

HAYDİ, VERİ TOPLAMAYA!

6. SINIF - 19. HAFTA

DERS NOTU

MATEMATİKTEN BİLGİSAYARA EĞLENCİLİ BİR KÖPRÜ

Bilgisayarlardaki programlar matematikteki formüllerden, yaklaşımlardan ve hesaplamalardan yararlanırlar. İlk bilgisayar çok basit matematiksel işlemleri yapabilen bir abaküstür.

İlk modern bilgisayar **ENIAC**'dır. Elektrikle çalışan ve veri işleme kapasitesine sahip ilk modern bilgisayardır. ENIAC bir ev büyüklüğündedir (167 metrekare) ve ağırlığı 30 tondur.

Matematikten bilgisayara uzanan süreçte pek çok bilim insanından bahsedebiliriz:



- Bu bilim insanlarından biri **Ada Lovelace**'dir. Ada Lovelace kadınların çalışma hayatında bulunmadığı 1830'lu yıllarda bilgisayar programcısı olarak çalışmalar yapmış ve gelecek kuşaklara örnek olmuştur.

- **Blaise Pascal** tarafından icat edilen Pascaline, 17. yüzyıl boyunca çalışan ilk ve tek mekanik hesap makinesiydi.



- Mantık üzerine çalışmalar yapan **Gottfried Leibniz** ve **Charles Babbage**,
- Matematiksel dil yapısının öncüsü **George Boole**,
- Kuramsal makineler tasarlayan **Alan Turing**,
- Modern bilgisayarlara katkı sağlayan **John von Neumann**,
- Kuramsal programlama çalışmaları yapan **Dana Scott**.

Bilişim Teknolojileri ve Yazılım Dersi'nin en önemli materyali sizin de tahmin edebileceğiniz gibi bilgisayardır. Bilgisayar için Türkiye'de önce "**Kompüter**" kelimesi kullanılmış. Daha sonra bu kavrama Türkçe bir karşılık bulma gereksinimi sonucu çalışmalara başlanmış ve "**Bilgisayar**" kavramı ortaya çıkmış.



Türkiye'nin İlk Bilgisayarı:

Türkiye'de kullanılan ilk bilgisayar, 1960 yılında kullanılan "**(Data Processing Machine)**" adlı bilgisayardır. Bu bilgisayar, yol yapımında gereken hesaplamaları daha hızlı yapabilmek için Karayolları Genel Müdürlüğü tarafından alınmış ve 12 yıl kullanılmıştır.

BİLGİSAYARLAR NASIL ÇALIŞIR?

İkili Sistem: Bilgisayarda veriler 0 ve 1'ler halinde iletilir ve depolanır. Bu 0 ve 1'lerden her birine "bit" adı verilir.

Veriler, sinyaller halinde bir yere gönderilirken, gelen sinyalin voltajına bakılır. Voltaj varsa 1, yoksa 0 kabul edilir.



Bilgisayarın bir bilgiyi anlayabilmesi ve işlem yapabilmesi için, mutlaka ikilik sisteme çevrilmesi gerekmektedir. Örneğin klavyeden bir harfe basıldığında, o harf için 0 ve 1'lerden oluşan bir kod bilgisayara gönderilir. Bu sayede gelen bilgi bilgisayar tarafından anlaşılabilir.

Makine Dili: Geliştirilen ilk programlama dilidir. Bu dilde yazılan tüm komutlar 0 ve 1 lerden oluşur. Makine dili örneği: Bu program ekrana "Hello world" yazısını yazar.

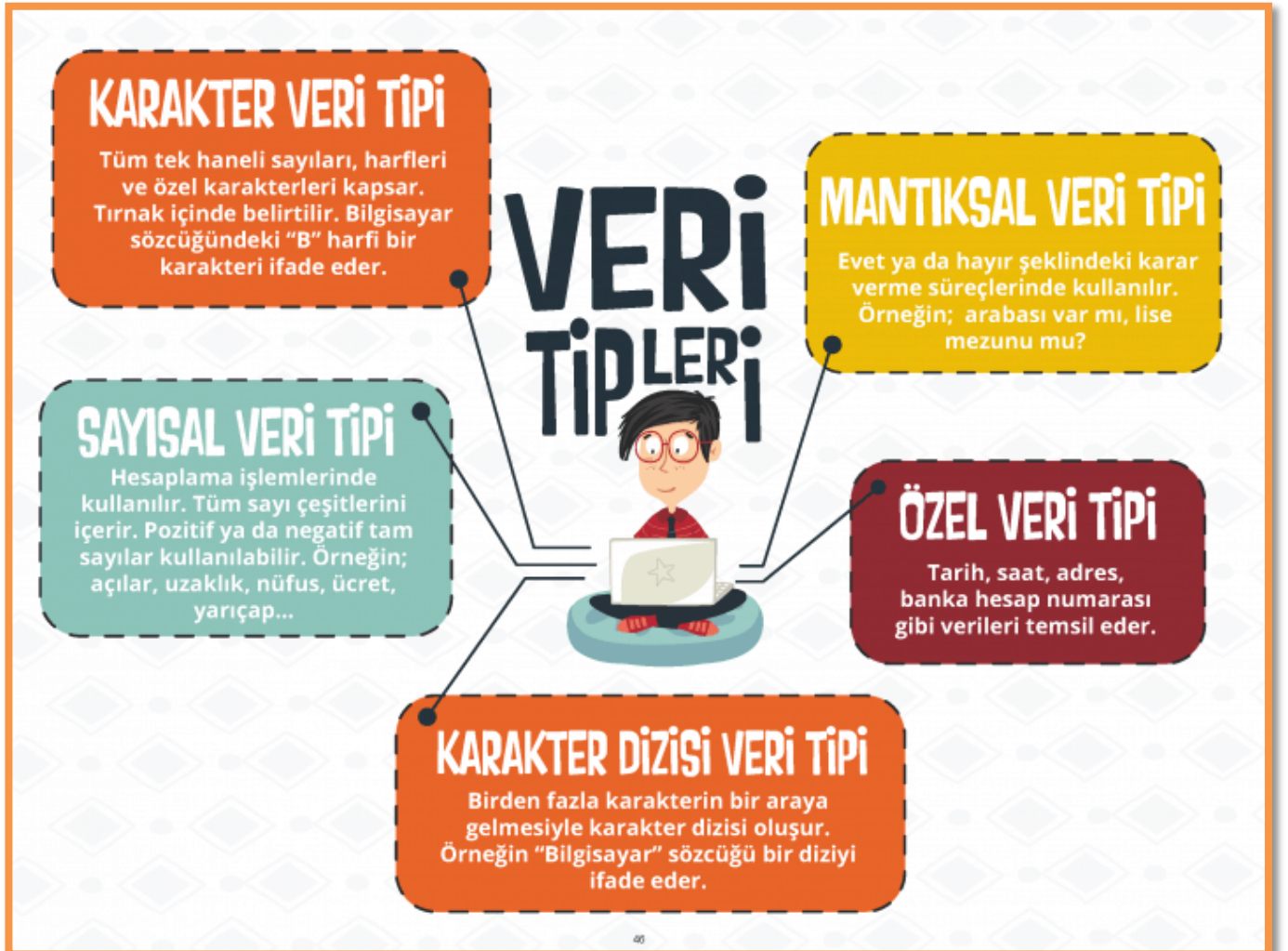


1011101101000101	0000000110111001	0000110100000000	1011010000001110
1000101000000111	0100001111001101	0001000011100010	1111100111001101
0010000001001000	0110010101101100	0110110001101111	0010110000100000
0101011101101111	0111001001101100	0110010000100001	

VERİ TİPLERİ

Veri: Kavram veya komutların, iletişim, yorum ve işlem için elverişli biçimli gösterimi.

