdır. Bunun için aşağıdaki komut yazılır.

```
robot=DriveBase(motorB,motorC,56,114)
```

Bu komut ile robotun sol motorunun motorB (ilk parametre), sağ motorunun motorC (ikinci parametre), tekerleğinin çapının 56 mm (üçüncü parametre) ve iki tekerleğin aksları arasındaki mesafenin 114 mm (dördüncü parametre) olduğu söylenir. Bu parametreler girilmeden robota ileri geri komutları verilemez. Aşağıdaki komut yazılarak robot ileri doğru hareket ettirilebilir.

```
robot.drive_time(360,0,1000)
```

Bu komut ile robotun tekerleklerinin 360 mm/saniye hız ile hareket edeceği (ilk parametre), ileriye doğru gideceği (ikinci parametre) ve 1 saniye boyunca ilerleyeceği (üçüncü parametre) söylenmiş olur. İlk parametrenin alabileceği maksimum hız değeri 960 derece/saniye ile 1020 derece/saniye arasındadır. Bu değerler large motorların maksimum hız değerine eşittir. İkinci parametre robotun yönünü tayin eder. Pozitif değerler sağa doğru hareket, negatif değerler sola doğru hareket ve sıfır ise doğrudan ileriye doğru hareket için kullanılır. Üçüncü parametre milisaniye cinsinden yazılır. 1000 milisaniye 1 saniye yapar. Robotun 3 saniye hareket etmesi isteniyorsa 3000 değeri girilmelidir.

Program tamamlanmıştır. F5 tuşunu kullanarak program çalıştırılır. Program çalıştırılırken VS Code çeşitli hatalar verebilir. Bu hataların nedeni kodların tam olarak doğru yazılmaması olabilir. Metin tabanlı programlama dillerinde yazılan kodlar tam olarak doğru yazılmalıdır, eksik olan bir parantez, harf, nokta veya alt çizgi programın çalışmamasına yol açar. Bu tip hatalara sentaks (syntax) hatası denir. Eğer programda bu tip hatalar varsa bunlar bulunup düzeltilmelidir.

```
----------
Traceback (most recent call last):
  File "/home/robot/yeni_proje/main.py", line 11, in <module> # hatanın kaynağı
NameError: name 'PortB' is not defined # Hatanın çeşidi
----------
Exited with error code 1.
```

Yukarıda sentaks hatasına bir örnek verilmiştir (**Dikkat**: Kırmızı renkli yazılar yazar tarafından eklenmiştir. Konu anlatımındaki örneklerde hata yoktur, hatalı kod örnek olarak verilmiştir).

Burada hatanın bir isim hatası olduğu ve PortB diye bir şeyin tanımlanmadığı söylenmektedir (hatanın çeşidi). Hatanın bulunduğu yer ise bir üst satırda söylenmiştir (hatanın kaynağı). Hata 11. satırdadır (Öğrencilerin oluşturduğu programda farklı bir satırda olabilir). Bazen hata söylenilen satırın bir üstünde veya bir altında da olabilir. Bu satırlar da kontrol edilmelidir. 11. satıra gidildiğinde aşağıdaki kod görülür:

```
motorB=Motor(PortB)
```

Burada Port.B yazılması gerektiği hâlde PortB yazıldığı görülür. Bu hata düzeltildiğinde program çalışır.

Yukarıda verilen örnekte robot ileriye doğru 360 mm/saniye hızında bir saniye ilerler ve kendi motor kuvveti ve sürtünme kuvveti ile durur. Fren yapılmamıştır. Fren yapmak için aşağıdaki komut kullanılabilir:

```
robot.stop(Stop.BRAKE)
```

Burada parametre olarak Stop.BRAKE verilir. Bu, fren yap anlamındadır. Bunun yerine Stop.HOLD veya Stop.COAST yazılabilir. Stop.HOLD ani fren yapmak için kullanılır. Stop.COAST ise robotun kendiliğinden durmasını sağlar. Durma komutlarının her biri öğrencilere denetilmelidir.

1.4. Gözle: Geri Hareket

Robotun geri gitmesi için yapılması gereken işlem, negatif değerli hız parametresi girmektir. Örneğin aşağıdaki şekilde bir kod yazılırsa robot geriye doğru 360 mm/saniye hızında bir saniye boyunca gider.

```
robot.drive_time(-360,0,1000)
```

Robot önce bir saniye geriye doğru, ardından 0,5 saniye bekledikten sonra yeniden bir saniye geriye doğru hareket ettirilmek istenmektedir. Bu işlemin gerçekleştirilebilmesi için “wait” komutunun bilinmesi gerekir. “wait(500)” komutu ile 500 mili saniye (yani 0,5 saniye) robotun bir sonraki görevi gerçekleştirmeden önce beklemesi sağlanır. Burada robotun 1 saniye beklemesi istenirse komuta parametre olarak 1000 verilmelidir.

```
robot.drive_time(-360,0,1000)
wait(500) # 1 saniye için wait(1000) yazılmalı
robot.drive_time(-360,0,1000)
robot.stop(Stop.BRAKE)
```

1.5. Uygula: İleri – Geri Hareket

Bu etkinlikteki amaç robotun bir saniye ileri doğru hareket etmesini ve ardından bir saniye geri hareket ederek başlangıç noktasına dönmesini sağlamaktır. Bunun için öğrencilere aşağıdaki kod verilir (**Dikkat**: Öğrenciler bu kodu çalıştırmadan önce, robotun başlangıç yerini mat üzerindeki cetvelde belirlemelidir).

```
motorB=Motor(Port.B)
motorC=Motor(Port.C)
robot=DriveBase(motorB,motorC,56,114)
robot.drive_time(360,0,1000)
robot.drive_time(-360,0,1000)
```

Robot tam olarak başlangıç noktasına dönmeyebilir. Bütün öğrenciler bu kodu denedikten sonra robotun neden başlangıç noktasına dönmediği sınıfça tartışılmalıdır. Bu tartışmanın ardından öğrencilerden robotun başlangıç noktasına dönmesi için gerekli kodu yazmaları istenir.

Robotun başlangıç noktasına dönmesi için ileri ve geri komutlarından sonra fren yapılmalıdır. Eğer fren yapılmazsa robot kendi hızıyla durmaya çalışırken geri komutunu işletmeye çalışır. Bu işlem de ileri ve geri gidilen mesafelerin farklı olmasına neden olur. Fakat tek başına fren yapmak bu hatanın giderilmesi için yeterli değildir çünkü belirli bir fren süresi bulunur. Bu fren süresi boyunca verilen geri komutu robotun tam olarak istenilen miktarda geri gitmesini sağlayamaz. Bu hatanın giderilebilmesi için ayrıca ileri ve geri komutları arasında bir miktar bekleme süresi bırakılmalıdır. Bu iş için kullanılması gereken örnek kod aşağıdadır.

```
motorB=Motor(Port.B)
motorC=Motor(Port.C)
robot=DriveBase(motorB,motorC,56,114)
robot.drive_time(360,0,1000)
robot.stop(Stop.BRAKE)
wait(100)
robot.drive_time(-360,0,1000)
robot.stop(Stop.BRAKE)
```

1.6. Gözle: Robotu 90 Derece Döndüren Program

Bu etkinlikte robotu sağ tarafa doğru 90 derece döndüren program yazılacaktır. Bu görev için sol motor çalıştırılıp sağ motor sabit tutulur. MicroPython'da bir motoru hareket ettirmenin birden fazla yolu vardır.

1.6.1. Run Metodu

"run(*hız*)" metodu ile motorun sabit bir hızda ilerlemesi sağlanır. Buraya girilecek hız parametresi derece/saniye cinsidendir. "Run" metodu ile ilerleyen motorun çalışması bu motora "stop()" komutu girilinceye, motora yeni bir komut girilinceye veya program bitinceye kadar devam edecektir. Aşağıdaki komutlar bu iş için kullanılabilir (**Dikkat**: Buradaki hız ve bekleme süresi değerlerini öğrencilerin kendi durumlarına göre değiştirmesi gerekebilir).

```
motorB=Motor(Port.B) #Sol motor tanımlandı
motorB.run(500) # Sol motor 500 derece/saniye hızıyla çalışır
wait(800) # run komutu 800 mili saniye çalışması için
motorB.stop(Stop.BRAKE) # Sol motorun durması için
```

Yukarıda wait(800) satırı yazılmazsa motor dönmeye başlayacağı sırada program biteceği için motor hiç dönmeyecektir.

1.6.2. Run_time Metodu

"run_time" (*hız, süre, durma şekli, bekletme*) metodu ile motorun istenilen hızda ve sürede hareket etmesi sağlanabilir. Durma şekli için Stop.COAST, Stop.BRAKE ve Stop.HOLD kullanılabilir. Bekletme parametresi/seçeneği "True" olarak ayarlanırsa programın geri kalan kısmı motorun manevrasını bitirmesini bekler. Verilen görevi yerine getirmek için aşağıdaki komutlar kullanılabilir (**Dikkat**: Buradaki hız ve bekleme süresi değerlerini öğrencilerin kendi durumlarına göre değiştirmeleri gerekebilir).

```
motorB=Motor(Port.B) # Sol motor tanımlandı
motorB.run_time(500,1050, Stop.BRAKE, True)
```

1.6.3. Run_angle Metodu

"run_angle" (*hız, dönme açısı, durma şekli, bekletme*) metodu "run_time" metoduna çok benzer. İkisinin arasındaki tek fark "run_angle" metodunun verilen dönme açısı miktarı kadar motoru döndürmesidir. Dönme açısı robotun ne kadar döneceğini değil, motorun kaç derece döneceğini ifade eder. Verilen görevi yerine getirmek için aşağıdaki komutlar kullanılabilir (**Dikkat**: Buradaki hız ve bekleme süresi değerlerini öğrencilerin kendi durumlarına göre değiştirmesi gerekebilir).

```
motorB=Motor(Port.B) #Sol motor tanımlandı
motorB.run_angle(500,360,Stop.BRAKE,True) # Motor 360 derece döner
```

1.6.4. Drive_time Metodu

"drive_time" (*hız, dönüş oranı, süre*) metodunun kullanılması için bir robot tanımlanmalıdır. Pozitif dönüş oranları sağa ve negatif dönüş oranları sola dönüş için kullanılır. Aşağıdaki kodlar verilen görevi yerine getirmek için kullanılabilir.

```
motorB=Motor(Port.B)
motorC=Motor(Port.C)
robot=DriveBase(motorB,motorC,56,114)
robot.drive_time(500,500,650)
robot.stop(Stop.BRAKE)
```

Bu komutları öğrencilere gösterdikten sonra öğrencilerden dönüş oranını azaltıp dönme süresini artırarak aynı görevi yerine getirmeleri istenir. Öğrencilerin farklı dönüş oranlarında robotun nasıl hareket ettiğini gözlemlemeleri sağlanır.

1.7. Uygula: Kare Üzerinde Hareket Eden Robot

Bu görevde öğrencilerden kare şeklinde bir yörüngede hareket eden robot kodunu yazmaları istenir. Öğrenciler çözümleri için herhangi bir sensör kullanmayacaktır. Yalnızca motor ile ilgili komutları kullanabilirler.

1.8. Gözle: Robotum Ses Veriyor

MicroPython ile akıllı tuğlanın üç farklı şekilde/yöntemle ses çalması veya ses dosyasını oynatması sağlanır. Bunlar "beep", "beeps" ve "file" metodlarıdır.

1.8.1. Beep Metodu

"beep" (*frekans, süre, ses şiddeti*) metodu akıllı tuğlanın "bip" sesini çıkarmasını sağlar. Görüldüğü gibi bu metodun üç tane parametresi bulunur. Frekans parametresi ile çıkarılan "bip" sesinin Hertz cinsinden frekansı belirlenir. Süre parametresi ile milisaniye cinsinden sesin ne kadar çalınacağı belirlenir. Ses gücü ise yüzde olarak çıkarılacak sesin şiddetini belirler. Eğer bu parametrelerin hiçbiri girilmezse ön tanımlı değerler olan frekans=500, süre=100 ve ses şiddeti=30 kullanılır.

Rehber öğretmen aşağıdaki kodu yazarak öğrencilere çalışmasını gösterir. Ardından "wait" komutu kaldırıldığında akıllı tuğladan çıkan sesi dinletip öğrencilerden neden böyle bir ses çıktığını açıklamalarını ister. Son olarak onlara "wait" komutu kullanılmadığında akıllı tuğlada seslerin üst üste bineceği bilgisini verir.

```
brick.sound.beep()
wait(100)
brick.sound.beep(1500,100,50)
```

1.8.2. Beeps Metodu

"beeps" (bip sayısı) metodu akıllı tuğlanın belirlenen sayıda "bip" sesini çıkarmasını sağlar. Rehber öğretmen aşağıdaki kodu yazarak öğrencilere açıklar.

```
brick.sound.beeps(4) # 4 defa bip sesi çalar
```

1.8.3. File Metodu

"file" (*oynatılacak dosya ismi, ses şiddeti*) metodu akıllı tuğlada ismi bildirilen ses dosyasını oynatır. MicroPython ile akıllı tuğlada iki farklı şekilde ses dosyası oynatılabilir. Bunlardan ilkinde sistemde ön tanımlı olan dosyalar kullanılır. Bu ses dosyaları MicroPython'da bulunan ses dosyalarıdır. Örneğin sistemde CRYING isimli ağlama sesi bulunur. Bu dosyaların isimleri büyük harfle yazılır. Aslında bu, programcıların oluşturduğu bir gelenektir. Sabitler büyük olarak yazılır. Akıllı tuğlada ağlama sesini oynatmak için aşağıdaki komut satırı kullanılır.

```
brick.sound.file(SoundFile.CRYING)
```

GOOD_JOB isimli, İngilizcede aferin anlamına gelen dosyayı çalıştırmak için aşağıdaki kod kullanılır.

```
brick.sound.file(SoundFile.GOOD_JOB)
```

Görüldüğü gibi dosya ismini yazarken SoundFile.DosyaİSmi formatı kullanılır. Yani dosya isminden önce SoundFile. ifadesi bulunur. Bu ifade yazılmazsa komut çalışmaz ve hata verir. Sistemde ön tanımlı olarak bulunan ses dosyalarının isimlerine ve onların Türkçe karşılıklarına ulaşmak için klasörün içine bakılmalıdır.

"File" metodu ile ikinci ses dosyası oynatma yöntemi ise var olan yani daha önceden kaydedilmiş bir ses dosyasının çalıştırılmasıdır. Örneğin daha önceden kaydedilen, içinde köpek sesi olan kopek_sesi.wav isimli bir dosya olduğu kabul edilsin. Bu dosyayı akıllı tuğlada oynatmak için öncelikle dosyanın projenin bulunduğu klasöre kopyalanması gerekir. Eğer

projenin bulunduğu klasör içerisinde böyle bir dosya yoksa ses dosyası oynatılamaz. Bu dosyanın oynatılması için aşağıdaki kodun yazılması gerekir.

```
brick.sound.file('kopek_sesi.wav')
```

Yukarıda dosya ismi tırnak içerisinde ve uzantısı tam olarak yazılıdır. Aksi hâlde dosya oynatılamaz. Rehber öğretmen, öğrencilerinden bir ses kaydederek (kayıt için Ses Kaydedicisi kullanılabilir) veya internetten bir ses dosyası indirerek bu komutu çalıştırmalarını ister.