Bilgi: Araştırma, gözlem ve benzeri öğrenme yolları ile elde edilen gerçektir.



SABİT Mİ? DEĞİŞKEN Mİ?

6. SINIF - 20. HAFTA

DERS NOTU

SABİT ve DEĞİŞKEN

Sabit: İlk biçimiyle kalan, değişmeyen ifade ya da nesnelerdir.

Değişken: İlk biçimiyle kalmayıp yeni değerler ya da biçimler alabilen ifade ya da nesnelerdir.

Örnek: 100 °C gibi sayısal değerler **sabittir**. Sıcaklık gibi farklı değerler alabilen ifadeler ise **değişken** olarak adlandırılır.

AKIŞ ŞEMALARI

Algoritma: İşte açık ve net ifadelerle problemin adım adım çözümünü gösteren bu taslağa algoritma adı verilir. Programlamanın ilk adımı algoritma oluşturmaktır.

Akış Şemaları: Algoritmaları ve işlemleri birbirine oklarla bağlı değişik tiplerdeki kutular içerisinde gösteren yaygın bir şema tipidir. Akış şemaları çeşitli alanlardaki işlem ve uygulamaların yönetilmesi, belgelendirilmesi, tasarlanması ve çözümlenmesinde kullanılır. Şemada yer alan her şeklin bir kullanım amacı vardır.

➤ **Elips:** Başla ve Bitir adımları için kullanılır. Akış şemasının başlangıç ve bitiş noktasında yer alır.



➤ **Paralel Kenar:** Giriş ya da Çıkış işlemleri için kullanılır. Örneğin; klavyeden bir sayı girilmesi istenmesi veya ekrana işlem sonucunun yazdırılması gibi.

➤ **Dikdörtgen:** Hesaplama ya da Değişkene Değer Atama işlemleri için kullanılır. Örneğin; iki sayıyı topla veya girilen ilk sayıyı A olarak kabul et.



➤ **Eşkenar Dörtgen:** Karşılaştırma ya da Karar Verme işlemleri için kullanılır. Örneğin; girilen sayı 5'ten büyük mü?

➤ **Yön Okları:** Akış şemasının ilerleme yönünü gösterir.



Karşılaştırma Akış Şeması



Sıralama Akış Şeması



Toplama Akış Şeması



Yazdırma Akış Şeması



BÖL, PARÇALA, ÇÖZ

6. SINIF - 21. HAFTA

DERS NOTU

PROBLEM ve ALGORİTMA

Problem Nedir? Problem, çözülmesi gereken sorun ya da aşılması gereken engel anlamına gelir. Günlük yaşamda karşılaştığımız problemleri bilerek veya farkında olmadan adım adım çözmeye çalışırız.

Örnek: Yazı yazarken kaleminizin ucu kırıldığında şu adımları takip ederek bu sorunu çözersiniz.

1. Kalem tıraşı çıkar.
2. Kalemi al.
3. Çöp kovasının yanına git.
4. Kalemin ucunu aç.
5. Sırana geri dön.
6. Yazmaya devam et.



Basit Problem: Basit adımlardan oluşan ve her koşulda aynı yönde ilerleyerek çözülebilen problemlerdir.

Örnek: Araba yıkamak, kek yapmak, evden okula gelmek vb.

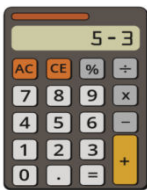
Karmaşık Problem: Duruma özgü ve şartlara göre değişebilen çözüm adımlarından oluşan ve alt problemlere ayrılabilen problem türleridir. Çözüm için takım çalışması gerekebilir.

Örnek: Araba lastiği değiştirmek, pazar alışverişi yapmak, okulda başarılı olmak vb.

Algoritma: Bir problemi çözmek ya da belirli bir amaca ulaşmak, bir işi gerçekleştirmek için tasarladığımız yola algoritma denir. Algoritma yardımıyla bir işi adım adım gerçekleştirebiliriz. Aslında algoritmalar yaşamımızın bir parçasıdır. Pek çok işimizi farkında olalım ya da olmayalım algoritma yardımıyla yaparız. Bu işlerin tümünde, algoritmalarındaki gibi belirli bir sıra bulunur.

Örnek: Ayran yapıp bardağa dolduralım.

- | | |
|----------------------------------|-------------------------|
| Adım 1: Başla | Adım 5: Tuz koy. |
| Adım 2: Yoğurdu kaba koy. | Adım 6: Bardağa |
| Adım 3: Su ekle. | doldur. |
| Adım 4: Çırp. | Adım 7: Bitir. |



Fonksiyon: Bağımsız değişkenler ile bağımlı değişken arasındaki ilişkinin matematiksel ifadesidir.

Hesap makinesi üzerinde yer alan tuşların her birisi birer fonksiyon ifade eder. Bizler de bu dönem öğreneceğimiz Scratch programı aracılığıyla bir hesap makinesi yapabiliriz.

PROBLEM ÇÖZMEK BENİM İŞİM!

6. SINIF - 22. HAFTA

DERS NOTU

HAYATIM PROBLEM, ÇÖZÜMÜM ALGORİTMA



Algoritma kullanarak;

- Problemleri daha hızlı ve sistematik olarak çözeriz.
- Problem çözme sürecini takip ederiz ve nerede hata yapıldığını görebiliriz.
- Tüm olasılıkları gözden geçirebiliriz.
- Hatalı işlem yapma olasılığımızı azaltırız.
- Olası hatalarımızı düzeltebiliriz.
- Çözüme ulaşmak için farklı yolları deneyebiliriz.
- Problemin çözümü için harcayacağımız süreyi kısaltırız.

Algotirma'yı Kim Buldu?

Algoritma matematikte sayılarla yapılan her türlü hesaplamanın sistematik metoduna verilen genel isimdir. Algoritmanın kurucusu dokuzuncu yüzyıl başlarında yaşayan, matematik gökbilim ve coğrafya alanlarında çalışmış bilim adamı Hârizmî'dir. Matematikçiler için temel olan Kitab-ül Muhtasar fi Hesab-il Cebri ve'l-Mukabele adlı eseri meşhurdur.



FARKLI YOLLARDAN AYNI ÇÖZÜME

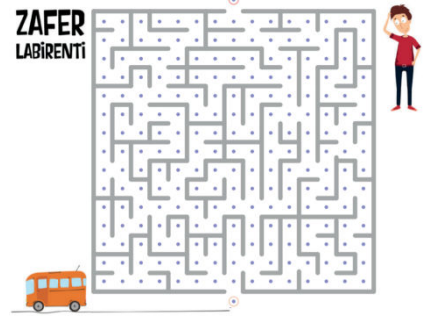
6. SINIF - 23. HAFTA

DERS NOTU

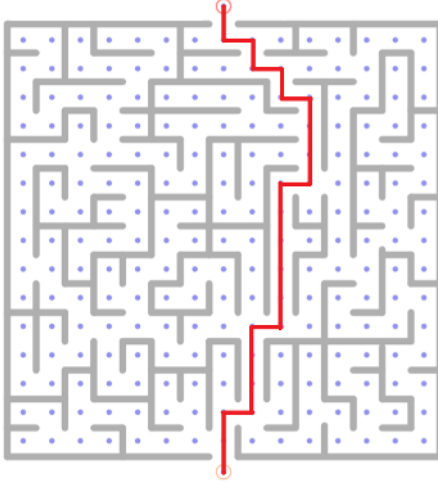
FARKLI YOLLARDAN AYNI ÇÖZÜME

Karşılaştığımız bir sorunu farklı yollardan çözebiliriz.

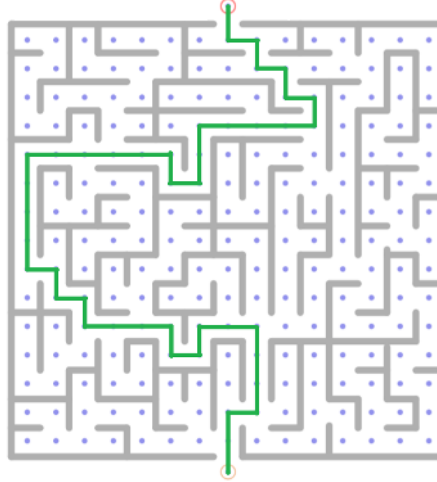
Soru: Yandaki labirentte Zafer'i okulun çıkış kapısında bulunan servis aracına ulaştıran üç yol bulunuyor. Öncelikle bu yolları bulacağız. İkinci görevimiz ise bu yolları uzunluklarına göre sıralamak. En kısa yol Zafer'i servis aracına en çabuk ulaştıracak olan yol olacaktır. Bu yolları çizerken ya da uzunluklarına göre sıralarken birim noktalarını kullanabiliriz.



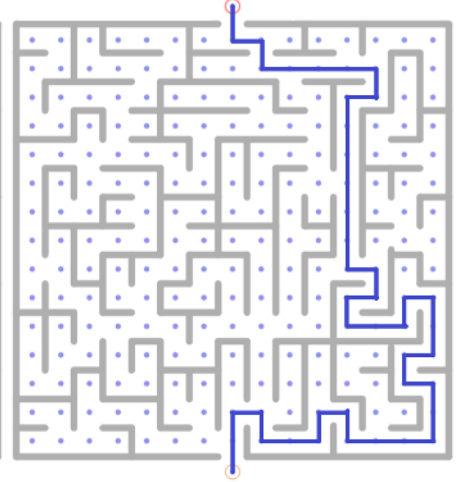
Çözüm: 1 (22 Adım)



Çözüm: 2 (42 Adım)



Çözüm: 3 (42 Adım)



EN KISA YOL

Soru: Aşağıda A'dan B'ye giderken kullanacağınız 3 farklı yol yönergesi bulunmaktadır. Her bir yönergenin altında bulunan algoritmaları inceledik. Algoritmalarından yararlanarak en kısa ve en uzun yolu belirledik.

Yanıt:

1. A'dan B'ye bir yeşil bir maviye uğrayarak git.

a) 2 birim doğuya, 1 birim güneye, 3 birim doğuya, 1 birim güneye, 3 birim batıya, 1 birim güneye, 3 birim doğuya git. **(Uzun)**

b) 3 birim güneye, 2 birim doğuya, 2 birim kuzeye, 3 birim doğuya, 2 birim güneye git. **(Kısa)**

c) 2 birim doğuya, 1 birim güneye, 3 birim doğuya, 3 birim batıya, 2 birim güneye, 2 birim doğuya git.

2. A'dan B'ye iki kırmızıya uğrayarak git.

a) 3 birim doğuya, 1 birim güneye, 2 birim batıya, 1 birim güneye, 4 birim doğuya, 1 birim güneye git.

b) 1 birim güneye, 1 birim doğuya, 1 birim kuzeye, 2 birim doğuya, 3 birim güneye, 2 birim doğuya git. **(Kısa)**

c) 3 birim doğuya, 3 birim batıya, 1 birim güneye, 1 birim doğuya, 1 birim güneye, 4 birim doğuya, 1 birim güneye git.

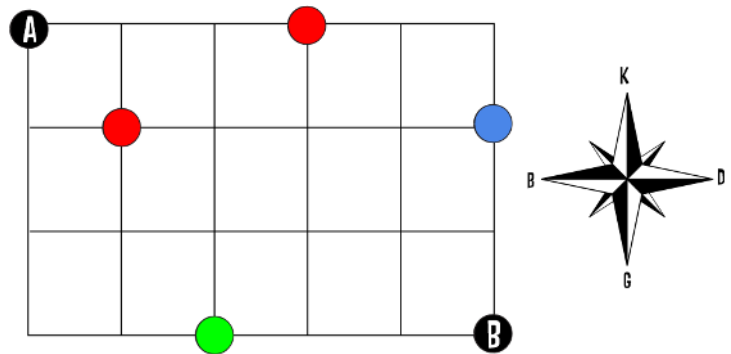
(Uzun)

3. B'den A'ya önce yeşile sonra maviye uğrayarak git.

a) 3 birim güneye, 2 birim doğuya, 2 birim kuzeye, 3 birim doğuya, 2 birim güneye git. **(Kısa)**

b) 2 birim doğuya, 3 birim güneye, 2 birim doğuya, 3 birim kuzeye, 1 birim doğuya, 3 birim güneye git.

c) 2 birim doğuya, 3 birim güneye, 1 birim doğuya, 2 birim kuzeye, 2 birim doğuya, 2 birim batıya, 2 birim güneye, 2 birim doğuya git. **(Uzun)**



AYIKLA PİRİNCİN TAŞINI

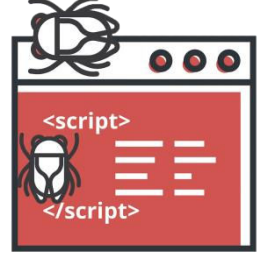
6. SINIF - 24. HAFTA

DERS NOTU

HATA AYIKLAMA

Hata Ayıklama: Programlama, karmaşık bir süreçtir ve programcılar programlamada hata (bug) yapabilirler. Programlama hatalarını bulma ve düzeltme işlemine hata ayıklama (debugging) denilir.

Hata ayıklama, bir bilgisayar programında ya da elektronik donanımda hataları bulmanın, kusurları düzeltmenin ve bütün bunların sayılarını en aza indirmenin yoludur. Hata ayıklama işlemi sonunda, bu işi uyguladığımız yazılımın ya da donanımın istediğimiz ve beklenen şekilde çalışması gerekir.



FARKI FARK EDELİM

Senaryo 1

Yağmur okulun basketbol takımında oynamaktadır. Haftada üç gün (pazartesi, çarşamba ve cuma) antrenmana gitmektedir. Antrenman günlerinde antrenman çantasını yanına alması gerekmektedir.

Antrenmanı 15.30'da okul çıkışı yapmaktadırlar ve antrenman çantasını unutursa eve dönüp alması için zamanı kalmamaktadır. Ders saatlerinde cep telefonu kullanamadığı için antrenman çantasını antrenman günleri sabahtan yanına alması gerekmektedir.

Yağmur süreci sağlıklı yürütebilmek için telefonunun alarm sistemini kurmuştur. Yağmur'un algoritmasında her şeye karşın aksayan bir şeyler olmuştur.

Algoritmasını düzenleyebilmesi için Yağmur'a yardımcı olur musunuz?

Hatalı Algoritma:

- 1- Başla
- 2- Antrenman çantası al.
- 3- Hayır ise 8. adıma git.
- 4- Evet ise 6. adıma git.
- 5- Antrenman çantası yanına almak için saat 07.00'ye hatırlatma kur.
- 6- Pazartesi, çarşamba veya cuma mı?
- 7- Okula git.
- 8- Bugün günlerden ne?
- 9- Bitir.

Doğru Algoritma:

- 1) Başla.
- 2) Bugün günlerden ne?
- 3) Pazartesi, çarşamba veya cuma mı?
- 4) Evet ise 6. adıma git.
- 5) Hayır ise 8. adıma git.
- 6) Antrenman çantası yanına almak için saat 07.00'de hatırlatma kur.
- 7) Antrenman çantası al.
- 8) Okula git.
- 9) Bitir.

Senaryo 2

Ercan okuldan çıkınca çalışmak ve ödevlerini yapmak için dayısının iş yerine gitmektedir. Dayısının iş yeri çarşıdaki Günay Han'ın 7. katındadır. Handa bir asansör yer almaktadır ama kat düğmelerinde sorun bulunmaktadır. Asansörde 8 katlı düğme sistemi olmasına karşın asansör yalnızca 5. kata kadar çıkmaktadır. Yani siz gitmediği katların düğmesine bassanız bile asansör kabini sizi 5. kata götürmektedir.