1.9. Uygula: Fil Sesi ile Kare Üzerinde Hareket Eden Robot

Bu etkinlikte robot daha önce yapıldığı gibi kare şeklinde hareket edecektir. Fakat bu uygulamada akıllı tuğla köşelere gelince fil sesi çıkarır. Akıllı tuğlanın fil sesini her köşede çıkarması gerekir (Sistemde ön tanımlı ELEPHANT_CALL isimli fil sesi dosyası bulunur).

1.10. Gözle: Döngüleri Kullanıyorum

Fil sesi ile kare üzerinde hareket eden robot etkinliğinde öğrenciler aşağıdakine benzer bir kod kullanarak görevi yerine getirmiştir (**Dikkat**: programlardaki dönüş değerleri farklı olabilir. Program öğrencilere gösterilmeden önce denenmelidir). Bu noktada öğrencilere tekrarlanan görevler için döngülerin kullanılacağı anlatılır.

```
motorB=Motor(Port.B)
motorC=Motor(Port.C)
robot=DriveBase(motorB,motorC,56,114)

# 1. ilerle- Sağa Dön- Ses Çal
robot.drive_time(500,0,500)
robot.stop(Stop.BRAKE)
motorB.run_time(500,950,Stop.BRAKE,True)
brick.sound.file(SoundFile.ELEPHANT_CALL)
# 2. ilerle- Sağa Dön- Ses Çal
robot.drive_time(500,0,500)
robot.stop(Stop.BRAKE)
motorB.run_time(500,950,Stop.BRAKE,True)
brick.sound.file(SoundFile.ELEPHANT_CALL)
# 3. ilerle- Sağa Dön- Ses Çal
robot.drive_time(500,0,500)
robot.stop(Stop.BRAKE)
motorB.run_time(500,950,Stop.BRAKE,True)
brick.sound.file(SoundFile.ELEPHANT_CALL)
# 4. ilerle- Sağa Dön- Ses Çal
robot.drive_time(500,0,500)
robot.stop(Stop.BRAKE)
motorB.run_time(500,950,Stop.BRAKE,True)
brick.sound.file(SoundFile.ELEPHANT_CALL)
```

Döngüler yinelenen görevler için kullanılır. Yukarıda aynı görev dört defa arka arkaya yazılarak işlem yerine getirilmiştir. Bunun yerine “for” döngüsü kullanılarak aynı görev aşağıdaki şekilde yazılabilir.

```
motorB=Motor(Port.B)
motorC=Motor(Port.C)
robot=DriveBase(motorB,motorC,56,114)

for sayac in range(4):
    robot.drive_time(500,0,500)
    robot.stop(Stop.BRAKE)
    motorB.run_time(500,950,Stop.BRAKE,True)
    brick.sound.file(SoundFile.ELEPHANT_CALL)
```

Burada "for" döngüsü ilk olarak gövdesindeki ilk komutu yani aşağıdaki komutu çalıştırır. Burada rehber öğretmen "for" döngüsünün yazımını ve kullanımını anlatır.

```
    robot.drive_time(500,0,500)
```

Bunun ardından gövdesindeki diğer komutları sırayla çalıştırır ve gövdesindeki son komuta geldiğinde (fil sesi) yeniden döngü gövdesindeki ilk komuta döner ve çalıştırır. Bu işlem "range" ifadesinin içerisinde yazan sayıya ulaşılana kadar devam eder. Burada "range(4)" yazıldığı için döngü gövdesindeki komutlar adım adım, 4 defa işletilir. Burada "for" döngüsünde kullanılan "sayac in range(4)" ifadesini biraz açmakta fayda vardır. "Range(4)" ifadesi sırasıyla sayaç için 0, 1, 2, 3 değerlerini üretir ve döngü sayac'ın 0, 1, 2, 3 değerleri için tekrar edilir. Eğer "sayac in range(10)" yazılmış olsaydı sayac 0, 1, 2, …, 9 değerlerini alacaktı ve döngü sayacın her bir değeri için çalıştırılacaktı. Görüldüğü gibi sayac'ın başlangıç değeri 0'dır. Fakat bunu değiştirmek mümkündür.

```
for sayac in range(2, 6):
    print(sayac)
```

Yukarıdaki kod çalıştırıldığında aşağıdaki çıktı, VS Code'un Output ekranında görülür. "print(sayac)" komutu sayacın değerlerini ekrana yazar (**Dikkat**: Çıktı bilgisayar ekranından görünür, akıllı tuğla ekranında değil). Sayaç 2, 3, 4, 5 değerlerini almıştır. "range(n,m)" komutu, n'den başlayarak m'e kadar (m dâhil değil) tam sayıları üretir.

```
2
3
4
5
```

Uyarı: Öğretmenler, bazen döngüler kullanılarak daha hızlı çalışan kodlar yazıldığını söylerler. Bu ifade doğru değildir. Yukarıda döngü kullanılarak ve döngü kullanılmadan yazılan iki kod da aynı hızda çalışır. Döngülerde asıl amaç yinelenen görevlerin tekrar eden yapılar içerisinde yapılmasını sağlamaktır.

1.11. Uygula: Tekrarlayan Sesler

Bu etkinlikteki amaç, tuğlanın birinci adımda 2, ikinci adımda 3, üçüncü adımda 4 ve dördüncü adımda 5 defa "bip" sesini çıkarmasını sağlayan programı yazmaktır. Öğrencilere bu görev için "beeps" metodunun kullanılacağı ipucu olarak verilebilir. Bu görevin çözümü aşağıdadır.

```
for sayac in range(2,6):
    brick.sound.beeps(sayac)
    wait(400)
```

1.12. Gözle: Akıllı Tuğla Ekranını Kullanıyorum

1.12.1. Metin Yazdırma

Akıllı tuğla ekranına istenilen metin yazılabilir. Aşağıdaki komut kullanılarak akıllı tuğlanın ekranına "MicroPython EV3" yazdırılır. Burada "wait" komutu kullanılmazsa metin akıllı tuğla ekranında çok kısa bir süre gösterilip program kapatılır. Yazılan "wait" komutu ile "MicroPython EV3" metni tuğla ekranında 5 saniye gösterilir.

```
brick.display.text("MicroPython EV3")
wait(5000)
```

Tuğla ekranına yeni bir yazı yazmadan önce tuğla ekranındaki yazının silinmesi istenir, bunun için "clear()" metodu kullanılır. Aşağıdaki kod ile önce tuğla ekranına "MicroPython EV3" yazılır ve iki saniye sonra ekran silinerek tuğla ekranına "TUBITAK" yazılır. Bu kodun nasıl çalıştığı öğrencilere açıklanır. Ardından öğrencilerden brick.display.clear() satırını silerek aşağıdaki kodları çalıştırıp sonucu tuğla ekranında gözlemlemeleri istenir.

```
brick.display.text("MicroPython EV3")
wait(2000)
brick.display.clear()
brick.display.text("TUBITAK")
wait(2000)
```

1.12.2. Resim Bastırma

MicroPython'da ön tanımlı bazı resimler bulunur. Bu resimleri tuğla ekranına basmak için "image" metodu kullanılır. Aşağıdaki kod ile gözleri yukarı bakan bir resim tuğla ekranına basılmıştır. Bu resmin ismi UP'dır. Görüldüğü gibi resim isminden önce ImageFile. ifadesi kullanılmalıdır. MicroPython'da ön tanımlı olarak bulunan bütün resimlere ve onların Türkçe karşılıklarına ulaşmak için klasörün içine bakılır.

```
brick.display.image(ImageFile.UP)
wait(5000)
```

Daha önceden oluşturulmuş herhangi bir resim de akıllı tuğlanın ekranına basılabilir. Fakat bu resim Python projesi ile aynı klasörde olmalıdır. Dokuzuncu haftanın klasöründe "tubitak.png" isimli bir resim dosyası bulunur. Bu resmi tuğla ekranına basmak için aşağıdaki kod kullanılır.

```
brick.display.image("tubitak.png")
wait(5000)
```

1.13. Uygula: Göz Egzersizi Animasyonu

Bu görevdeki amaç, akıllı tuğlanın ekranında gözleri sırasıyla aşağı, alt sağa, alt sola, orta sağa, orta sola ve yukarıya bakacak şekilde bir animasyon yapmaktır. Bu kod aşağıdaki gibi olabilir.

```
brick.display.image(ImageFile.DOWN)
wait(1000)
brick.display.image(ImageFile.BOTTOM_RIGHT)
wait(1000)
brick.display.image(ImageFile.BOTTOM_LEFT)
wait(1000)
brick.display.image(ImageFile.MIDDLE_RIGHT)
wait(1000)
brick.display.image(ImageFile.MIDDLE_LEFT)
wait(1000)
brick.display.image(ImageFile.UP)
wait(1000)
```

2. TASARLA

2.1. Dans Eden Robot

Robot, dans edecek şekilde programlanır. Dansta hareket, müzik ve tuğlanın ekranında görseller arka arkaya gelecek şekilde kullanılmalıdır.

Dans eden robot yapabilmek için öğrencilere dansta hareket, müzik ve tuğlanın ekranında görsellerin arka arkaya gelecek şekilde kullanılması gerektiği belirtilir. Öğrencilerden dans eden robotun kodunun nasıl yazılacağı ile ilgili düşünmeleri istenir. Öğrenciler grup olarak olası çözümler hakkında tartışırlar. Gerektiği noktada rehber öğretmen onlara yardımcı olabilir. Fakat öğrencilere tam bir çözüm verilmez. Gruplar çözümü kendileri üretmelidir, tam bir çözümün verilmesi öğrencilerin yaratıcılıklarını olumsuz yönde etkileyebileceğinden tavsiye edilmez. Tasarlama sürecinde öğrencilerin dans eden robot için aşağıda örnek olarak verilen iki adıma benzer bir süreci gerçekleştirmeleri gerekir.

Tanımlama: Öğrencilerin öncelikle istenilen dans işleminin neler gerektirdiğini belirlemesi/ortaya koymaları gerekir. Bu, bir dans stili, farklı şekillerde hareket eden, ses çıkaran ve ekranda uygun ifade gösteren robotlar olabilir. Öğrencilerin önce gerekli işlemleri maddeler hâlinde detaylı olarak yazmaları gerekir. Örnek:

- Ses çalar,
- Ses ile ardışık olarak dans eder (Sağ, sol, ileri, sağ, sol, geri hareket),
- Tuğla ekranında göz hareketleri ile sağa sola dönüş ifadesi, ileri gidince mutlu, geri gidince mutsuz ifadesi verir,
- Dans hareketini sonlandırır.

Fikir üretme: Bu aşamada öğrencilerin tanımlamada belirlenen işlemlerin nasıl yapılabileceği ile ilgili fikir yürütmesi gerekir. Örnek olarak, öğrenciler aşağıdaki maddelere benzer fikirler üretebilir.

- Öncelikle robotun nasıl hareketler yapacağı belirlenmelidir? Planlanan her bir hareketin tanımları yapılmalıdır (örneğin, ilk harekette robot önce 0,5 saniye sağa doğru döner ve durur. Sonra 2 katı hızlı bir şekilde yine 0,5 saniye sağa doğru döner ve durur). Öğrencilerin hareketleri tanımlarken hareketlerin programlamalarını hızlıca

denemelerine izin verilebilir. Öğrencilerin hareketi planlayıp sonraki adıma geçmeleri tavsiye edilir.

- Sonra her bir harekette tuğlanın çıkaracağı ses belirlenir. Öğrencilerin internetten müzik dosyaları indirmelerine veya kendi müziklerini kaydetmelerine izin verilir.
- Hareket ve ses ile tuğlanın ekranında gösterilecek görseller de planlanır.
- Bu sürecin sonunda öğrenci dans hareketini, müziğini, tuğla görüntülerini ve durum ışığı bilgilerini birlikte anlatır.

3. ÜRET

3.1. Dans Eden Robot

Bu bölümde de öğrenciler aktif rol üstlenir. Öğretmen yine rehber pozisyonundadır. Rehber öğretmen öğrencilere takıldıkları noktalarda destek olur. Bu destek yalnızca ihtiyaç anında sağlanmalıdır. Bu aşamada, öğrencilerden bir önceki adımda tasarladıkları dans eden robot planını kullanarak probleme daha yapılandırılmış bir çözüm önerisi geliştirmeleri istenir. Öğrenciler bilgisayar ve robot başında çalışarak gerekli yazılım çözümlerini geliştirirler. Burada öğrencilerin en çok zorlanacağı konulardan biri, ses, hareket ve tuğla ekranı görüntüsünün uygun bir şekilde arka arkaya gelmesini sağlamaktır. Örnek bir program aşağıda verilmiştir.

```
motorB=Motor(Port.B)
motorC=Motor(Port.C)
robot=DriveBase(motorB,motorC,56,114)

brick.sound.file(SoundFile.CHEERING)
for sayac in range(4):
    brick.display.image(ImageFile.DOWN)
    motorB.run_time(700,900,Stop.BRAKE,False)
    brick.sound.file(SoundFile.SONAR)

    brick.display.image(ImageFile.UP)
    motorB.run_time(-700,900,Stop.BRAKE,False)
    brick.sound.file(SoundFile.SONAR)

    brick.display.image(ImageFile.MIDDLE_LEFT)
    robot.drive(200,0)
    brick.sound.file(SoundFile.ERROR_ALARM,100)
    robot.stop(Stop.BRAKE)

    brick.display.image(ImageFile.MIDDLE_RIGHT)
    robot.drive(-200,0)
    brick.sound.file(SoundFile.ERROR_ALARM,100)
    robot.stop(Stop.BRAKE)

    brick.display.image(ImageFile.BOTTOM_RIGHT)
    motorB.run_time(800,800,Stop.BRAKE,False)
    brick.sound.file(SoundFile.DETECTED)

    brick.display.image(ImageFile.BOTTOM_LEFT)
    motorC.run_time(800,800,Stop.BRAKE,False)
    brick.sound.file(SoundFile.DETECTED)

    brick.display.image(ImageFile.BOTTOM_RIGHT)
    motorB.run_time(-800,900,Stop.BRAKE,False)
    brick.sound.file(SoundFile.SNAKE_RATTLE)

    brick.display.image(ImageFile.BOTTOM_LEFT)
    motorC.run_time(-800,900,Stop.BRAKE,False)
    brick.sound.file(SoundFile.SNAKE_RATTLE)

brick.sound.file(SoundFile.CHEERING)
```

4. DEĞERLENDİR

Aşağıdaki soruları ekrana yazdırın ve öğrencilerin bilgisayar kullanmalarına izin vermeden sınıfça tartışmalarını sağlayın.

Aşağıdaki programda yanlış yazılan satırlar bulunup düzeltilir.

```
motorB=motor(Port.B)
motorC=motor(Port.C)
robot=DriverBase(motorB,motorC,56,114)

brick.sound.file(CHEERING)
for sayac in range(4):
    brick.display.image(DOWN)
    motorB.run(700,900,Stop.BRAKE,Falsa)
    brick.sound.file(SONAR)
```

Aşağıdaki komut çalıştırıldığında kaç adet “bip” sesi duyulur?

```
for sayac in range(1,10):
    brick.sound.beeps(sayac)
    wait(300)
```

Akıllı tuğlanın ekranına 0,5 – 1 - 1,5 – 2 - 2,5 - 3 sayıları, döngüler kullanılarak nasıl yazdırılır (Bu sayılar aşağıda … yerine gerekli komutlar girilerek yazdırılmalıdır)?

```
for sayac in range(1,7):
    ……………………………………………………………………
    ……………………………………………………………………
```

Proje Hazırlıyorum

Bu hafta itibarıyla proje çalışmalarına başlanması önerilmektedir. Projenin “empati” süreci için planlamalar yapılmalıdır. Detaylar için EK’ler incelenebilir.