Ayrıca, çocukların yalnız başına binmelerini engelleyebilmek için de min. 30 kg. yük sınırı vardır.

Asansörü çalıştıran algoritmayı kurarken yapılan hatayı nasıl giderebilirsiniz? Yardımcı olur musunuz?

Hatalı Algoritma:

- 1- Başla
- 2- "Tek başınıza asansöre binemezsiniz" yaz.
- 3- Kat değeri 4'ten büyük ise 7. adıma git.
- 4- Kullanıcı ağırlığını, ağırlık değişkenine ata.
- 5- Kat değerine atanan kata çık ve 9. adıma git.
- 6- Kat değişkenine, basılan kat numarasını ata.
- 7- 5. kata çık ve 9. adıma git.
- 8- Ağırlık 30'dan küçük ise 8. adıma git.
- 9- Bitir

Doğru Algoritma:


- 1) Başla.
- 2) Kullanıcı ağırlığını ağırlık değişkenine ata.
- 3) Ağırlık 30'dan küçük ise 8. adıma git.
- 4) Kat değişkenine, basılan kat numarasını ata.
- 5) Kat değeri 4'ten büyük ise 7. adıma git.
- 6) Kat değerine atanan kata çık ve 9. adıma git.
- 7) 5. kata çık ve 9. adıma git.
- 8) "Tek başınıza asansöre binemezsiniz" yaz.
- 9) Bitir.

AYIKLA PİRİNCİN TAŞINI

AYIKLA PİRİNCİN TAŞINI


Aşağıda bulmacalarda piyonu varış noktasına ulaştırmaya çalışacağız. Bunun için sırası karışık olarak verilmiş algoritmaları doğru bir şekilde sıralaman ve bu sıralamayı bulmacaların altındaki boş sütuna yazman gerekiyor. Piyon tek seferde yalnızca bir çukurun üzerinden atlayabilir. X'ler aşılamayan engelleri gösterir.

: Piyon : Çukur : Engel : Varış Noktası

X	X	X			X			X	
X	X		X	X		X		X	
X	X	X		X	X	X	X		X
X								X	X
X				○	X	X	X		X
X					X		X	X	
X		X	X	○	X	X	X		X
X	X					X	X	X	
○	○	○			X	X			X
					X	X		X	

Zıpla
4 adım ilerle
2 adım ilerle
Zıpla
2 adım ilerle
Sağa dön
Sola dön

4 adım ilerle
Sola dön
2 adım ilerle
Zıpla
Zıpla
Sağa dön
2 adım ilerle

X		X			X			X	
X	X		X	X		X		X	
X	X	X		X	X	X	X		X
				X				X	
X	X			○	X	X	X		X
X	X		○		X		X	X	
		X	X	○	X	X			X
X	X					X	X	X	
○		○			X				X
			X	X		X		X	

2 adım ilerle
Sola dön
Sola dön
Sağa dön
2 adım ilerle
Zıpla
Sola dön
Zıpla
2 adım ilerle
Zıpla
Sağa dön

2 adım ilerle
Sola dön
Zıpla
Sağa dön
2 adım ilerle
Sola dön
Zıpla
Sola dön
Zıpla
Sağa dön
2 adım ilerle

X	X	X	○		X			X	
X	X		X	X		X		X	
X	X	X				○			X
X	○			X			X	X	
X			○	○	X	X	X		X
X					X		X	X	
X		X	X	○	X	X	X		X
X						X	X	X	
○	○	○			X	X			X
					X	X		X	



1 adım ilerle
1 adım ilerle
Zıpla
3 adım ilerle
Sağa dön
Sola dön
Sağa dön
Sola dön
1 adım ilerle
1 adım ilerle
1 adım ilerle
Sola dön

1 adım ilerle
Sola dön
Zıpla
3 adım ilerle
Sağa dön
1 adım ilerle
Sola dön
1 adım ilerle
Sağa dön
1 adım ilerle
Sola dön
1 adım ilerle

X	X	X			X			X	
X	X		X	X		X		X	
X	X	X		X	X	X	X		X
X			X			○		X	
X		○			X	X			X
X		X			X	X		X	
X		X	X	○	X	X	○		X
X	X					○		X	
○	○	○			X	X			X
					X	X		X	

4 adım ilerle
1 adım ilerle
2 adım ilerle
Zıpla
Sola dön
Sola dön
1 adım ilerle
Sola dön
Zıpla
2 adım ilerle

4 adım ilerle
Sola dön
2 adım ilerle
Zıpla
1 adım ilerle
Sola dön
1 adım ilerle
Zıpla
Sola dön
2 adım ilerle

X	X	X			X			X	
X	X		X	X		X		X	
X	X	X		X	X	X	X		X
X								X	
X				○	X	X	○		X
X					X			X	
X		X	X	○	X	X			X
X	X					○		X	
○	○	○			X	X			
			X		X	X		X	

2 adım ilerle
Sola dön
Sola dön
Sağa dön
3 adım ilerle
2 adım ilerle
Sağa dön
1 adım ilerle
Zıpla
Zıpla

2 adım ilerle
Sola dön
Zıpla
Sağa dön
3 adım ilerle
Zıpla
Sağa dön
1 adım ilerle
Sola dön
2 adım ilerle

BENZER SORUN BENZER ÇÖZÜM

6. SINIF - 25. HAFTA

DERS NOTU

SORUNLAR VE ÇÖZÜMLER



Genelleme: Bir probleme ait çözümü benzer özelliklere sahip farklı problemlerin çözümünde kullanılabilecek ortak bir yöntem bulmak.

Günlük yaşamımızda çeşitli problemlerle karşılaşırız. Bu problemlerin bazılarını yakından incelediğimizde aslında çözüm yollarının çok da zor olmadığını görürüz. Dahası bu basit problemler için bulduğumuz çözümleri benzer özellikler taşıyan ve daha karmaşık gözükten problemler için de kullanabiliriz.

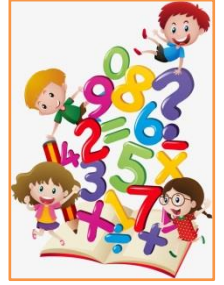
SAYILARIN ŞİFRESİ

Kalıp Cümle: “ İlk/Önceki sayının ... katının ... fazlası/eksiği.”

Bu gördüğünüz cümle bir sayı örüntüsü oluşturmak üzere hazırlanmış bir kural cümlesidir. Şimdi bu kural cümlesini kullanarak çeşitli sayılar elde edeceğiz. Ama sayılarımızı oluşturmadan önce başlangıç sayısına ve kuralına karar vermeliyim.

Başlangıç sayım 2, kuralım da “ önceki sayının 3 katının 2 eksiği.” olsun. Bu durumda sayı dizimin ilk beş sayısı şu şekilde olacaktır:

Örüntü: 2 -> 4 -> 10 -> 28 -> 82



Örnek Soru: Sayı dizisinin ilk 5 terimi aşağıdaki gibiyse kuralımız nedir?

6 -> 11 -> 21 -> 41 -> 81

Cevap: İlk terimimiz 6 olduğuna göre ikinci terim olan 11 ile ilişkisini çözdüğümüzde cevabı bulacağız. 11 sayısı 6'nın iki katının 1 eksiğidir. Bu durumda kuralımız; Sayının 2 katının 1 eksiği.

Mandala: Kökeni çok eski zamana dayanan bir çizim ve boyama tekniğidir. Mandala doğu dillerinde ‘enerjiyi tutan kap’ anlamına gelmektedir. Bizim kültürümüzde ise mandala örneklerine dantel desenlerinde ve çini boyama sanatında rastlayabiliriz.