10. Hafta: MicroPython ile Dokunma Sensörü

Ön Bilgi:

- Öğrenciler robot kavramını bilir.
- Öğrenciler robot setiyle farklı robot tasarımları yapmıştır.
- Öğrenciler robot setini programlamak için grafik arayüzünü kullanmıştır.
- Öğrenciler EV3 yazılımında robotun hareket etmesi için gerekli programlama adımlarını oluşturmuştur.
- Öğrenciler EV3 yazılımında sensörlerin kullanıldığı programlar yapmıştır.
- Öğrenciler EV3 yazılımında veriler üzerinde matematik ve mantık işlemlerini gerçekleştirmiştir.
- Öğrenciler MicroPython'da hareket, ses, ekran görüntüsü ve döngüleri kullanarak programlar oluşturmuştur.

Haftanın Kazanımları:

- MicroPython'da dokunma sensörünü kullanarak program oluşturur.
- MicroPython'da farklı döngü ifadelerini kullanır.
- MicroPython'da koşul ifadelerini kullanır.
- MicroPython'da dokunma sensörü, döngü ve koşul ifadelerini birlikte kullanarak problemleri çözer.

Haftanın Amacı:

Bu haftanın amacı MicroPython ile dokunma sensörü, "while" döngüsü ve "if" koşul ifadelerinin kodlarını öğretmektir. Sonraki süreçte amaç, öğrencilerin çeşitli problemlerin çözümünde bu kodları kullanmasını sağlamaktır.

Kullanılacak Malzemeler:

Robot seti, bilgisayar, dokunma sensörü, gerekli yazılımlar

Haftanın İşlenişi:

Gözle: MicroPython ile dokunma sensörü, "while" döngüsü ve "if" koşul ifadelerinin kodlarını inceleme
Uygula: MicroPython ile dokunma sensörü, "while" döngüsü ve "if" koşul ifadelerinin kodlarını çeşitli örnekler üzerinde uygulama
Tasarla: MicroPython ile dokunma sensörü, "while" döngüsü ve "if" koşul ifadelerinin kodlarının bir problemin çözümünde kullanılabilmesi için tasarımını yapma
Üret: MicroPython ile dokunma sensörü, "while" döngüsü ve "if" koşul ifadelerinin kodlarının bir problemin çözümü için oluşturulan tasarımını ürün hâline getirme
Değerlendir: Konu değerlendirme etkinliği
Proje: Projeye empati etkinliği

1. GÖZLE VE UYGULA

1.1. Gözle: Dokunma Sensörünü Kullanıyorum

1.1.1. Dokunma Sensörü

Dokunma sensörü MicroPython'da "TouchSensor" olarak isimlendirilir. Önceki hafta motor hareketleri konusunda anlatıldığı gibi dokunma sensörü de programda kullanılmadan önce hangi porta bağlandığı bilgisi ile birlikte oluşturulmalıdır. Dokunma sensörünün oluşturulması için aşağıdaki komut kullanılır.

```
dokunmaSensoru=TouchSensor(Port.S2)
```

Burada ikinci porta bağlanan (Port.S2) bir dokunma sensörü oluşturulur. (TouchSensor(Port.S2)) ve bu sensöre dokunmaSensoru ismi verilir. Programcılar oluşturdukları sensörlere istediği ismi verebilir. Bu örnekte kullanılan dokunmaSensoru ismi zorunlu değildir.

Dokunma sensörünün "pressed" isimli bir metodu bulunur. Bu metot dokunma sensörünün butonuna basılıp basılmadığını kontrol eder. Eğer butona basılı ise doğru (True), basılı değilse yanlış (False) değeri döndürür.

```
dokunmaSensoru.pressed() # True:Basılı --- False:Basılı Değil
```

EV3 yazılımında dokunma sensörü için basılı olmadığı durum (0), basılı olduğu durum (1) ve basılıp bırakılmış olduğu durum (2) için üç farklı değer bulunmaktadır. Fakat MicroPython'da basılıp bırakılmış olduğu durum için herhangi bir değer bulunmaz. Bu durum MicroPythonda yazılacak bir kod ile çözülebilir.

1.1.2. While Döngüsü

Bu hafta yapılacak etkinliklerde döngüler kullanılacaktır. Bir önceki hafta "for" döngüsü incelenmiştir. Bu hafta "while" döngüsü incelenecektir. "While" döngüsü gövdesindeki işlemi koşul ifadesi doğru olduğu müddetçe tekrar eder.

```
dokunmaSensoru=TouchSensor(Port.S2)
while dokunmaSensoru.pressed()==False:
    brick.sound.beep()
    wait(300)
```

Yukarıdaki kod incelendiğinde, ikinci porta bir dokunma sensoru bağlandığı ve ismine dokunmaSensoru dendiği görülür. Bu komutun arkasından "while" döngüsü oluşturulur. Bu döngünün gövdesinde aşağıdaki komutlar bulunur (**Dikkat**: Döngü gövdesindeki ifadeler Tab kullanılarak içeriye kaydırılmıştır).

```
    brick.sound.beep()
    wait(300)
```

Bu komutlar döngü yinelendiği müddetçe tekrarlanır. Döngünün tekrarlanıp-sonlandırılmasına karar veren test ifadesi aşağıdaki gibidir (**Dikkat**: Test ifadesinde iki eşittir ifadesi yan yana kullanılır. Bir eşittir ifadesi kullanılırsa yanlış olur).

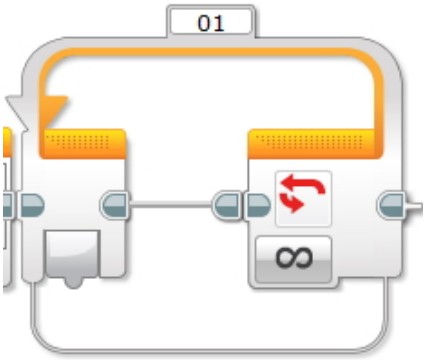
```
dokunmaSensoru.pressed()==False
```

DokunmaSensoru.pressed() dokunma sensörünün butonuna basılmadığında “False” değeri üretir. Test ifadesi dokunma sensörüne dokunulmadığını kontrol eder. Test ifadesi “while” ile : (üst üste iki nokta) arasına yazılmalıdır. Yani “while dokunmaSensoru.pressed()==False:” şeklinde yazılmalıdır. Sonuç olarak bu döngü dokunma sensörünün butonuna basılmadığı müddetçe döngü gövdesindeki “"bip” sesi çıkarma komutlarını çalıştırır. Dokunma sensörünün butonuna basıldığı zaman test ifadesi yanlış olacağı için döngüden çıkılır.

Rehber öğretmen yukarıdaki kodu çalıştırıp öğrencilere anlattıktan sonra “for” döngüsü ile “while” döngüsü arasındaki farkları sorar. Grupların görüşlerini aldıktan sonra “for” döngüsünde range komutunda belirtilen sayıda / adımda döngü gövdesinin tekrar edildiğini, while döngüsünde ise koşul ifadesi doğru olduğu müddetçe döngü gövdesindeki komutların tekrar edildiğini vurgular.

1.1.3. Sonsuz Döngü

EV3 yazılımındaki gibi bir veya birkaç işlemi sürekli tekrarlamak için aşağıdaki blok kullanılır.



Resim 160. EV3 Döngü Bloğu

Aynı görevi MicroPython ile gerçekleştirmek için döngüler kullanılır. Aşağıdaki kod bloğunda while döngüsünün gövdesi test ifadesi doğru olduğu müddetçe çalışır. Fakat aşağıda görüldüğü üzere test ifadesi “True” değerinden oluşur. Yani test ifadesi herhangi bir değere bağlı olmadan sürekli “True” (doğru) değerini alır. Bu durumda döngünün test ifadesi sürekli doğru olduğu için döngü gövdesi sürekli tekrar eder. Program sonlandırılana kadar “while” döngüsünün dışına çıkmaz, sürekli tekrar eder. Sonuç olarak aşağıdaki kod sürekli olarak “bip” sesini çalar.

```
while True:
    brick.sound.beep()
    wait(300)
```

1.1.4. Koşul / Seçim İfadeleri

Yukarıda, sürekli “bip” sesi çıkaran kod incelendi. Bu sefer dokunma sensörünün butonuna basıldığında “bip” sesi çıkaran programın yazılması istenir. Koşul ifadelerinde belirli bir koşul durumunda işlem gören kodlardan bahsedilir. Yani butona basılma koşulu gerçekleştiğinde akıllı tuğlanın “bip” sesi çıkarması gerekir. İşte MicroPython’da bu tür ifadeleri yazmak için “if” ifadesi kullanılır. Aşağıdaki komutlar incelendiğinde “if” ile “:” arasında kalan kısım koşulu ifade eder. Bu koşul doğru olduğunda “if” bloğunun gövdesindeki komutlar çalıştırılır.

Yani aşağıdaki komut programın çalışma sırası "if" bloğuna geldiğinde dokunma sensörüne basılı ise bir kere "bip" sesi çıkarır. Eğer bu esnada dokunma sensörüne basılı değilse döngünün gövdesi göz ardı edilir ve buradaki komutlar çalıştırılmaz.

```
if dokunmaSensoru.pressed()==True:
    brick.sound.beep()
    wait(300)
```

Yukarıdaki komutlar çalıştırıldığında, sadece program başladığında bir kere dokunma sensörünün butonuna basılı olup olmadığı kontrol edilir. Eğer program başladığı esnada butona basılıysa tuğla "bip" sesini çıkarır. Eğer o esnada butona basılı değilse herhangi bir ses çıkarmaz. Bu işlem sürekli tekrar edilecektir. Yani dokunma sensörüne basıldığı müddetçe akıllı tuğlanın bip sesi çıkartması sağlanacaktır. Bunun için sonsuz döngü kullanılmalıdır. Aşağıda görüldüğü gibi koşul ifadesi sonsuz döngü içine alındığı için sürekli tekrar eder.

```
while True:
    if dokunmaSensoru.pressed()==True:
        brick.sound.beep()
        wait(300)
```

Rehber öğretmen burada döngülerin ve koşul ifadelerinin gövdesine yazılan komutların tab kullanılarak içeri yazılması gerektiğini vurgular. Tab ile içeride yazılmayan ifadeler döngü veya koşul ifadesi içerisinde algılanmaz.

Uyarı: Bazı durumlarda öğrenciler "if" koşul ifadesi yerine "if" döngüsü tabirini kullanırlar. Bu durumlarda öğrenciler uyarılmalı, "if"in koşul / seçim / kontrol / karar ifadesi olduğu fakat döngü olmadığı vurgulanmalıdır.

1.2. Uygula: Engele Çarpınca Geri Giden Robot

Öğrencilerden engele çarpınca geri dönen bir robot programlamaları istenir. Aşağıdaki adımlar takip edilir:
(i) Öğrencilerden robotun bir engele çarpıncaya kadar düz gitmesini sağlamaları istenir.
(ii) Robot engele çarpınca bir miktar geri gitmelidir.
(iii) Robot son olarak kendi etrafında dönme hareketini yaparak durmalıdır.

Bu görev için örnek bir kod aşağıda verilmiştir.

```
motorB=Motor(Port.B)
motorC=Motor(Port.C)
robot=DriveBase(motorB,motorC,56,114)
dokunmaSensoru=TouchSensor(Port.S2)
while dokunmaSensoru.pressed()==False: # Touch sensör dokunulana kadar ilerle
    robot.drive(500,0)
robot.stop(Stop.BRAKE)
wait(150) #robotun tam olarak durduğunu garantilemek için
robot.drive_time(-500,0,500) # geriye git
robot.drive_time(360,360,2700) # etrafında dön
```

1.3. Uygula: Engellere Çarpan ve Her Seferinde Geri Dönme İşlemini Yapan Robot

Bu uygulamada öğrencilerden önceki uygulamaya benzer bir robot programını yazmaları istenir. Fakat bu defa robot engele çarptığı her seferde geriye dönme işlemini yapmalıdır. Yani önceki görevde bir kez gerçekleştirilen görev sürekli gerçekleştirilmelidir. Bunun için öğrencilere sonsuz "while" döngüsünün kullanılabileceği söylenebilir. Bu görev için örnek kod aşağıdadır (**Dikkat**: Etrafında 360 derece dönmesi için yazılan kod içerisindeki parametre değerleri değişiklik gösterebilir).

```
motorB=Motor(Port.B)
motorC=Motor(Port.C)
robot=DriveBase(motorB,motorC,56,114)
dokunmaSensoru=TouchSensor(Port.S2)
while True: # Sürekli tekrarla
    while dokunmaSensoru.pressed()==False: #Touch sensör dokunulana kadar ilerle
          robot.drive(200,0)
    robot.stop(Stop.BRAKE)
    wait(150) #robotun tam olarak durduğunu garantilemek için
    robot.drive_time(-500,0,500) # geriye git
    robot.drive_time(360,360,2600) # etrafında dön
```

1.4. Gözle: Değişkenler

Programcılar yazdıkları programlarda bazı değerlerin hatırlanmasını isterler. Bu değerler programın sonraki adımlarında kullanılacaktır, dolayısıyla hatırlanmalıdır. Bilgisayarlarda hatırlama görevini RAM adındaki bellekler yerine getirir. Bu belleklerde hatırlanacak değerlerin saklanması için değişken oluşturulması gerekir. Değişkenler, Bellek içerisine sayısal, sözel veya mantıksal değerlerin saklanabildiği raf zinciri olarak düşünülebilir. Her bir raf bir değişkene karşılık gelir ve içerisinde bir değer saklar. Bir raf aynı zamanda sadece bir değer tutabilir çünkü birden fazla değer saklaması mümkün değildir. MicroPython'da bir değişken tanımlamak için ona bir isim vermek gerekir. Örneğin aşağıda mevcudu 30 olan bir sınıf için sınıf mevcudunu tutan bir değişken tanımlanır.

```
sinifMevcudu=30
```

Sınıf mevcudu 7 öğrenci artırıldığında bu işlem aşağıdaki şekilde gerçekleştirilir.

```
sinifMevcudu= sinifMevcudu + 7
```

İki sınıfın mevcutları toplanıp yeni bir değişkene de atanabilir. Bunun için aşağıdaki kod örnek olarak verilebilir.

```
sinifA=32
sinifB=28
toplamMevcut=sinifA+sinifB
```

Değişken değerleri sadece sayısal değildir. Değişkenler metin değerleri de saklayabilir. Örneğin bir okulun adı saklanmak isteniyorsa aşağıdaki kod kullanılabilir.

```
okulIsmi="Mehmet Akif Ersoy"
```

Metin ifadeleri çift tırnak (veya tek tırnak) içerisinde gösterilmek zorundadır. Aksi hâlde MicroPython yazılan ifadenin metin olduğunu anlayamaz. Yukarıda tanımlanan okul isminin daha sonra sonuna "Ortaokulu" ibaresi eklenmek istenirse aşağıdaki kod kullanılabilir.

```
okulIsmi=okulIsmi+" Ortaokulu"
```

Sonuç "Mehmet Akif Ersoy Ortaokulu" olacaktır. Görüldüğü gibi + işlemi sayısal ve metin değerleri için farklı işlemleri yerine getirmiştir. + işlemi sayısal değerleri toplarken metin değerlerini yan yana ekler.

Son olarak değişkenler içerisinde mantıksal değerler (True, False) de saklanabilir. Buna bir örnek aşağıdadır.

```
testIfadesi=True
```

Rehber öğretmen son olarak burada değişken isimlendirme kurallarını anlatır.