Mandala çizimi daire şeklinde yapılır. Bir dairenin merkezinden başlayarak kullanmak istediğiniz çeşitli şekilleri simetrik ve uyumlu olarak, tekrarlı bir biçimde çizerek mandala desenleri oluşturabilirsiniz.



SCRATCH 3.0 DERS NOTLARI

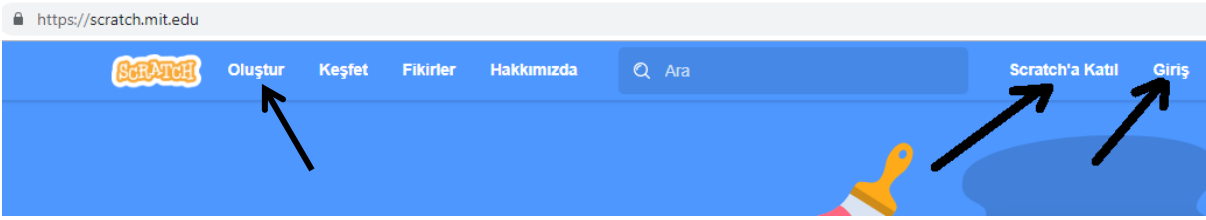


SCRATCH NEDİR?

Scratch programı eğlenceli bir ortamda resim, ses, müzik gibi çeşitli medya araçlarını bir araya getirebileceğimiz, kendi animasyonlarımızı, bilgisayar oyunlarımızı tasarlayabileceğimiz ya da interaktif hikayeler anlatabileceğimiz ve paylaşılabileceğimiz bir grafik programlama dilidir.

Scratch web sayfası, Scratch' i öğrenebilmemiz için birçok kaynak içermektedir. Web sayfasını ziyaret etmek isterseniz 'http://scratch.mit.edu' adresinden Scratch ana sayfasına ulaşabilirsiniz.

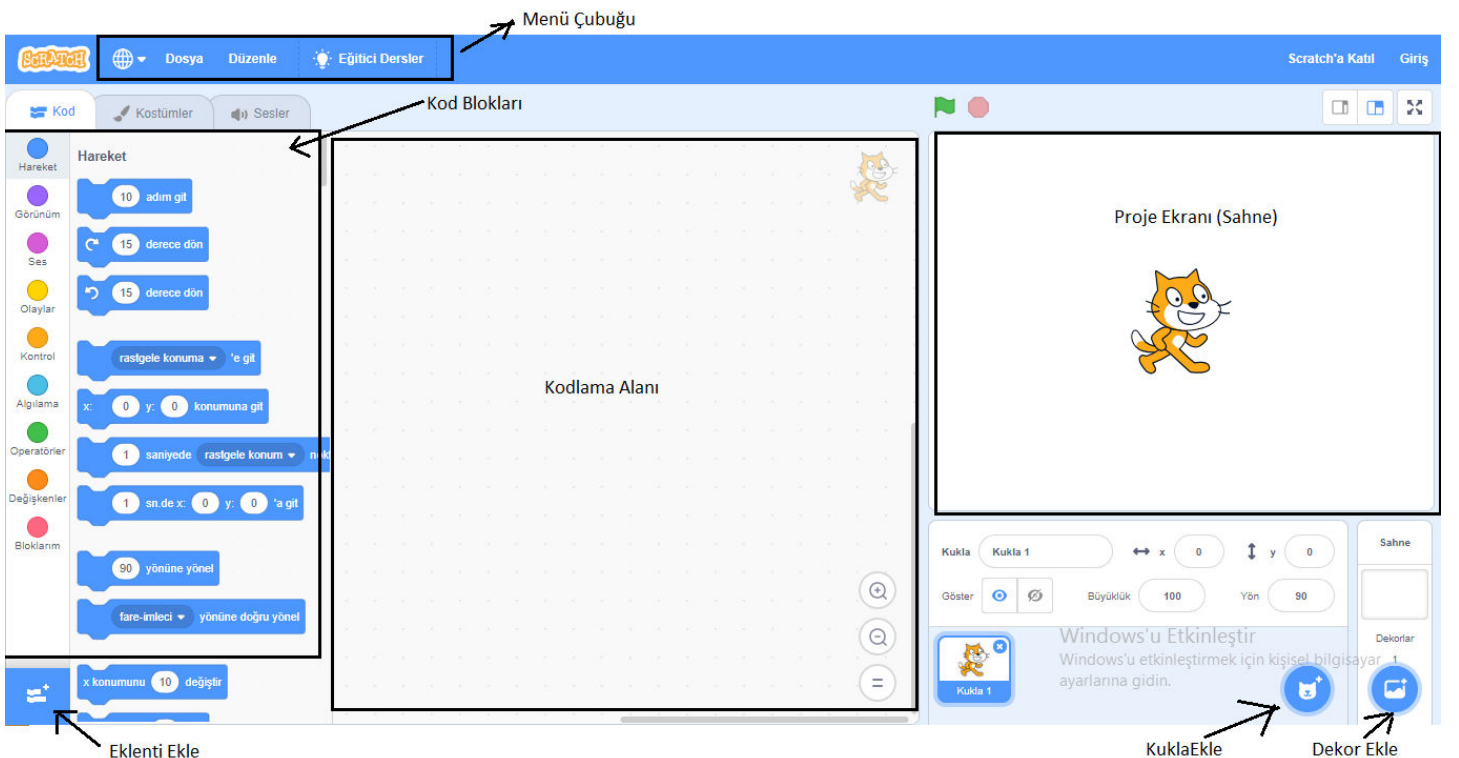
ÜYE OLMA VE SİTEDE ÇALIŞMA



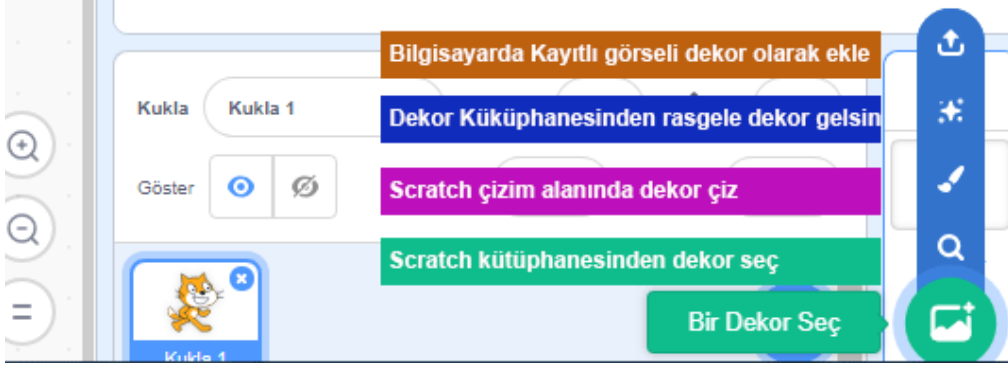
Scratch.mit.edu adresi açıldığında sağ üstteki **Scratch'a Katıl** bağlantısından siteye üye olabilirsiniz. Daha sonraki kullanımlarınızda aynı sayfadaki **Giriş** bağlantısıyla da sisteme kullanıcı adı ve şifrenizle girebilirsiniz. Bu sayfa üzerindeki **Oluştur** bağlantısıyla doğrudan site üzerinde çalışarak programlar geliştirebilirsiniz.

Keşfet bağlantısını kullanarak diğer üyeler tarafından hazırlanan programları hazırlanma aşamalarını ve kullanılan komutları görebilirsiniz.