1.5. Uygula: Her Butona Basıldığında Bir Kere "Bip" Sesi Çalan Program

Bu derste dokunma sensörünün butonuna basılı olduğu sürece "bip" sesi çıkaran tuğla programı yapılmıştır. Bunun kodu aşağıdadır.

```
while True:
    if dokunmaSensoru.pressed()==True:
        brick.sound.beep()
        wait(300)
```

Bu program ile aslında her butona basıldığında tuğla "bip" sesi çıkarmaz. Butona hızlı bir şekilde arka arkaya basıldığında da tuğla bazı basmalar için "bip" sesini çıkarmaz. Bunun nedeni "wait" komutudur. "Wait" komutu programı 300 ms beklettiği için bu süre içerisinde butona basılırsa program bunu algılamaz. Bu durumu daha iyi gözlemlemek için "wait" komutu içerisindeki parametrenin değeri artırılabilir. Örneğin "wait(300)" komutu "wait(1000)" olarak değiştirilir ve arka arkaya dokunma sensörüne basılır. Oluşan sonuç öğrencilere açıklanır.

İstenilen programın yazılabilmesi için değişkenler kullanılarak butonun önceki durumu ve şimdiki durumu modellenecektir. Butona basılması demek butonun önceki durumunun "False" (basılı değil) ve şimdiki durumunun ise "True" (basılı) olması demektir. Bu düşünce temel alınarak aşağıdaki kod yazılabilir. Rehber öğretmen aşağıdaki kodu öğrencilere açıklar.

```
dokunmaSensoru=TouchSensor(Port.S2) # Dokunma sensörü 2. porta bağlandı
basili=False # butona basılı ise True basılı değilse False
oncekiDurum=False  # butona henüz basılmamış
simdikiDurum=False  # butona henüz basılmamış
while (True):
    simdikiDurum=dokunmaSensoru.pressed() # buton değeri okunuyor
    if oncekiDurum==False and simdikiDurum==True: #Butona basıldı
        basili=True
    oncekiDurum=simdikiDurum
    if basili:
```

```
        brick.sound.beep()
    basili=False # yeniden basılma işlemi için gerekli
```

1.6. Uygula: Butona Her Basılıp Çekildiğinde "Bip" Sesi Çıkaran Program

Bu etkinlikteki amaç butona basıldığında değil, butona basılıp çekildiğinde "bip" sesi çıkaran programın yazılmasıdır. Burada öğrencilere butonun önceki ve şimdiki hâlini modelleyerek (önceki) problemin çözüleceği ipucu olarak söylenebilir. Bu etkinlik için örnek kod aşağıdadır.

```
dokunmaSensoru=TouchSensor(Port.S2) # Dokunma sensörü 2. porta bağlandı
bumped=False #basılıp bırakma. EV3 yazılımdaki dokunma sensörünün 2 değeri
onceki_durum=False  # bump demek bir kere butona basıp eli çekmek demek.
simdikiDurum=False  # önceki ve sonraki buton durumlarına ihtiyaç duyulur
while (True):
    simdikiDurum=dokunmaSensoru.pressed()
    if onceki_durum==True and simdikiDurum==False:
        bumped=True
    onceki_durum=simdikiDurum
    if bumped:
        brick.sound.beep()
    bumped=False
```

2. TASARLA

2.1. Dokunma Sensörleri ile Yönlendirilen Robot

Bu etkinlik için akıllı tuğlaya iki adet dokunma sensörü takılır. İlk sensör birinci porta, ikinci sensör ise ikinci porta takılmalıdır. Birinci porta takılan sensörün butonuna basıldığında robot sağa, ikinci porta takılı sensörün butonuna basıldığında sola dönmelidir. Her iki butona basıldığında ise robot düz ilerlemelidir. Öğrenciler grup olarak olası çözümleri tartışır. Gerektiği noktada rehber öğretmen onlara yardımcı olabilir. Fakat öğrencilere tam bir çözüm verilmemelidir. Gruplar çözümü kendileri üretmelidir. Tasarlama için öğrencilerin aşağıda örnek olarak verilen iki adıma benzer bir süreci gerçekleştirmeleri gerekir.

Tanımlama: Öğrencilerin istenilen işlemin neler gerektirdiğini belirlemesi / ortaya koyması gerekir. Öğrenciler gerekli işlemleri maddeler hâlinde yazar. Örnek:

- Sol taraftaki butona (1. sensör) basılınca sağa döner,
- Sağ taraftaki butona (2. sensör) basılınca sola döner,
- Her iki butona basılınca düz gider,
- Herhangi bir buton basılı değilse durur.

Fikir üretme: Bu aşamada öğrencilerin yukarıda belirlenen işlemlerin nasıl yapılabileceği ile ilgili fikir yürütmesi gerekir. Örnek olarak öğrenciler aşağıdaki maddelere benzer fikirler üretebilir.

- Sol butona basılı olup olmadığını kontrol etmek için "if" kullanılabilir,
- Sağ butona basılı olup olmadığını kontrol etmek için "if" kullanılabilir,
- Bu kontrollerin sürekli yapılmasını sağlamak için sonsuz döngüsü kullanılabilir,
- Sol butona basılınca sol motor çalışmalıdır,

- Sol buton bırakılınca sol motor durmalıdır,
- Sağ butona basılınca sağ motor çalışmalıdır,
- Sağ buton bırakılınca sağ motor durmalıdır.

Temel olarak bu işlemlerle robota istenilen işler yaptırılabilir. Her grubun çözümü farklı olabilir, önemli olan bu fikirlerin gruplar tarafından ortaya konulması ve ortaya konulan fikirlerin problemin çözümünü sağlayabilecek olmasıdır.

3. ÜRET

3.1. Dokunma Sensörleri ile Yönlendirilen Robot

Öğrenciler çözüm tasarımlarını yaptıktan sonra bilgisayar ve robot başında çalışarak istenilen görevi yerine getirir. Bu görev için örnek bir kod aşağıda verilmiştir.

```
motorB=Motor(Port.B) # sol motor
motorC=Motor(Port.C) # sağ motor
dokunmaSensoru1=TouchSensor(Port.S1) # 1. sensör
dokunmaSensoru2=TouchSensor(Port.S2) # 2. sensör
while True:
    if dokunmaSensoru1.pressed()==True:
        motorB.run(360)
    if dokunmaSensoru1.pressed()==False:
        motorB.stop()
    if dokunmaSensoru2.pressed()==True:
        motorC.run(360)
    if dokunmaSensoru2.pressed()==False:
        motorC.stop()
```

Aynı görev için aşağıdaki kod da kullanılabilir. Burada “pressed()” ve “not pressed()” komutları kullanılır. İki kod da aynı görevi yerine getirir fakat aşağıdaki kodu takip etmek daha kolaydır. Etkinliği erken bitiren öğrencilere aşağıdaki kod anlatılır ve kodun daha okunaklı olduğu vurgulanır.

```
motorB=Motor(Port.B) # sol motor
motorC=Motor(Port.C) # sağ motor
dokunmaSensoru1=TouchSensor(Port.S1) # 1. sensör
dokunmaSensoru2=TouchSensor(Port.S2) # 2. sensör
while True:
    if dokunmaSensoru1.pressed():
        motorB.run(360)
    if not dokunmaSensoru1.pressed():
        motorB.stop()
    if dokunmaSensoru2.pressed():
        motorC.run(360)
    if not dokunmaSensoru2.pressed():
        motorC.stop()
```

4. DEĞERLENDİR

Aşağıdaki sorular ekrana yazdırılır ve öğrencilerin bilgisayar kullanmadan sınıfça tartışmaları sağlanır.

Öğrencilerden aşağıda bulunan kod parçacığındaki yazım yanlışını / yanlışlarını bulup düzeltmeleri istenir.

```
motorB=Motor(Port.B)
motorC=Motor(Port.C)
dokunmaSensoru1=TouchSensor(Port.1)
dokunmaSensoru2=TouchSensor(Port.2)
while true:
if dokunmaSensoru1.pressed()=true:
    motorB.run(360)
if dokunmaSensoru1.pressed()=false:
    motorB.stop()
if dokunmaSensoru2.pressed()=true:
    motorC.run(360)
if dokunmaSensoru2.pressed()=false:
    motorC.stop()
```

Aşağıdaki kod parçacığı projektör ile ekrana yansıtılır ve öğrencilere bu kod parçası çalıştırıldığında motorB'nin kaç defa tam tur döneceği sorulur.

```
motorB=Motor(Port.B)
for i in range(4):
    for j in range(5):
        motorB.run_angle(500,360,Stop.COAST,True)
```

Aşağıdaki kod projektör ile ekrana yansıtılır. Öğrencilere kod çalıştırıldığında akıllı tuğlanın ekranında ne yazılı olacağı sorulur.

```
isim=" TUBITAK "
i=1
while i<5:
    isim="*" + isim
    isim=isim + "*"
    i=i+1
brick.display.text(isim)
wait(5000)
```

5. İLAVE ETKİNLİK

5.1. Oyuncak Köpeğe Çarpınca Havlayan Robot

Bu etkinlikte öğrenciler yaklaşık 50 cm uzaklıkta oyuncak bir cisme (örneğin köpeğe) çarpınca havlama sesi çıkarıp, ekranında sersemlemiş (DIZZY) görselini gösterip SORRY isimli ses dosyasını çalıştıracak bir robot tasarlayacaklardır. Öğrenciler grup olarak tartışırlar. Gerektiği noktada rehber öğretmen onlara yardımcı olabilir. Fakat öğrencilere tam bir çözüm verilmemelidir. Gruplar çözümü kendileri üretmelidir. Tasarlama için öğrencilerin aşağıda örnek olarak verilen iki adıma benzer bir süreci gerçekleştirmeleri gerekir.

Tanımlama: Öğrencilerin istenilen takip işleminin neler gerektirdiğini belirlemesi / ortaya koyması gerekir. Öncelikle öğrencilerin gerekli işlemleri maddeler hâlinde yazması gerekir. Örnek:

- Robot sürekli olarak hareket edecek.
- Karşısına bir nesne (örneğin oyuncak köpek) çıkıp çıkmadığı kontrol edilecek.
- Çapınca duracak.
    - Ekranda DIZZY görselini gösterecek.
    - Nesneye çarpınca SORRY sesini çalıştıracak.

Fikir üretme: Bu aşamada öğrencilerin yukarıda belirlenen işlemlerin nasıl yapılabileceği ile ilgili fikir yürütmesi gerekir. Örnek olarak öğrenciler aşağıdaki maddelere benzer fikirler üretebilir.

- Araç sürekli ilerlemelidir.
- Önünde herhangi bir cismin olup olmadığını kontrol etmek için dokunma sensörü ve döngü kullanılabilir.
- Çarpma olduğunu kontrol etmek için “if” (kontrol) kullanılabilir.
- Çarpma olduğunda “if” ifadesi ile ekran görüntüsü değişebilir ve istenilen ses dosyası çalıştırılabilir.
- Çarpma olmadığı sürece robotun ilerlemesi sağlanabilir.

Temel olarak bu işlemlerle robottan istenilen işlemler yapılabilir. Ama öğrenci isterse farklı nesneleri de kullanarak birden fazla ses çalıştırabilir. Her grubun çözümü farklı olabilir, önemli olan bu fikirlerin gruplar tarafından ortaya konulması ve ortaya konulan fikirlerin problemin çözümünü sağlayabilecek olmasıdır.

Öğrenciler çözüm tasarımlarını yaptıktan sonra bilgisayar ve robot başında çalışarak istenilen görevi yerine getirir. Bu görev için örnek kod aşağıdadır.

```
motorB=Motor(Port.B)
motorC=Motor(Port.C)
dokunmaSensoru=TouchSensor(Port.S2)
robot=DriveBase(motorB,motorC,56,114)
test=True
while test:
    robot.drive(200,0)
    if dokunmaSensoru.pressed():
        robot.stop()
        brick.display.image(ImageFile.DIZZY)
        wait(500)
        brick.sound.file(SoundFile.SORRY)
        wait(500)
        test=False
```

Proje Hazırlıyorum

Bu haftada proje çalışmalarına devam edilmesi önerilmektedir. Projenin "tanımlama" süreci için planlamalar yapılmalıdır. Detaylar için EK'ler incelenebilir.

11. Hafta: MicroPython ile Açı ve Mesafe Sensörü

Ön Bilgi:

- Öğrenciler EV3 yazılımında robotun hareket etmesi için gerekli programlama adımlarını oluşturmuştur.
- Öğrenciler EV3 yazılımında sensörlerin kullanıldığı programlar yapmıştır.
- Öğrenciler EV3 yazılımında veriler üzerinde matematik ve mantık işlemlerini gerçekleştirmiştir.
- Öğrenciler MicroPython'da hareket, ses, ekran görüntüsü ve döngüleri kullanarak programlar oluşturmuştur.
- Öğrenciler MicroPython'da dokunma sensörü, döngü ve koşul ifadelerini kullanarak programlar oluşturmuştur.

Haftanın Kazanımları:

- MicroPython'da mesafe sensörünü kullanarak program oluşturur.
- MicroPython'da açı sensörünü kullanarak program oluşturur.
- MicroPython'da karmaşık döngü ifadelerini kullanır.
- MicroPython'da karmaşık koşul ifadelerini kullanır.
- MicroPython'da açı sensörü, mesafe sensörü, karmaşık döngü ve karmaşık koşul ifadelerini birlikte kullanarak problemleri çözer.

Haftanın Amacı:

Bu haftanın amacı MicroPython ile açı sensörü, mesafe sensörü, karmaşık döngü ve karmaşık koşul ifadelerinin kodlarını öğretmektir. Sonraki süreçte öğrencilerin çeşitli problemlerin çözümünde bu kodları kullanmalarını sağlamaktır.