SCRATCH'IN EKSPANINI TANIYALIM



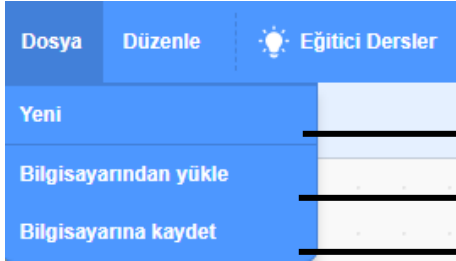
1. **Menü Çubuğu:** Scratch'ın yönetim, ayar komutlarını içerir.
2. **Proje Ekranı (Sahne):** Bizim sahnemizdir. Projemizi çalıştırdığımızda tasarladığımız her şey burada hayat bulur.
3. **Kod Blokları:** Karakterimizi programlamak için kullanabileceğimiz kod bloklarının bulunduğu kısımdır.
4. **Kodlama Alanı:** Blokları sürükleyerek komut dizileri oluşturacağımız alandır.
5. **Kukla Ekle:** Kukla ekleme işlemini yaptığımız bölümdür.
6. **Dekor Ekle:** Sahnemizin arka planına yeni dekor eklememizi sağlayan alandır.



7. **Eklenti Ekleme:** Ekranın sol alt tarafında bulunan “+” işaretine tıkladığınızda karşınıza yeni diziler çıkacaktır. Kalemle tutunda Makey Makey’e Müzikten video algılamaya kadar birçok dizi karşınıza çıkacaktır.

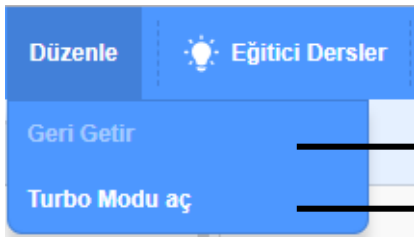
MENÜ ÇUBUĞU

Dosya Menüsü



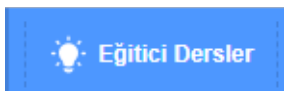
- Projemizi kaydettikten sonra yeni proje oluşturmak için Dosya menüsünden Yeni'ye tıklarız.
- Daha önceden hazırladığımız bir projeyi açmak için tıklarız.
- Buseçenekile oluşturduğumuz projeyi diske kaydedebiliriz.

Düzenle Menüsü



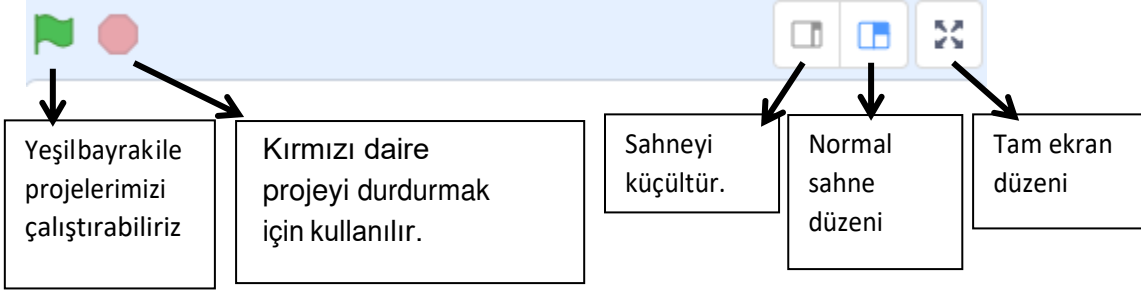
- Sildiğimiz komut bloklarını geri alır.
- Projeyi hızlı çalıştırmak için kullanılır. Örneğin bu modda hareketler hızlanır.

Eğitici Dersler



Farklı örnek sunumlarıyla eğitici bilgiler veren çalışmalara buradan ulaşılabilir.

SAHNE

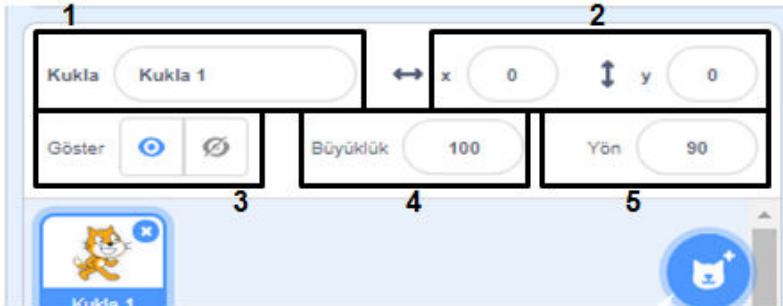


Ekran 480 birim genişlikte ve 360 birim uzunluğundadır. Scratch ekranı aslında bir koordinat düzlemidir. Scratch programı açıldığında karşımıza çıkan kedi karakteri başlangıçta (0,0) noktasındadır