Kullanılacak Malzemeler:

Robot seti, bilgisayar, mesafe sensörü, açı sensörü, mat (çalışma alanı)

Haftanın İşlenişi:

Gözle: MicroPython ile açı sensörü, mesafe sensörü, karmaşık döngü ve karmaşık koşul ifadelerinin kodlarını inceleme

Uygula: MicroPython ile açı sensörü, mesafe sensörü, karmaşık döngü ve karmaşık koşul ifadelerinin kodlarını çeşitli örnekler üzerinde uygulama

Tasarla: MicroPython ile açı sensörü, mesafe sensörü, karmaşık döngü ve karmaşık koşul ifadelerinin kodlarının bir problemin çözümünde kullanılabilmesi için tasarımını yapma

Üret: MicroPython ile açı sensörü, mesafe sensörü, karmaşık döngü ve karmaşık koşul ifadelerinin kodlarının bir problemin çözümü için oluşturulan tasarımını ürün hâline getirme

Değerlendir: Konu değerlendirme etkinliği

Proje: Projeye tanımlama etkinliği

1. GÖZLE VE UYGULA

1.1. Gözle: Mesafe Sensörünü Kullanıyorum

Mesafe sensörü MicroPython'da UltrasonicSensor olarak adlandırılır. Mesafe sensörünü de diğer sensör ve motorlarda olduğu gibi MicroPython'a tanıtmak gerekir. Tanıtmak için onun hangi porta bağlandığını ve ismini belirtmek gerekir. Mesafe sensörünün oluşturulması için aşağıdaki komut kullanılır. Aşağıdaki kod ile sensöre, mesafeSensoru ismi verilir ve üçüncü porta bağlanır.

```
mesafeSensoru=UltrasonicSensor(Port.S3)
```

Mesafe sensörü, sensörün karşısındaki cisimlerin varlığını ve uzaklığını algılamak için kullanılır. Mesafe sensörü en fazla 255 cm uzaklıktaki cisimleri algılar. Mesafe sensörü ile karşısındaki cisim arasındaki mesafeyi ölçmek için aşağıdaki komut kullanılır. Bu komut ile karşıdaki cisme olan mesafe milimetre cinsinden bulunur.

```
mesafeSensoru.distance()
```

"Distance" metodu, "True" (EV3 yazılımında ping ifadesinin karşılığı) veya "False" (EV3 yazılımında continuous ifadesinin karşılığı) olmak üzere bir parametre alır. Python parametre yazılmadığı zaman ön tanımlı olarak "False" girildiğini varsayar. Parametre olarak "False" girildiğinde, sensör baktığı yönde sürekli ölçüm yapar. Bundan farklı olarak "mesafeSensoru.distance(True)" komutu verildiği zaman mesafeSensoru ile ölçüm yapıldıktan sonra sensör kapatılır. Bu sayede birden fazla mesafe sensörü olduğu durumlarda sensörlerin birbirlerini etkilemesi engellenmiş olur. Bu durumu daha iyi anlamak için aşağıdaki kodu öğrencilerin çalıştırması ve mesafe sensörünün ışığını gözlemlemeleri istenir. Ardından öğrencilere kod açıklanır.

```
uzaklik=0
mesafeSensoru=UltrasonicSensor(Port.S3)

uzaklik=mesafeSensoru.distance(True)
uzaklik=uzaklik/10
brick.display.text(uzaklik)
wait(5000)
brick.display.clear()

uzaklik=mesafeSensoru.distance()
uzaklik=uzaklik/10
brick.display.text(uzaklik)
wait(5000)
brick.display.clear()
```

Bazı durumlarda etrafta başka mesafe sensörleri olabilir. Bu durumlarda programcı bu mesafe sensörlerinin varlığını sorgulamak ister. Bu iş için aşağıdaki komut kullanılabilir. Bu komut ile etrafta başka bir mesafe sensörü varsa "True" yoksa "False" değeri üretilir.

```
mesafeSensoru.presence()
```

1.2. Gözle: Belirli Bir Mesafeye Kadar İlerleme

Bu etkinlikteki amaç robotu engele 5 cm kalana kadar hareket ettirip engele 5 cm kala durdurmaktır. Bu görev için aşağıdaki kod kullanılabilir. Rehber öğretmen aşağıdaki kodu öğrencilere anlatır ve çalışmasını gösterir.

```
ultrasonikSensor=UltrasonicSensor(Port.S3)
motorB=Motor(Port.B)
motorC=Motor(Port.C)
robot=DriveBase(motorB,motorC,56,114)
while ultrasonikSensor.distance()>50:
    robot.drive(180,0)
robot.stop()
```

1.3. Uygula: Belirli Bir Mesafeye Kadar Geri Gitme

(i) Öğrencilerden mesafe sensörünü kullanarak robotlarınnı mesafe sensörünün önünde bulunan engelden 20 cm geriye gittikten sonra durmasını sağlayacak programı yazmaları istenir.

(ii) Rehber öğretmen sınıfta dolaşarak öğrencilerin programı yazmasına yardımcı olur. Aşağıda gösterilen programı yazmaları için öğrencileri yönlendirir. Rehber öğretmen, öğrencilerden gelen farklı ve mantıklı fikirleri de değerlendirir ve eğer uygun ise kendi fikirleri doğrultusunda programı yazmaları için öğrencileri cesaretlendirir.

```
ultrasonikSensor=UltrasonicSensor(Port.S3)
motorB=Motor(Port.B)
motorC=Motor(Port.C)
robot=DriveBase(motorB,motorC,56,114)
while ultrasonikSensor.distance()<200:
    robot.drive(-180,0)
robot.stop()
```

1.4. Gözle: Karşıdaki Cismin Uzaklığı

Bu etkinlikteki amaç robotun karşısında bulunan bir cisme olan uzaklığını bulup cm cinsinden akıllı tuğlanın ekranına yazdırmaktır. Bu etkinlik için örnek kod aşağıdadır. Rehber öğretmen kodu öğrencilere gösterir ve çalışma mantığını anlatır.

```
uzaklik=0
mesafeSensoru=UltrasonicSensor(Port.S3)
while True:
    uzaklik=mesafeSensoru.distance()
    uzaklik=uzaklik/10
    brick.display.text(uzaklik)
    wait(500)
    brick.display.clear()
```

1.5. Uygula: Karşıdaki Cisim

Bu etkinlikteki amaç eğer karşıda bir cisim varsa akıllı tuğlanın “DETECTED” sesini çıkarması ve eğer karşıda (2550 mm mesafede) bir cisim yoksa “bip” sesi çıkarmasıdır. Öğrencilere karşıda bir cisim yokken “distance” metodunun 2550 değerini döndüreceği ve eğer karşıda bir

cisim varsa 2550'den daha küçük bir değer döndüreceği ipucu olarak söylenebilir. Bu etkinliğin örnek çözümü aşağıdadır.

```
uzaklik=0
mesafeSensoru=UltrasonicSensor(Port.S3)

while True:
    uzaklik=mesafeSensoru.distance()
    if uzaklik==2550:
        brick.sound.beep()
        wait(300)
    if uzaklik<2550:
        brick.sound.file(SoundFile.DETECTED)
        wait(300)
```

1.6. Gözle: Yeniden Koşul İfadeleri

Yukarıda yazılan kod biraz daha geliştirilecektir. Bu etkinlikte:

- Eğer karşıdaki cismin mesafe sensörüne olan uzaklığı 1000 ile 2550 (2550 dâhil değil) mm arasında ise "BACKWARDS" sesi oynatılır,
- Eğer 0 ile 1000 (1000 dâhil) arasında ise "FORWARD" sesi oynatılır,
- Eğer 2550 mm içerisinde karşıda bir nesne yoksa "STOP" sesi oynatılır.

Bu işlemi gerçekleştirmek için birden fazla "if" kullanılabilir. Fakat bu görev "if… elif… else" ifadesi ile yerine getirilecektir. Rehber öğretmen aşağıdaki kodu projeksiyon ile tahtaya yansıtır ve öğrencilere anlatır. Ardından aynı kodu öğrencilerden yazıp çalıştırmalarını ister.

```
uzaklik=0
mesafeSensoru=UltrasonicSensor(Port.S3)

while True:
    uzaklik=mesafeSensoru.distance()
    if 1000<uzaklik<2550:
        brick.sound.file(SoundFile.FORWARD)
        wait(300)
    elif 0<uzaklik<=1000:
        brick.sound.file(SoundFile.BACKWARDS)
        wait(300)
    else:
        brick.sound.file(SoundFile.STOP)
        wait(300)
```

1.7. Uygula: Engele Yaklaştıkça Yavaşlayan Robot

Bu etkinlikte karşıda bulunan engele yaklaştıkça yavaşlayan ve engele 5 cm kaldığında duran bir robot programı oluşturulur. Öğrencilere, robot ile engel arasındaki mesafeyi robotun hızı olarak alındığında robotun engele yaklaştıkça yavaşlayacağı detaylıca anlatılır. Bu görev için örnek kod aşağıdadır.

```
ultrasonikSensor=UltrasonicSensor(Port.S3) #ultrasonik sensör 3. porta takıldı
motorB=Motor(Port.B) #sol motor
motorC=Motor(Port.C) #sağ motor
robot=DriveBase(motorB,motorC,56,114) #driving base parametreleri
while ultrasonikSensor.distance()>=50: #mesafe 5cm olana kadar ilerle
    hiz=ultrasonikSensor.distance()
    robot.drive(hiz,0) #ultrasonikten gelen mesafeyi hiz olarak kullan
robot.stop()
```

1.8. Gözle: Açı Sensörünü Kullanıyorum

Açı sensörü MicroPython'da GyroSensor olarak adlandırılır. Açı sensörünü MicroPython'a tanıtmak için onun hangi porta bağlandığını ve ismini belirtmek gerekir. Açı sensörünü tanıtmak için aşağıdaki komut kullanılır. Sensöre aciSensoru ismi verilmiştir ve birinci porta bağlanmıştır.

```
aciSensoru=GyroSensor(Port.S1)
```

Açı sensörü kullanımında sıklıkla açı değerlerinin okunması gerekir. Bu iş için "angle" metodu kullanılır. "Angle" metodu derece cinsinden değer döndürür. Aşağıdaki komut açı sensörünün kaç derece döndüğünü öğrenmek için kullanılabilir.

```
aciSensoru.angle()
```

Açı sensörünü kullanmadan önce sensörün kalibrasyonunu yapmak gerekir. Bu görev için "reset_angle" metodu kullanılır. Bu metodun kullanımı aşağıda gösterilmiştir. Bu komut ile açı sensörünün açı değeri sıfırlanır. "Reset_angle" metoduna parametre olarak sıfır değil de başka bir değer girilirse açı sensörünün ilk değeri olarak bu açı alınırdı. Örneğin "aciSensoru.reset_angle(180)" yazılsaydı, açı sensörünün değeri 180 derece olarak ayarlanmış olurdu.

```
aciSensoru.reset_angle(0)
```

Açı sensörü sadece açı değeri döndürmez. Dönme hızının değerini de gösterebilir. Bu görev için "speed" metodu kullanılır. "Speed" metodu derece / saniye cinsinden açısal dönme hızını hesaplar. Bu metodun kullanımı aşağıdaki gibidir.

```
aciSensoru.speed()
```

1.9. Uygula: 360 Derece Dönen Robot

Bu uygulamanın adımları aşağıdadır:

(i) Rehber öğretmen öğrencilerden mat üzerinde yer alan açıölçeri kullanarak robotlarını olduğu yerde 360 derece dönmesini sağlayan programı yapmalarını ister.

(ii) Rehber öğretmen dolaşarak öğrencilerin programı yazmasına yardımcı olmalıdır. Öğrencileri aşağıda gösterilen programı yazmaları için yönlendirir. Öğrencilerden gelen farklı ve mantıklı fikirleri de değerlendirir ve eğer uygun ise öğrencileri kendi fikirlerindeki programı yazmaları için cesaretlendirir.

Bu görev için örnek bir kod aşağıda verilmiştir. Robotun 360 derece dönmesi için "while" döngüsünün test ifadesi içerisinde 360 kullanılmıştır. Fakat bu değer duruma göre farklılık gösterebilir. Gerekli durumda ayarlamalar yapılmalıdır.

```
motorB=Motor(Port.B)
motorC=Motor(Port.C)
robot=DriveBase(motorB,motorC,56,114)
aciSensoru=GyroSensor(Port.S1)
aciSensoru.reset_angle(0)
while aciSensoru.angle()<=360:
    robot.drive(300,300)
robot.stop(Stop.BRAKE)
```

1.10. Uygula: Açı Sensörü Kullanarak Kare Şekilli Rotada Hareket Eden Robot

Rehber öğretmen öğrencilerden robotun kare şeklinde bir rotada hareket etmesini sağlayan programı yazmalarını ister. Bu uygulamanın bulunduğu yerde 360 derece dönen robot tasarımına çok benzediği söylenir. Farklı olarak neler olması gerektiği tartışılır. Bu uygulamanın ardından istenirse üçgen ya da çokgen şeklinde rotalarda hareket eden robotlar da yapılabilir. Aşağıda bu etkinlik için örnek bir kod verilmiştir. "While" döngüsü içerisindeki dönme açısı öğrencilerin bulunduğu durumlara göre farklılık gösterebilir. Öğrenciler "while" döngüsündeki dönme açılarını kendi durumlarına göre uyarlamalıdır (**Dikkat**: Örnek kodda iç içe döngü kullanılmıştır, öğrencilere ihtiyaç duyulduğunda çözüm için iç içe döngü kullanmaları gerektiği hatırlatılabilir).

```
motorB=Motor(Port.B)
motorC=Motor(Port.C)
robot=DriveBase(motorB,motorC,56,114)
aciSensoru=GyroSensor(Port.S1)
for i in range(4):
    robot.drive_time(300,0,1000)
    robot.stop(Stop.BRAKE)
    aciSensoru.reset_angle(0)
    while aciSensoru.angle()<=62:
        robot.drive(300,300)
    robot.stop(Stop.BRAKE)
```

2. TASARLA VE ÜRET

2.1. Tasarla: Öndeki Aracı Takip Eden Robot

Bu etkinlikte robotun önünde bulunan bir arabayı (araba olmak zorunda değil herhangi bir cismi) takip etmesi sağlanır.

Öğrencilerden öndeki arabayı takip eden, yani öndeki araba ilerledikçe ilerleyen bir robotun nasıl programlanacağı üzerinde düşünmeleri istenir. Öğrenciler grup olarak tartışırlar. Gerektiği noktada rehber öğretmen onlara yardımcı olabilir. Öğrencilere tam bir çözüm verilmemelidir. Gruplar çözümlerini kendileri üretmelidir. Tasarlama için öğrencilerin aşağıda örnek olarak verilen iki adıma benzer bir süreci gerçekleştirmeleri gerekir.

Tanımlama: Öğrencilerin istenilen takip işleminin neler gerektirdiğini belirlemesi / ortaya koyması gerekir. Öncelikle öğrencilerin gerekli işlemleri maddeler hâlinde yazması gerekir. Örnek:

- Robot sürekli olarak öndeki araç ile arasındaki mesafeyi kontrol eder.
- Öndeki araç hareket ederse hareket eder.
- Öndeki araç hızlı hareket ediyorsa robot hızlanır, yavaş hareket ediyorsa yavaşlar.
- Öndeki araç durunca durur.

Fikir üretme: Bu aşamada öğrencilerin yukarıda belirlenen işlemlerin nasıl yapılabileceği ile ilgili fikir yürütmesi gerekir. Örnek olarak öğrenciler aşağıdaki maddelere benzer fikirler üretebilir.

- Önceki araç ile arasındaki mesafeyi sürekli kontrol etmek için mesafe sensörü ve döngü kullanılabilir.
- Öndeki araç ile robot arasındaki mesafe 20 cm ise robot mesafesini korur.
- Öndeki araç ile robot arasındaki mesafe 20 cm'den fazla ise robot mesafenin büyüklüğüne göre hızlanır ve aradaki mesafeyi 20 cm olarak tutmaya çalışır.
- Öndeki araç ile robot arasındaki mesafe 20 cm'den az ise robot mesafenin büyüklüğüne göre geriye doğru hızlanır ve aradaki mesafeyi 20 cm olarak tutmaya çalışır.

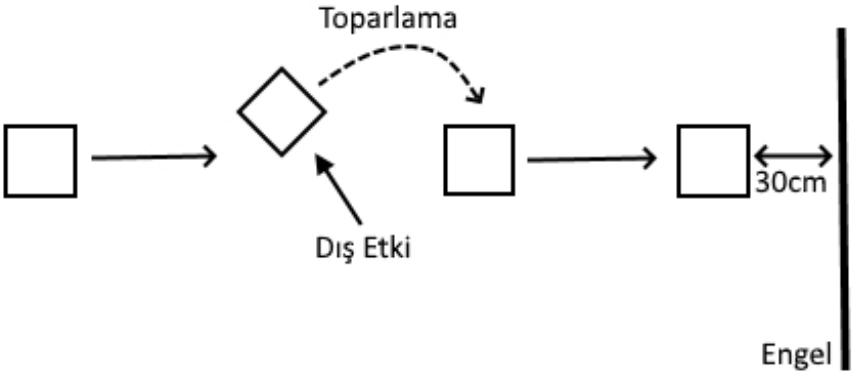
2.2. Üret: Öndeki Aracı Takip Eden Robot

Öğrenciler çözüm tasarımlarını yaptıktan sonra bilgisayar ve robot başında çalışarak istenilen görevi yerine getirir. Örnek çözüm aşağıda verilmiştir. Rehber öğretmen öğrencilerden gelebilecek farklı görüş ve çözüm önerilerine açık olmalıdır ve onları bu konuda özendirmelidir. Her grubun çözümü farklı olabilir, önemli olan bu fikirlerin gruplar tarafından ortaya konulması ve ortaya konulan fikirlerin problemin çözümünü sağlayabilecek olmasıdır.

```
ultrasonikSensor=UltrasonicSensor(Port.S3)
motorB=Motor(Port.B)
motorC=Motor(Port.C)
robot=DriveBase(motorB,motorC,56,114)
while True:
    mesafe=ultrasonikSensor.distance()/10
    hiz=(mesafe-20)*10
    robot.drive(hiz,0)
```

2.3. Tasarla: Açı ve Mesafe Sensörünün Birlikte Kullanılması

Bu etkinlikte öğrencilerden düz bir şekilde gitmekte olan (herhangi bir çizgiyi takip etmeden) robotun dışarıdan yapılacak herhangi bir müdahale sonrası rotasına tekrar dönmesini ve karşıdaki engele 30 cm yaklaşınca durmasını sağlayan robot tasarlamaları istenir.



Resim 161. Etkinlik Şeması

Öğrencilerden robotu düz bir şekilde ilerlemesi için programlamaları istenir. Fakat robot düz bir şekilde ilerlerken, robotun yönünü sağa veya sola döndürecek bir müdahale sonucunda yönünü tekrar başlangıçtaki yöne çevirecek şekilde program tasarlamaları gerekir. Bunun yanında karşıda bulunan duvar gibi büyük bir engele 30 cm kala robotun durması sağlanmalıdır. Öğrenciler yapılması gereken programı grup olarak tartışır. Rehber öğretmen gerekli noktalarda yönlendirici sorular sormalı ve önerilerde bulunmalıdır. Fakat çözümü hazır olarak vermemelidir.

Tanımlama: Öğrenciler problemi tanımlamalıdır. Problemi çözmelerine yardımcı olacak aşağıdaki soruların cevaplarını kendi aralarında tartışmalıdırlar.

- Robot düz olarak gidip gitmediğini nasıl algılayabilir?
- Robot başlangıçtaki yönünden ne kadar uzaklaştığını nasıl algılayabilir?
- Robot rotadan çıkıp çıkmadığını ne zaman kontrol edecek?
- Robotun tekrar rotasına dönüşü nasıl sağlanabilir?
- Robotun engele 30 cm kala durması nasıl sağlanabilir?

Fikir üretme: Bu aşamada öğrencilerin yukarda belirlenen işlemleri robotun nasıl gerçekleştirebileceği ile ilgili fikir yürütmesi gerekir. Örnek olarak öğrenciler aşağıdaki maddelere benzer fikirler üretebilir.

- Düz olarak gitmek için açı sensörü ve “loop” bloğu beraber kullanılır.
- Rotadan sapma açısı hesaplanmalıdır.
- Tekrar rotaya dönmek için sapma açısı kadar dönüş yapılır.
- Engele 30 cm kalınca durmasını sağlamak için mesafe sensörü ve döngüler kullanılır.

2.4. Üret: Açı ve Mesafe Sensörünün Birlikte Kullanılması

Bu bölümde de öğrenciler aktif rol üstlenirler. Öğretmen yine rehber pozisyonundadır. Rehber öğretmen öğrencilere takıldıkları noktalarda destek olur. Destek yalnızca ihtiyaç anında sağlanmalıdır. Üret aşamasında öğrencilerden probleme daha yapılandırılmış bir çözüm önerisi geliştirmesi istenir. Öğrenciler bilgisayar ve robot başında çalışarak gerekli yazılım çözümlerini geliştirirler. Burada öğrencilerin en çok zorlanacağı konu sapma açısının hesaplanarak aktarılmasının sağlanmasıdır. Bu konuda öğrencilere yardımcı olunabilir. Örnek bir çözüm aşağıda verilmiştir. Bu koddaki tepkiYonDegeri’ni bulmak için aci değişkeni değeri ile çarpılan

-5 katsayısı ve “drive” metodu içerisindeki 300 olan hız değeri öğrencilerin bulundukları durumlar için farklı değerler olarak ayarlanabilir. Gerekli düzenlemeler yapılmalıdır.

```
ultrasonikSensor=UltrasonicSensor(Port.S3)
aciSensoru=GyroSensor(Port.S1)
motorB=Motor(Port.B)
motorC=Motor(Port.C)
robot=DriveBase(motorB,motorC,56,114)
aciSensoru.reset_angle(0)
while ultrasonikSensor.distance()/10>=30:
    aci=aciSensoru.angle()
    tepkiYonDegeri=aci*(-5)
    robot.drive(300,tepkiYonDegeri)
```

Dikkat 1: Bu etkinlik mat üzerinde değil yerde ve duvara karşı yapılmalıdır. Aksi hâlde robot masanın üzerinden düşebilir.

Dikkat 2: Bazen açı sensörü akıllı tuğla üzerinden kalibre edilmelidir. Kalibre edilmediği durumlarda programda istenmeyen sonuçlar üretilebilir. Bu tip durumlarda akıllı tuğladaki menüden aşağıdaki işlemler gerçekleştirilir:

Device Browser→Sensors→Gyro at…→Set mode (Aşağı Ok Tuşu ile ulaşılır)→ GYRO CAL→GYRO G&A→GYRO ANG

Bu işlem gerçekleştirildikten sonra aşağıdaki şekilde açı sensörünün ürettiği değerler gözlenir. Eğer açı sensörü yanlış değerler üretiyorsa yukarıdaki işlem bir veya birkaç kez tekrar edilerek doğru değerler üretmesi sağlanır.

Device Browser→Sensors→Gyro at…→Watch vaules (Aşağı Ok Tuşu ile ulaşılır).

3. DEĞERLENDİR

Aşağıdaki kod parçası ekrana yansıtılır ve öğrencilere bu kod parçası çalıştıktan sonra akıllı tuğlanın ekranına ne yazılacağı sorulur.

```
sonuc=1
for sayac in range(1,6):
    sonuc=sonuc*sayac
brick.display.text(sonuc)
wait(5000)
```

Aşağıdaki kod ekrana yansıtılır ve öğrencilere bu kod çalıştırıldığında akıllı tuğlanın kaç defa "bip" sesi oynatacağı sorulur.

```
adet=0
i=0
while i<4:
    for j in range(4):
        if adet<=5:
            adet=adet+1
        elif 5<adet<=20:
            adet=adet+2
        else:
            adet=adet+5
    i=i+1
brick.sound.beeps(adet)
```

Aşağıdaki kod bloğu ekrana yansıtılır ve yazım yanlışının / yanlışlıklarının bulunup düzeltilmesi istenir.

```
ultrasonikSensor=ultrasonicSensor(Port.S3)
aciSensoru=gyroSensor(Port.S1)
motorSol=Motor(Port.B)
motorSag=Motor(Port.C)
robot=driveBase(motorSol,motorSag,56,114)
aciSensoru.reset(0)
while ultrasonikSensor.distance()/10>=30:
aci=aciSensoru.angle()
tepkiYonDegeri=aci*(-5)
robot.drive(300,tepkiYonDegeri)
```

4. İLAVE ETKİNLİK

4.1. Park Sensörü

Öğrencilerden robotu araçlardaki park sensörüne benzer şekilde programlamaları istenir. Robotun belirli bir mesafeye gelince uyarı vermesi, mesafe azaldıkça sesi yükseltmesi ve sesin tekrar aralığını artırması istenir.

Tasarla: Öğrencilerin programın adımlarını detaylı olarak planlamaları (mesafeleri belirlemeleri, belirlenen mesafelerdeki işlemleri tanımlamaları, kullanılacak seslere ve mesafeler değiştikçe robotun nasıl davranacağına karar vermeleri) ve bu ögelerin programla nasıl yapılabileceğini tasarlamaları gerekir. Programı yazmaya başlamadan önce grupların tasarlama adımı için tanımlama ve fikir üretme sürecini gerçekleştirmeleri gerekir.

Üret: Bu bölümde de öğrenciler aktif rol üstlenir. Öğretmen yine rehber pozisyonundadır. Rehber öğretmen öğrencilere takıldıkları noktalarda destek olur. Destek yalnızca ihtiyaç anında, rehber öğretmen tarafından sağlanmalıdır. Üret aşamasında, öğrencilerden bir önceki adımda tasarladıkları robot planını kullanarak probleme algoritmik -daha yapılandırılmış- bir çözüm önerisi geliştirmeleri istenir. Öğrenciler bilgisayar ve robot başında çalışarak gerekli

yazılım çözümleri geliştirirler. Burada öğrencilerin en çok zorlanacağı konulardan biri, ses, hareket ve mesafe sensörü senkronizasyonunun sağlanmasıdır.

```
motorB=Motor(Port.B)
motorC=Motor(Port.C)
ultrasonikSensor=UltrasonicSensor(Port.S3)
robot=DriveBase(motorB,motorC,56,114)

while ultrasonikSensor.distance()>=50:
    if ultrasonikSensor.distance()<=150:
        ses_siddeti=180-ultrasonikSensor.distance()
        ses_suresi=ultrasonikSensor.distance()
        brick.sound.beep(500,ses_suresi,ses_siddeti)
    hiz=ultrasonikSensor.distance()/3
    robot.drive(hiz,0)
robot.stop()
```

Proje Hazırlıyorum

Bu haftada, proje çalışmalarına devam edilmesi önerilmektedir. Projenin “fikir üretme” süreci için planlamalar yapılmalıdır. Detaylar için EK’ler incelenebilir.

12. Hafta: MicroPython ile Renk Sensörü

Ön Bilgi:

- Öğrenciler EV3 yazılımında sensörlerin kullanıldığı programlar yapmıştır.
- Öğrenciler EV3 yazılımında veriler üzerinde matematik ve mantık işlemlerini gerçekleştirmiştir.
- Öğrenciler MicroPython'da hareket, ses, ekran görüntüsü ve döngüleri kullanarak programlar oluşturmuştur.
- Öğrenciler MicroPython'da dokunma sensörü, döngü ve koşul ifadelerini kullanarak programlar oluşturmuştur.
- Öğrenciler MicroPython'da açı sensörü, mesafe sensörü, karmaşık döngü ve karmaşık koşul ifadelerini kullanarak programlar oluşturmuştur.

Haftanın Kazanımları:

- MicroPython'da renk sensörünü kullanarak program oluşturur.
- MicroPython'da akıllı tuğla tuşlarını kullanarak program oluşturur.
- MicroPython'da break ifadesini kullanır.
- MicroPython'da birden fazla sensörü birlikte kullanarak problemleri çözer.

Haftanın Amacı:

Bu haftanın amacı MicroPython ile farklı sensörlerin akıllı tuğla tuşlarını ve "break" komutu kodlarını öğretmektir. Sonraki süreçte, öğrencilerin çeşitli problemlerin çözümünde bu kodları kullanmasını sağlamaktır.

Kullanılacak Malzemeler:

Robot seti, bilgisayar, sensörler, mat (çalışma alanı)

Haftanın İşlenişi:

Gözle: MicroPython ile farklı sensörlerin akıllı tuğla tuşlarını ve "break" komutu kodlarını inceleme

Uygula: MicroPython ile farklı sensörlerin akıllı tuğla tuşlarını ve "break" komutu kodlarını çeşitli örnekler üzerinde uygulama

Tasarla: MicroPython ile farklı sensörlerin akıllı tuğla tuşlarının ve "break" komutu kodlarının bir problemin çözümünde kullanılabilmesi için tasarımını yapma

Üret: MicroPython ile farklı sensörlerin akıllı tuğla tuşlarının ve "break" komutu kodlarının bir problemin çözümü için oluşturulan tasarımını ürün hâline getirme

Değerlendir: Konu değerlendirme etkinliği

Proje: Projede fikir üretme etkinliği

1. GÖZLE VE UYGULA

1.1. Gözle: Renk Sensörünü Kullanmaya Başlıyorum

Renk sensörü MicroPython'da ColorSensor olarak adlandırılır. Renk sensörünü oluşturmak için aşağıdaki komut kullanılır. Burada renk sensörü dördüncü porta takılmıştır.

```
renkSensoru=ColorSensor(Port.S4)
```

Renk sensörü siyah (Color.BLACK), mavi (Color.BLUE), yeşil (Color.GREEN), sarı (Color.YELLOW), kırmızı (Color.RED), beyaz (Color.WHITE), kahverengi (Color.BROWN) renklerini algılayabilir. Bunun yanında renk sensörü, karşısında hiçbir renk olmadığını da (None) algılayabilir. Bu renkleri algılamak için "color" metodu kullanılır. "Color" metodu aşağıdaki şekilde kullanılabilir. Bu metod bahsi geçen yedi renkten birini (veya herhangi bir renk) algılamazsa None değerini döndürür.

```
renkSensoru.color()
```

Aşağıdaki komut renk sensörünün algıladığı değer siyah ise akıllı tuğlanın ekranına Siyah yazar. Yalnız bu komut sürekli çalışmaz; yalnızca program çalıştığı anda çalışır. Bu kodun sürekli çalıştırılması istenirse bir döngü kullanılmalıdır.

```
if renkSensoru.color()==Color.BLACK:
    brick.display.text("Siyah")
```

1.2. Uygula: Karşılaşılan Rengin İngilizcesini Söyleyen Program

Bu uygulamada, sabit duran bir robotun renk sensörüne gösterilen farklı cisimlerin renklerinin İngilizce adlarını söylemesini sağlayan bir program yazılacaktır. Program eğer herhangi bir renk algılamazsa akıllı tuğla "bip" sesi çıkaracaktır. Bu program için mat üzerinde bulunan farklı renkler kullanılabilir. Bu program için örnek kod aşağıdadır.

```
renkSensoru=ColorSensor(Port.S4)
while True:
    if renkSensoru.color()==Color.BLACK:
        brick.sound.file(SoundFile.BLACK)
    elif renkSensoru.color()==Color.WHITE:
        brick.sound.file(SoundFile.WHITE)
    elif renkSensoru.color()==Color.BROWN:
        brick.sound.file(SoundFile.BROWN)
    elif renkSensoru.color()==Color.BLUE:
        brick.sound.file(SoundFile.BLUE)
    elif renkSensoru.color()==Color.GREEN:
        brick.sound.file(SoundFile.GREEN)
    elif renkSensoru.color()==Color.YELLOW:
        brick.sound.file(SoundFile.YELLOW)
    elif renkSensoru.color()==Color.RED:
        brick.sound.file(SoundFile.RED)
    else:
        brick.sound.beep()
```

1.3. Gözle: Yansıyan Işık Şiddeti

Renk sensörü ışık şiddetini de ölçebilir. Işık şiddetini ölçmek için iki mod bulunur. Bunlar yansıyan ışık şiddeti ve ortam ışığı şiddetidir. Yansıyan ışık şiddetini ölçmek için renk sensörü, karşısındaki yüzeye kırmızı bir ışık gönderir ve o yüzeyden yansıyan ışığın şiddetini 0 ile 100 arasında bir değer olarak belirler. 0 en karanlık, 100 ise en aydınlık değeri ifade eder. Ölçülen değer 0'a ne kadar yakınsa yüzeyden yansıyan ışık o kadar azdır, yani o ölçüde karanlıktır. Ölçülen değer ne kadar 100'e yakınsa yüzeyden yansıyan ışık o kadar fazladır, yani o ölçüde aydınlıktır. Yansıyan ışık şiddetini bulmak için "reflection" metodu kullanılır. Bu metodun kullanımı aşağıdaki gibidir. "Reflection" metodu 0 ile 100 arasında bir değer döndürür.