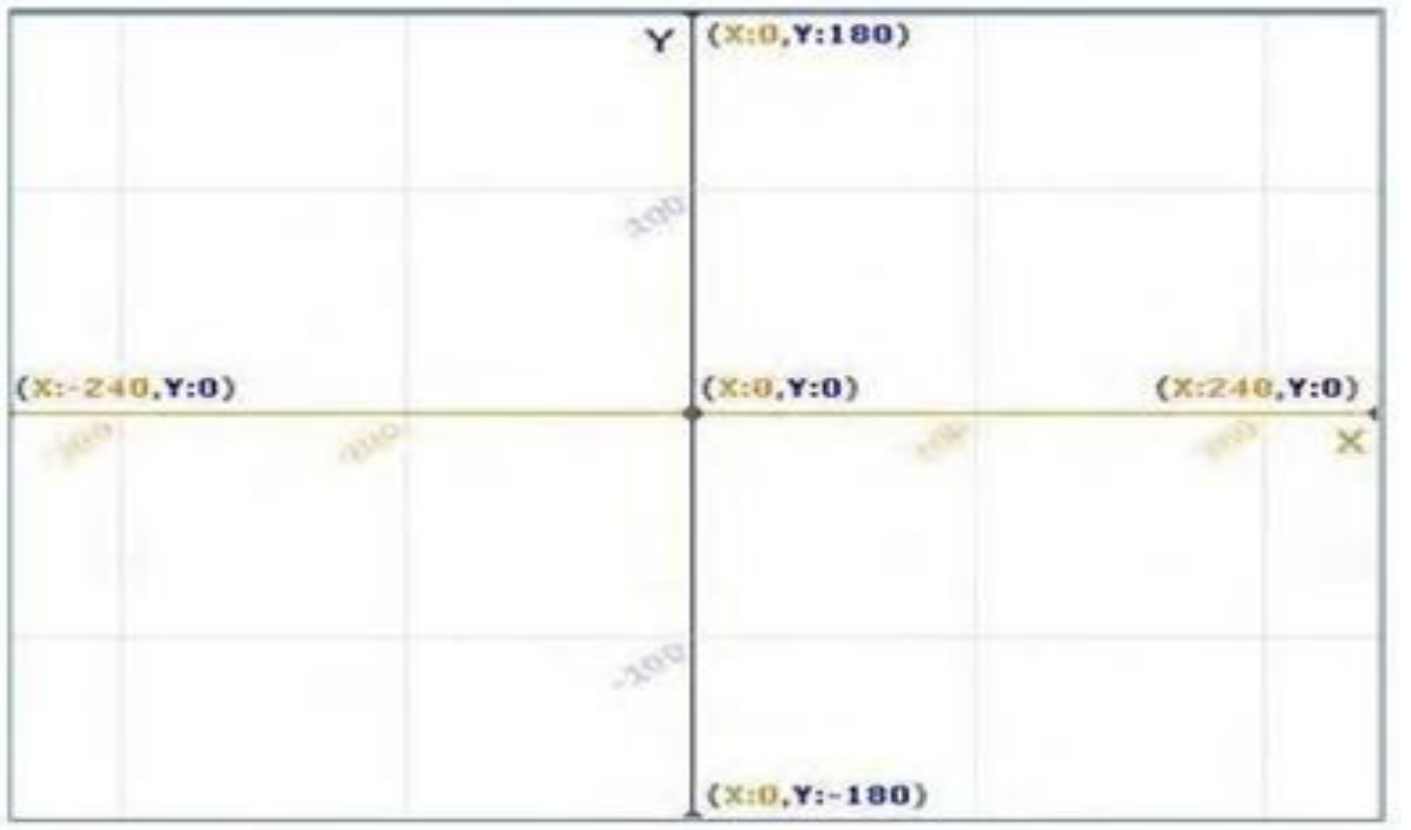
KUKLALAR



Kukla Bilgisi

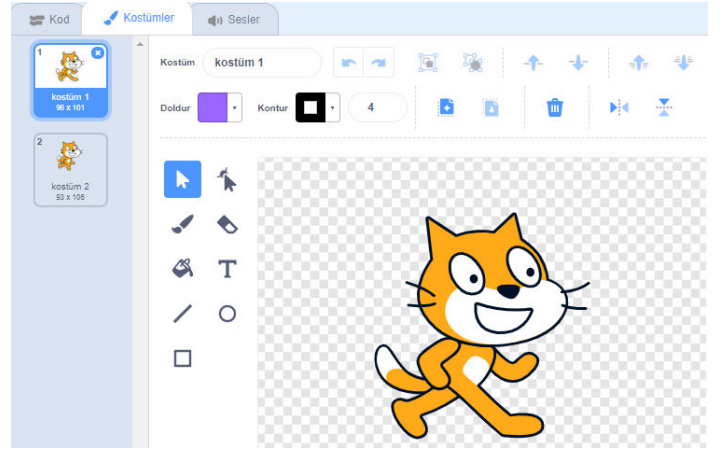


- 1. Kukla adı:** Kuklanın adını değiştirmek için kullanılır.
- 2. Konum:** Kuklanın koordinatlarını(konum) belirtir.
- 3. Göster:** Proje çalıştırıldığında kuklanın ekranda görünüp görünmeyeceğini ayarlar.
- 4. Büyüklük:** Kuklanın boyutunu değiştirmek için kullanılır.
- 5. Yön:** Kuklanın sahnede 360 derece dönmesini sağlar.



KOSTÜMLER

Kuklamızın başka görünümlerinin olmasını istiyorsak kütüphaneden ekleyebilir, kendimiz çizebilir, bilgisayarımızda kayıtlı bir resmi içe aktarabiliriz ya da kamera ile çekebiliriz. Var olan kostümleri kostümün sağ üst köşesindeki 'x' butonuna basarak silebiliriz, seçili kostümü sağdaki düzenleme alanında düzenleyebiliriz. Kostüm üzerinde sağ tık yaparak kopyalayabilir ve silebiliriz.



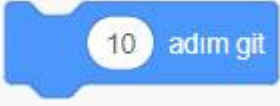
SESLER

Karakter için var olan sesleri görebilmek için bu bölüme tıklarız. Bu bölümden yeni sesler ekleyebilir (sırasıyla kütüphaneden ses ekler, ses kaydeder, bilgisayarda kayıtlı bir sesi ekler), var olan sesi dinleyebilir, düzenle ve etkiler altındaki komutlarla düzenleyebilir veya silebiliriz.

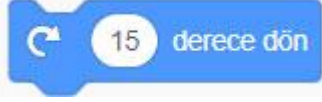


Scratch 3 Hareket Blokları

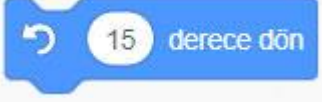
İsminden de anlaşılacağı üzere hareket blokları kuklalarımızı hareket ettirmek için kullandığımız kod bloklarıdır.



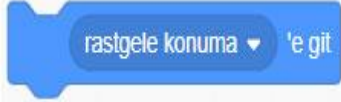
Kuklayı seçili yöne doğru 10 piksel hareket ettirir.



Karakterin istenilen derece kadar saat yönünde dönmesini sağlar.



Karakterin istenilen derece kadar saat yönü tersine dönmesini sağlar.



Kuklayı sahnede rastgele bir x,y konumuna götürür.



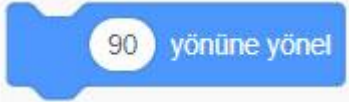
Kuklayı sahnede belirtilen x,y konumuna götürür.



Kuklayı belirli bir saniye sürede rastgele bir konuma götürür.



Kuklayı belirli bir saniye sürede belirtilen konuma götürür.



Karakterin hangi yöne döneceğini belirler. (0=yukarı, 90=sağ, 180=aşağı, -90=sol)



Kukla fare imlecine doğru döner.



Kuklanın bulunduğu konumun x değeri istenilen değer kadar artırılır veya azaltılır.



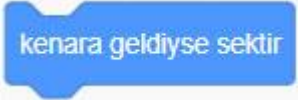

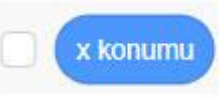
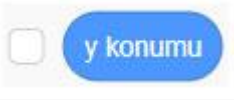

Kuklanın bulunduğu konumun x değeri istenilen değere çekilir.



Kuklanın bulunduğu konumun y değeri istenilen değer kadar artırılır veya azaltılır.

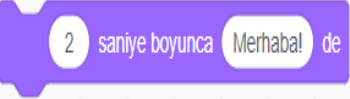
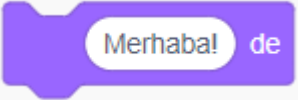
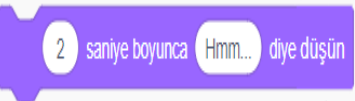
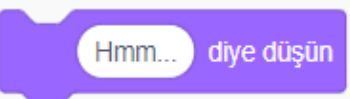
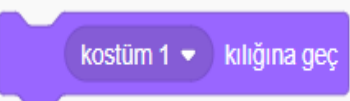
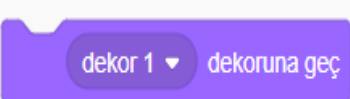


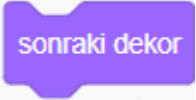
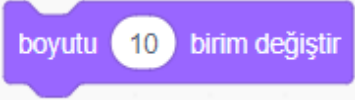
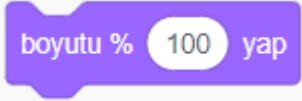
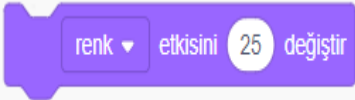
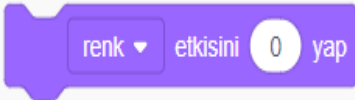
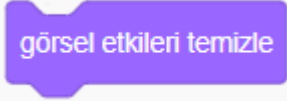

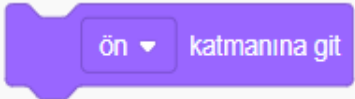
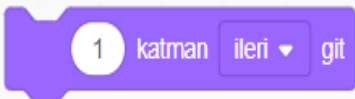
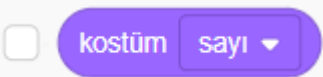
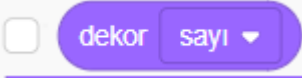
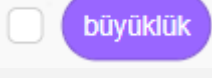
Kuklanın bulunduğu konumun y değeri istenilen değere çekilir.

	Kukla kenara değdiği zaman geldiği yönün tam tersine döner.
	Kuklanın sağa-sola dönme, etrafında dönebilme ve hiç dönememe ayarlarını yapar.
	Kuklanın x pozisyonu bilgisini verir. Bu seçenek işaretlendiğinde x değerini ekranda görebiliriz.
	Kuklanın y pozisyonu bilgisini verir. Bu seçenek işaretlendiğinde y değerini ekranda görebiliriz.
	Kuklanın yön bilgisini verir. Bu seçenek işaretlendiğinde yön bilgisini ekranda görebiliriz.