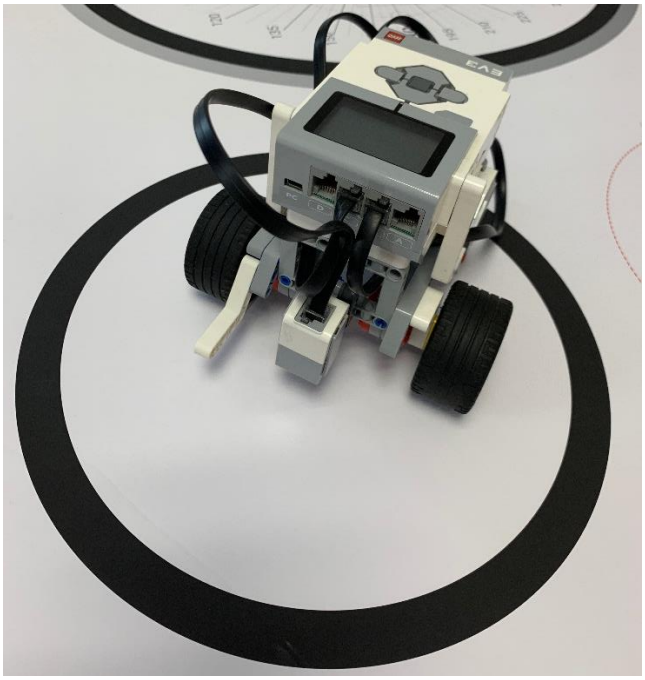
```
renkSensoru.reflection()
```

Aşağıdaki program, yansıyan ışık şiddetini akıllı tuğlanın ekranında gösterir. Öğrencilerden bu programı kullanarak mat üzerindeki farklı renklerden yansıyan ışık miktarlarını bulmaları istenir.

```
renkSensoru=ColorSensor(Port.S4)
while True:
    yansiyanIsik=renkSensoru.reflection()
    brick.display.text(yansiyanIsik)
    wait(500)
```

1.4. Uygula: Dairenin İçerisinden Çıkmayan Robot

Bu etkinlikte, beyaz bir zemin üzerinde, siyah şeritle çizilmiş dairesel bir bölgede hareket edip bu bölgeden çıkmayan robot yapılır.



Resim 162. Kullanılacak Mat Bölgesi

Öğrencilere programı yazmadan önce bu işlemi gerçekleştirmek için nasıl bir algoritma kullanılması gerektiği sorulur ve bu algoritma sınıfça tartışılır. Aşağıda örnek bir algoritma verilmiştir:

iv. Siyah şeridi görene kadar ilerle,
v. Siyah şeridi görünce robotun sol tekerini 100 derece geriye döndür,
vi. i ve ii adımlarını sürekli tekrarla.

Bu görev için gerekli algoritma başlıkları tartışıldıktan sonra öğrenciler programlarını yazmaya başlayabilirler. Örnek bir program aşağıdadır.

```
motorB=Motor(Port.B)
motorC=Motor(Port.C)
renkSensoru=ColorSensor(Port.S4)
robot=DriveBase(motorB,motorC,56,114)
while True:
    if renkSensoru.reflection()<10:
        robot.stop()
        motorB.run_angle(-100,100,Stop.BRAKE,True)
    else:
        robot.drive(100,0)
```

1.5. Gözle: Ortam Işığı Şiddeti

Ortam ışığı yoğunluğu modunda renk sensörü herhangi bir ışık göndermez. Doğrudan bulunduğu ortamdan gelen ışığın yoğunluğu ile ilgilenir. Bu yoğunluk değeri yine 0 ile 100 arasındadır. 0 en karanlık, 100 ise en aydınlık değeri ifade eder. Ortamda bulunan ışığın şiddeti "ambient" metodu aracılığı ile elde edilir. Bu metodun kullanımı aşağıdaki gibidir.

```
renkSensoru.ambient()
```

1.6. Uygula: Ortam Işığı

Bu etkinlikte renk sensörünün ölçtüğü ortam ışığı değerini her 0,5 saniyede ekrana yazdıran program yazılacaktır. Örnek bir program aşağıdadır.

```
renkSensoru=ColorSensor(Port.S4)
while True:
    a=renkSensoru.ambient()
    brick.display.clear()
    brick.display.text(a,(80,60))
    wait(500)
```

1.7. Gözle: Dokunma Sensörü ile Programı Sonlandırıyorum - Break İfadesi

Bu etkinlikte yine renk sensörünün ölçtüğü ortam ışığı değerini her 0,5 saniyede ekrana yazdıran program yazılacaktır. Fakat bu işlem sürekli gerçekleştirilmeyecektir. Bu defa dokunma sensörünün butonuna basıldığında "bip" sesi oynatılır ve programdan çıkılır. Bu işlem farklı yollardan gerçekleştirilebilir. Ancak burada "break" ifadesi kullanılarak gerçekleştirilecektir. "Break" ifadesi döngülerden çıkmak için kullanılır. Döngülerin içerisinde "break" ifadesi kullanıldığı anda ("break kod" satırı işlendiğinde) döngüden çıkılır. Aşağıda görev için istenilen kod yazılıdır. Bu kod öğrencilere gösterilerek anlatılır. Öğrencilerin kodu denemeleri sağlanır.

```
renkSensoru=ColorSensor(Port.S4)
dokunmaSensoru=TouchSensor(Port.S2)
while True:
    if dokunmaSensoru.pressed(): # uzun basmayı gerektirebilir
        brick.sound.beep()
        break
    a=renkSensoru.ambient()
    brick.display.clear()
    brick.display.text(a,(80,60))
    wait(100)
```

1.8. Gözle ve Uygula: Akıllı Tuğla Tuşları ile Programı Sonlandırıyorum

Akıllı tuğla üzerindeki tuşların kullanımı için "buttons" metodu çağrılır. Bununla birlikte butonun tuşlarının MicroPython karşılığı bilinmelidir. Bu ifadelerin karşılıkları aşağıdaki şekildedir:

Button.CENTER → Orta Tuş	Button.LEFT → Sol Tuş	Button.RIGHT → Sağ Tuş
Button.UP → Üst Tuş	Button.LEFT_UP → Sol Üst	Button.RIGHT_UP → Sağ Üst
Button.DOWN → Alt Tuş	Button.LEFT_DOWN→ Sol Alt	Button.RIGHT_DOWN → Sağ Alt

Bir tuşa basılıp basılmadığı aşağıdaki şekilde kontrol edilebilir. Aşağıdaki kod ile orta tuşa basıldığında "bip" sesi çıkar. Aynı şey sağ buton için yapılmak istenirse "if Button.RIGHT in brick.buttons():" ifadesi kullanılmalıdır.

```
while True:
    if Button.CENTER in brick.buttons():
        brick.sound.beep()
        wait(300)
```

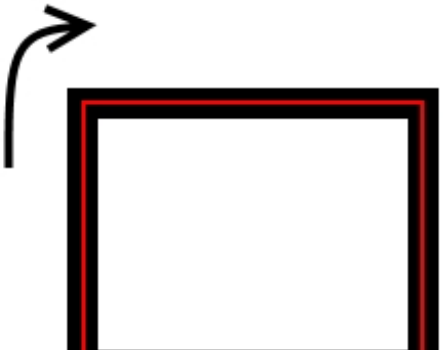
Aşağıda yazılan kod bir önceki programın tuğlanın orta butonu ile sonlandırılan hâlidir. Öğrencilerden bu programı yazmaları istenir.

```
renkSensoru=ColorSensor(Port.S4)
while True:
    if Button.CENTER in brick.buttons(): # uzun basmayı gerektirebilir
        brick.sound.beep()
        break
    a=renkSensoru.ambient()
    brick.display.clear()
    brick.display.text(a,(80,60))
    wait(100)
```

2. TASARLA VE ÜRET

2.1. Tasarla: Dikdörtgen Üzerinde Hareket Eden Robot

Bu etkinlikte amaç, robotun dikdörtgen şeklindeki çizgiyi takip etmesi için program yazmaktır. Burada dikdörtgenin kenarları siyah ve arka plan rengi beyazdır. Bu görev için mat üzerindeki dikdörtgen kullanılabilir. Robotun dikdörtgenin kenarlarının dış kısmını (iç kısmından takip eden de yapılabilir fakat burada dış kısmından takip etmelidir) takip ederek saat yönüne doğru ilerlediği varsayılacaktır. Aşağıda dikdörtgen örneği verilmiştir. Dikdörtgenin üzerindeki kırmızı hat robotun takip etmesinin hedeflendiği hattır. Kırmızı hat gösterim amaçlı çizilmiştir. Gerçekte dikdörtgenin üzerinde bulunmamaktadır. Robot birebir kırmızı hattı takip etmeyecektir fakat bu çizgiye yakın bir şekilde hareket etmesi planlanmıştır. Programı yazmaya başlamadan önce grupların tasarlama adımı için yukarıda bir örneği verilen tanımlama ve fikir üretme sürecini gerçekleştirmeleri gerekir.



Resim 163. Dikdörtgen Takip Bölgesi

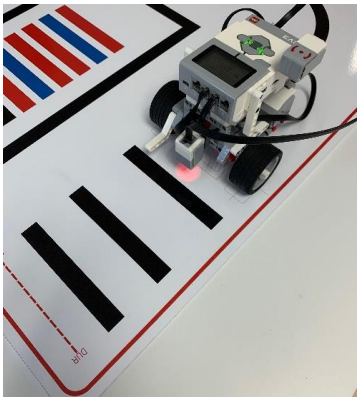
2.2. Üret: Dikdörtgen Üzerinde Hareket Eden Robot

Öğrenciler çözüm tasarımlarını yaptıktan sonra bilgisayar ve robot başında çalışarak istenilen görevi yerine getirmelidir. Aşağıdaki çözüme benzer bir örnek hazırlayabilirler. Bu çözümde yukarıdaki dikdörtgende, kırmızı bölgenin yansıyan ışık miktarı değeri 10 olarak alınmıştır. Fakat bu değer öğrencilerin durumuna göre farklılık gösterebilir. Öğrencilerin robotu kullanarak bu değeri bulup kendi durumlarına göre değiştirmeleri gerekir. "yonTepkisi" değerindeki 7 katsayısı öğrencinin durumuna göre farklılık gösterebilir. Bu değer, dönme tepkisinin ne kadar hızlı verileceği ile ilgilidir. Gerekli düzenlemeler yapılmalıdır. Aşağıdaki programa göre robot dikdörtgenin dış yüzünde olmalıdır. Eğer iç yüzüne konulursa program çalışmaz.

```
renkSensoru=ColorSensor(Port.S4)
motorB=Motor(Port.B)
motorC=Motor(Port.C)
robot=DriveBase(motorB,motorC,56,114)
while True:
    a=renkSensoru.reflection()
    yonTepkisi=(a-10)*7
    robot.drive(50,yonTepkisi)
```

2.3. Tasarla: Üçüncü Çizgiyi Geçince Duran Robot

Bu etkinlikteki amaç robotun aşağıda gösterildiği gibi üçüncü çizgiyi kısa bir miktar geçince durması için program yazmaktır. Robot ilk iki çizgiyi geçer ve üçüncü çizginin hemen ardından durur. Programı yazmaya başlamadan önce grupların tasarlama adımı için tanımlama ve fikir üretme sürecini gerçekleştirmeleri gerekir.



Resim 164. Kullanılacak Mat Bölgesi

2.4. Üret: Üçüncü Çizgiyi Geçince Duran Robot

Öğrenciler çözüm tasarımlarını yaptıktan sonra bilgisayar ve robot başında çalışarak istenilen görevi yerine getirir. Öğrenciler, aşağıdaki çözüme benzer bir örnek hazırlayabilirler.

Döngünün içerisindeki “robot.drive_time(180,0,300)” komutu siyah çizgilerin kalınlığına göre ayarlanmalıdır. Öğrenciler mat üzerindeki üç çizgiyi programlarını denemek için kullanabilir.

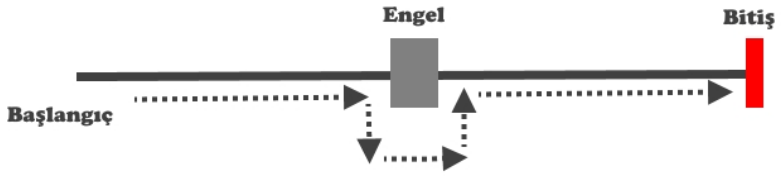
```
motorB=Motor(Port.B)
motorC=Motor(Port.C)
renkSensoru=ColorSensor(Port.S4)
robot=DriveBase(motorB,motorC,56,114)
for i in range(3):
    while renkSensoru.color()!=Color.BLACK:
        robot.drive(180,0)
    robot.drive_time(180,0,300)
```

2.5. Tasarla: Çizgi Takip Ederken Engel Aşan Robot

Rehber öğretmen öğrencilerden bu programın yazılmasını ister. Bu programda robot:

v. Herhangi bir engelle karşılaşmadığında çizgiyi takip eder,
vi. Engel ile karşılaştığında çizgiden ayrılıp engelin sağından geçer,
vii. Engeli geçtikten sonra yeniden çizgi üzerine gelir ve çizgi üzerinden devam eder,
viii. Kırmızı renkteki bitiş çizgisine geldiğinde durur.

Robotun yapması gereken görevin şeması aşağıdadır. Görev için birden fazla engel kullanılabilir.



Resim 165. Etkinlik Şeması

Bu programı yazmak için öğrenciler renk sensörü, açı sensörü ve mesafe sensörünü kullanmak zorundadır. Programı yazmaya başlamadan önce grupların tasarlama adımı için yukarıda bir örneği verilen “tanımlama” ve “fikir üretme” sürecini gerçekleştirmeleri gerekir.

2.6. Üret: Çizgi Takip Ederken Engel Aşan Robot

Öğrencilerden tanımladıkları ve hakkında fikir ürettikleri programı yazıp, kontrol edip eksikliklerini düzenlemeleri istenir. Örnek bir kod aşağıda verilmiştir.

```
motorB=Motor(Port.B)
motorC=Motor(Port.C)
robot=DriveBase(motorB,motorC,56,114)
ultrasonikSensor=UltrasonicSensor(Port.S3)
renkSensoru=ColorSensor(Port.S4)
aciSensoru=GyroSensor(Port.S1)
while True:
    if ultrasonikSensor.distance()>=140:
        if renkSensoru.reflection()>10:
            motorB.stop()
            motorC.run(200)
```

```
        else:
            motorC.stop()
            motorB.run(200)
    else:
        aciSensoru.reset_angle(0)
        while aciSensoru.angle()<=60:
            robot.drive(100,200)
        robot.stop(Stop.BRAKE)
        robot.drive_time(200,0,600)
        robot.stop(Stop.BRAKE)
        aciSensoru.reset_angle(0)
        print(aciSensoru.angle())
        while aciSensoru.angle()>=-70:
            robot.drive(100,-200)
        robot.stop(Stop.BRAKE)
        robot.drive_time(150,0,1000)
        robot.stop(Stop.BRAKE)
        robot.drive_time(100,-100,300)
        while renkSensoru.reflection()>=10:
            robot.drive(100,-10)
        robot.stop(Stop.BRAKE)
    if renkSensoru.color()==Color.RED:
        break
```

3. DEĞERLENDİRME

Aşağıdaki sorular ekrana yansıtılır ve öğrencilerin sınıfça tartışmaları sağlanır.

(i) Renk sensörü kullanılarak bir eğriyi izleyen robot programının yapıldığı varsayılır. Bu program için verilen eğrinin belirli noktalardaki eğimi fazlaca artırılırsa robotun çizgiden sapma ihtimali ortaya çıkar. Bu ihtimali ortadan kaldırmak için programda ne gibi değişiklikler yapılmalıdır?

(ii) Bir robota mesafe sensörü takıldığı fakat diğer sensörlerin takılmadığı varsayılır. Mesafe sensörü 3 cm'den daha yakın olan mesafeleri ölçerken hata verebilir. Fakat robotun karşıdaki engele 3 cm'den daha yakın bir mesafeden ilerlemesi istenir. (3 cm'den daha yakın fakat engele değmeyecek şekilde). Bu görevin yerine getirilebilmesi için nasıl bir algoritma oluşturulmalıdır? Tartışınız.

(iii) Sıra öğrencilerin soru yazmasına gelmiştir. Her bir grup MicroPython ile ilgili bir soru hazırlar. Soru hazırlamak için gruplara beş dakika süre verilir. Öğrenciler soruyu bir kelime işlemci kullanarak yazar ve rehber öğretmene iletir. Rehber öğretmen oluşturulan sorulardan bir veya iki tanesini seçer ve ekrana yansıtarak tüm sınıfa gösterir. Bu sorular sınıfça tartışılır.

Proje Hazırlıyorum
Bu hafta, projelerin tamamlanması önerilmektedir. Projenin “geliştirme-test etme” süreci için planlamalar yapılmalıdır. Bütün süreçler tamamlandıktan soran “sunum” süreci planlanmalıdır. Detaylar için EK’ler incelenebilir.

EK - Proje Hazırlıyorum

Öğrencilerin öğrendiklerini grup hâlinde uygulayabilecekleri proje geliştirme sürecinin, eğitimin yaklaşık son 4 haftasını kapsayacak şekilde planlanması beklenmektedir. Projelerin tanımlama, hazırlık, empati, fikir üretme, geliştirme-test etme ve sunum gibi etkinliklerle yapılması ve rehber öğretmenler gözetiminde öğrencilere ders dışı etkinliklerle destek olunması önerilmektedir.

1. HAZIRLIK (Projeye Başlama)

Etkinliğin Amacı: Gruplara açık uçlu bir proje vererek öğrencilerin yenilikçi fikir üretme, problem çözme, analitik düşünme ve grup hâlinde çalışma becerilerini geliştirmektir. Öğrenciler bu etkinlikte kendi robotlarını geliştirebildiklerini görürler ve bu sayede özgüvenleri artar.

Açıklama: Bu etkinlikten itibaren gruplar iki hafta boyunca projelerini tanımlama ve tasarlama üzerinde çalışacaklardır. Daha sonra, önceki haftalarda tanımlayıp tasarladıkları robotu oluşturmaları ve gerekli programları geliştirmeleri beklenir. Öğrenciler öncelikle çevrelerinden veya günlük hayatlarından uygun bir problem belirleyip bu probleme kendi tasarladıkları bir robot ile çözüm üretirler. Öğrencilerin problem üzerinde sistematik bir şekilde çalışmaları için grupların problemi analiz etmeleri ve kendi çözüm önerilerini tasarlamaları, geliştirmeleri ve değerlendirmeleri istenir. Bu süreçte karşılaşılacak tasarım ve programlamayla ilgili problemlerde rehber öğretmenler, öğrencilere uygun zaman ve miktarda destek olurlar. Bu destek öğrencilerin ihtiyacından ne az ne de fazla olmalıdır.

Proje çalışmasında son ürünü geliştirmek için “Tasarım ve Üretim” dersinde uygulanan tasarım odaklı düşünmenin beş aşamasının uygulanacağı söylenir.

1. Empati kurma: Hedef kullanıcı kitlenin ihtiyaç ve beklentilerini irdeleme.
2. Tanımlama: Kullanıcılardan alınan bilgilerden çıkan sonuçları yorumlayıp ihtiyacı öngörme.
3. Fikir üretme: Çözüm için çeşitli, yaratıcı ve yenilikçi fikirler üretme.
4. Prototip geliştirme: Fikirlerin hızlıca uygulandığı prototipler hazırlama.
5. Test etme: Ortaya çıkan ürünü yeniden hedef kitle ile değerlendirme.

Sonraki Haftaya Hazırlık: Bir sonraki hafta için öğrencilerin çevrelerinden ve günlük hayatlarından robotlar kullanılarak çözülebilecek bir problemi belirlemeleri gerektiği iletilir. Problemi belirlemeden önce "empati kurma" adımını gerçekleştirmeleri söylenir. Uygun bir problem sorusu belirlemeden önce problemi anlamak gerekir. Belirlenen problem, projenin zorluklarını ve hedeflerini tanımlamaya yardımcı olur. Sonrasında problemle ilgili yoğun bir araştırma ve gözlem yapılması gerekir. Süreç, projenin tamamlanması için gerekli bilgiler sağlar ve ortam hakkında bilgiler sunar. Süreç sonunda çok vakit ve emek harcanarak geliştirilen robotun, insanların işine yaraması, bir problemini çözmesi ve hayatlarını kolaylaştırması istenir. Empati süreci sonunda tasarlanan robotun kullanılabilmesi için planlanan/öngörülen ortamlardaki ihtiyaçların ve beklentilerin iyi belirlenmesi gerekir.

2. EMPATİ

Etkinliğin Amacı: Öğrenciler çevrelerinden veya günlük hayatlarından uygun bir problem belirleyip bu probleme kendi tasarladıkları bir robot ile çözüm üreteceklerdir.

Açıklama: Gruplardan belirledikleri robot projesi için yaptıkları empati çalışmasının sonucunu ve problem tanımlarını sınıfta açıklamaları istenir. Açıklamalarının aşağıdaki bilgileri içermesi gerektiği belirtilir. Empati çalışması sonucunu sunmaya başlamadan önce gruplara hazırlanma, gözden geçirme ve tekrar etme için 10 dakika süre verilir.

- Problem cümlesi: Robotun ne yapması isteniyor?
- Robotun kullanılması planlanan ortamdaki ihtiyaçlar nelerdir?
- Robotun kullanılması planlanan ortamdaki beklentiler nelerdir?

Açıklanan projelerle ilgili olarak diğer öğrencilerin ve rehber öğretmenlerin fikirleri alınır.

Rehber Öğretmene Not: Öğrenci tarafından belirlenen robot görevinin gerçekleştirilebilir olması önemlidir. Atölyelerde bulunan imkânlarla bu görevin başarılabilir olması gerekir. Bu safhada gerçekleştirilemeyecek bir görevin seçilmesi veya öğrencilerin bu tür bir projeye yönlendirilmesi, proje sürecinde vakit kaybetmelerine ve motivasyonlarının olumsuz etkilenmesine neden olabilir. Bu nedenle rehber öğretmenin proje önerilerini çok iyi incelemesi/irdelemesi ve proje önerilerinin gerçekleştirilebilir/yapılabilir olduğundan emin olması gerekir.

Sonraki Haftaya Hazırlık: Öğrencilerden, sonraki hafta için belirledikleri projeler için "tanımlama" adımlarını gerçekleştirmeleri istenir. Tanımlama adımında, "empati" adımında toplanan bilgiler bir bütün hâline getirilir. Süreçte ortaya çıkabilecek problemleri çözebilmek için nelerin yapılması gerektiği ile ilgili fikirler ortaya konur. Çözümün başarılı olabilmesi için ihtiyaçların iyi tanımlanmış olması ve bu ihtiyaçların giderilmesi için uygun çözüm önerilerinin sunulması önemlidir.

3. TANIMLAMA

Etkinliğin Amacı: Öğrenciler çevrelerinden veya günlük hayatlarından uygun bir problem belirleyip bu probleme kendi tasarladıkları bir robot ile çözüm üretirler.

Açıklama: Gruplardan belirledikleri robot projesi için yaptıkları empati ve tanımlama adımlarının sonuçlarını ve projelerini sınıfta açıklamaları istenir. Açıklamalarının aşağıdaki bilgileri içermesi gerektiği belirtilir. Empati ve tanımlama çalışmasının sonuçlarını sunmaya başlamadan önce gruplara hazırlanma, gözden geçirme ve tekrar etme için 10 dakika süre verilir.

- Problem cümlesi: Robotun ne yapması isteniyor?
- Robotun kullanılması planlanan ortamdaki ihtiyaçlar nelerdir?
- Robotun kullanılması planlanan ortamdaki beklentiler nelerdir?
- Hedeflerin gerçekleştirilmesi için yapılması gerekenler nelerdir?
- Robotun belirlenen hedefleri gerçekleştirebilmesi için tasarımı nasıl olmalıdır?

- Robotun belirlenen hedefleri gerçekleştirebilmesi için hangi işlemleri yapabilmesi gerekmektedir?

Açıklanan projelerle ilgili olarak diğer öğrencilerin ve rehber öğretmenlerin fikirleri alınır.

Sonraki Haftaya Hazırlık: Sonraki hafta için belirlenen işlemlerin gerçekleştirilebilmesi için olası çözümlerin fikir üretme aşamasında belirlenmesi ve uygun çözümlerin seçilmesi gerekir. Fikir üretme aşamasında, empati ve tanımlama adımında toplanan bilgiler bir bütün hâline getirilir ve çözüm önerisi taslak olarak ortaya konur. Süreçte ortaya çıkabilecek problemleri çözebilmek için nelerin yapılması gerektiği ile ilgili fikirler sunulur. Çözümün başarılı olabilmesi için ihtiyaçların iyi tanımlanmış olması ve bu ihtiyaçların giderilmesi için uygun çözüm önerilerinin sunulması önemlidir.

Rehber Öğretmen İçin Tanım: Fikir üretme aşamasında çok sayıda fikir ve çözüm üretmek önemlidir. Bu çalışma "'Beyin Fırtınası" tekniğiyle gerçekleştirebilir. Ayrıca problem alanının genişletilerek olaylara dışarıdan bakılması önerilir. Böylece sorunu incelemek ve çözüm bulmak üzere farklı yöntemler aranmaya başlanır.

4. FİKİR ÜRETME

Etkinliğin Amacı: Öğrenciler çevrelerinden veya günlük hayatlarından uygun bir problem belirleyip bu probleme kendi tasarladıkları bir robot ile çözüm üretirler.

Açıklama: Çözümün başarılı olabilmesi için ihtiyaçların iyi tanımlanmış olması ve bu ihtiyaçların giderilmesi için uygun çözüm önerilerinin sunulması önemlidir. Bu hafta, empati ve tanımlama adımında toplanan bilgiler bir bütün hâline getirilir ve çözüm önerisi taslak olarak ortaya konur. Farklı çözümlerin önerilmesi gereklidir. Bunlar çok detaylı ve tamamlanmış fikirler olmak zorunda değildir fakat öneriler gerçekçi ve üretilmeye uygun olmalıdır.

Gruplardan belirledikleri robot projesi için yaptıkları fikir üretme çalışmasının sonucunu sınıfta açıklamaları istenir. Açıklamaların aşağıdaki bilgileri içermesi gerektiği belirtilir. Bundan önce gruplara hazırlanma, gözden geçirme ve tekrar etme için 10 dakika süre verilir.

- Problem cümlesi: Robotun ne yapması isteniyor?
- Robotun kullanılması planlanan ortamdaki ihtiyaçlar nelerdir?
- Robotun kullanılması planlanan ortamdaki beklentiler nelerdir?
- Hedeflerin gerçekleştirilmesi için yapılması gerekenler nelerdir?
- Robotun belirlenen hedefleri gerçekleştirebilmesi için tasarımı nasıl olmalıdır?
- Robotun belirlenen hedefleri gerçekleştirebilmesi için hangi işlemleri yapabilmesi gerekir?
- Problem için üretilen alternatif tasarım ve programlama çözümleri nelerdir?
- Tasarım ve programlama çözümünün seçim süreci nasıl gerçekleştirilmiştir?
- Seçilen programlama çözümünün şematik veya maddeler hâlinde gösterimi (algoritması) nasıldır?

Açıklanan projelerle ilgili olarak diğer öğrencilerin ve rehber öğretmenlerin fikirleri alınır.

Sonraki Haftaya Hazırlık: Gruplara sonraki hafta, üretilen fikirlere bağlı olarak robotun tasarlanıp programlanması aşamasına geçileceği söylenir.

5. GELİŞTİRME – TEST ETME

Etkinliğin Amacı: Öğrenciler tanımladıkları problemi çözmek için kendi tasarladıkları bir robotu geliştirir ve test ederler.

Prototip Geliştirme: Grupların belirledikleri robot projesi için yaptıkları fikir üretme çalışması sonucunda üretilmesine karar verdikleri robotu oluşturup programa başlamaları gerekir.

Gruplardan, fikir üretme aşamasında belirledikleri çözümleri etkinlik sonuna kadar hızlıca uygulamaları, robot tasarımı ve programı geliştirmeleri istenir. Çözüm için geliştirilen robotlarda ve programlarda bazı aksaklıkların/problemlerin olması olağandır. Planlanan robotun ilk denemede oluşturulması ve geliştirilen programın sorunsuz çalışması çok karşılaşılan bir durum değildir. Bu tür sorunların ortaya çıkmasının olağan olduğu rehber öğretmen tarafından öğrencilere söylenmeli ve/veya hissettirilmelidir. Ortaya çıkan sorunların çözümüne yönelik bir plan hazırlanıp uygulanmalıdır. Çözüm için gruplara aşağıdakilere benzer bir süreç önerilebilir:

i. Problemler maddeler hâlinde listelenmeli (beklenen robot davranışı ile gerçekleşen arasındaki farklar),
ii. Problemler arasındaki ilişkiler belirlenmeli,
iii. Öncelik sırası belirlenmeli,
iv. Öncelik sırasına göre çözümler geliştirilmeli (Çözüm için hipotezler oluşturulmalı),
v. Geliştirilen çözümler (hipotezler) denenmeli ve
vi. Çözümlerin her koşulda istenen şekilde çalışması için gerekli düzenlemeler yapılmalıdır.

Bütün grupların, hazırladıkları prototipleri (tam olarak çalışmasa bile) rehber öğretmene göstermesi sağlanır. Böylece rehber öğretmenin, geliştirilen ürünü test etmesi/görmesi, çeşitli çözüm önerileri getirmesi ve/veya gözden kaçan noktaları ortaya çıkarması sağlanabilir.

Test Etme: Ortaya çıkan son ürünü hedef kitle ile/hedef ortamda değerlendirmek geliştirilen çözümlerin kusursuz bir şekilde çalışması için önemlidir. Hedef kitleye sınıfta ulaşmak her zaman mümkün olmayabilir. Bu nedenle grupların, ortaya çıkardıkları ürünleri diğer öğrencilerin kendi başlarına deneyimlemelerine izin vermesi, gözden kaçan unsurların ortaya çıkarılmasına yardımcı olabilir.

6. SUNUM

Projenin son aşamasında gruplara projelerini sunmaları konusunda yardımcı olunmalıdır. Gruplar projelerini bir senaryo çerçevesinde teatral olarak sunabilir.

ROBOTİK VE KODLAMA

LİSE

DENEYAP TÜRKİYE