Scratch 3 Görünüm Blokları

Görünüm blokları, sahnenin ve sahnedeki kuklanın görünümüyle ilgili değişiklikleri yapmamızı sağlayan kod bloklarıdır.

	Kukla istenilen süre boyunca 'Merhaba!' yazan kutudaki değeri ekranda konuşma balonu içerisinde gösterir.
	Kukla 'Merhaba' yazan kutudaki değeri ekranda balon içerisinde gösterir.
	Kukla istenilen süre boyunca 'Hmm...' yazan kutudaki değeri ekranda düşünme balonu içerisinde gösterir.
	Kukla 'Hmm...' yazan kutudaki değeri ekranda balon içerisinde gösterir.
	Kuklanın istenilen kostüme geçmesini sağlar.
	Kuklanın o anki kostümünden bir sonraki kostümüne geçmesini sağlar.
	Dekoru seçili dekor ile değiştirir.

	Dekoru bir sonraki dekor ile değiştirir.
	Kuklanın boyutunu istenilen değer kadar değiştirir.
	Kuklanın boyutunu istenilen % değerinde değiştirir.
	Kuklanın rengi, balıkgözü, Hızlı dön, Piksellere böl, mozaik, parlaklık, hayalet efekti sayı yazan yerdeki değer kadar değişir.
	Kuklanın rengi, balıkgözü, Hızlı dön, Piksellere böl, mozaik, parlaklık, hayalet efekti sayı yazan yerdeki değer olur.
	Kuklanın üzerinde uygulanmış bütün efektleri geri alır.
	Kuklayı gösterir ve gizler.
	Kuklanın diğer bir üst katmana çıkmasını sağlar. Böylece kukla diğer karakterlerin önünde görünebilir.
	Kuklanın istenilen katman kadar öne veya alta gitmesini sağlar.
<input type="checkbox"/> 	Kuklanın kostüm sayısını veya ismini ekranda gösterir.
<input type="checkbox"/> 	Dekorun sayısını veya ismini ekranda gösterir.
<input type="checkbox"/> 	Kuklanın büyüklüğünü ekranda gösterir.

Scratch 3 Ses Blokları

Ses blokları uygulamamızı daha ilgi çekici hale getirmemizi sağlayan ses bloklardır.

	Seçili olan müzik çalmaya başlar ve bitene kadar diğer bloğa geçilmez.
	Seçili olan müzik çalmaya başlar. Program müziğin bitmesini beklemeden diğer blokları çalıştırmaya devam eder.
	Çalmaya devam eden müzikler bu blok çalıştığında durur.
	Çalan sese efekt ekler.
	Çalan sese efekt ekler.
	Efektleri kaldırır.
	Ses yüksekliğini girilen değer kadar değiştirir.
	Ses düzeyini girilen değere getirir.
	Ses düzeyini ekranda gösterir.

Scratch 3 Olaylar Blokları

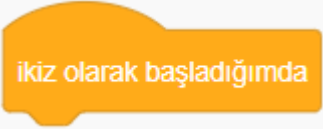
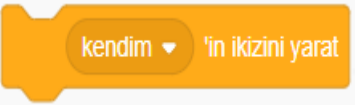
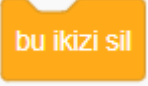
Bir uygulamanın çalışmaya başlayabilmesi için bir tetikleyiciye ihtiyacımız vardır. Uygulamanın başlangıcı için genellikle sahnenin sağ üst köşesindeki yeşil bayrağı kullanırız. Benzer şekilde uygulamayı sonlandırmak için sabit görevi bulunan kırmızı buton görevlendirilmiştir. Ancak yeşil bayrağın haricinde farklı seçeneklerimiz de mevcuttur.

	Bir kodun çalışmaya başlayabilmesi en sık kullanılan tetikleyicidir. Yeşil bayrağa basıldığında bu kod bloğuna eklediğimiz diğer bloklar çalışmaya başlar.
	Klavyeden basılacak herhangi bir tuşu tetikleyici olarak seçtiğimizde kullanacağımız kod bloğudur.
	Bir kuklaya tıklandığında çalışmasını istediğimiz kodları bu kod bloğunun altına ekleriz.
	Sahne dekorlarını yeri ve zamanı geldiğinde program içerisinde değiştirebiliriz. Dekor belirtilen bir dekor olduğunda çalışmasını istediğimiz kodları bu kod bloğunun altına ekleyebiliriz.
	Ses şiddeti, süre ölçer ve video hareketi değerlerinin belirli bir sayının üzerinde olduğunda çalışmasını istediğimiz kodları bu kod bloğunun altına ekleriz
	“Haber1” haberi geldiğinde yapılmasını istediğimiz işlerin kodlarını bu kod bloğunun altına ekleriz.
	“Haber1” haberi tüm kuklalar ve dekorlar için çalışma ortamında yayımlanır. Herhangi bir kukla haber1 haberi geldiğinde yapacağı görev var ise görevini yerine getirir.
	“Haber1” haberini tüm kuklalara gönderir ve kuklanın kodu bitirmesini bekler.

Scratch 3 Kontrol Blokları


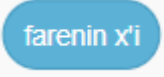
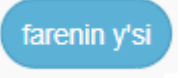
Kontrol kategorisinde bulunan kod blokları programın akışını belirli şartlara göre yönlendirebilir, tekrar eden görevleri yerine getirebilir, şart ifadesi meydana gelene kadar akışı bekletebilir.

	Bir kuklaya yazılan kod akışının belirtilen süre kadar beklemesini sağlar. Bu esnada diğer kuklalara yazılan kodlar çalışmasını devam ettirir.
	Belirtilen sayı kadar, yazılan kodun tekrar etmesi sağlanır. Döngü bloğudur.
	Bu kod bloğunun içerisine yazılan komutların sürekli olarak tekrar etmesi sağlanır. Sonsuz döngü olarak da bilinir. Uygulama durmadan döngüden çıkılamaz.
	“Eğer” kod bloğu bir şart ifadenin yerine gelip gelmediğini sorgular. Başka bir deyişle şart ifadenin sonucu “doğru” veya “yanlış” değer üretir. “Eğer” şart ifade yerine geliyor ise yani “doğru” değer üretiyorsa bu kod bloğunun içine yazılan kodlar çalıştırılır.
	Eğer şartı sağlanıyor yani “doğru” değer üretiyorsa eğer bloğu içerisine yazılan komutlar çalıştırılır. Şart ifade yerine sağlanmıyor yani “yanlış” değer üretiyorsa, değilse bloğu içindeki kodlar çalıştırılacaktır.
	Bir şart sağlanıncaya kadar program akışını o kukla için bekletir.
	Bir şart sağlanıncaya kadar tekrar edilmesi gereken komutları çalıştırır. While döngüsü olarak da bilinir.
	Tüm komutların çalışmasını, sadece eklendiği komut dizisini veya eklendiği kuklanın diğer komut dizilerini durdurmak amacıyla kullanılır.

	Bir kuklanın klonunu (ikizini) oluşturduğumuzda bu klonun yapacağı görevleri belirtmek için kullanılan başlangıç bloğudur.
	Sahnede bulunan diğer kuklaların veya mevcut kuklanın klonunu (ikizini) yaratmak için kullanılan kod bloğudur.
	Oluşturulmuş bir klon kuklanın silinmesini sağlar.

Scratch 3 Algılama Blokları

Algılama kod blokları sahnede bulunan kuklalar, fare imleci, klavye tuşları, ses şiddeti, video hareketi gibi birçok olayı algılamak için kullanılır. Kullanıcı ile etkileşimli uygulamalar geliştirmek için oldukça faydalı kod bloklarıdır. Genellikle bir şart ifadesi içerir. Fareye değdi mi? Tuşa Basıldı mı? gibi...

	Sahnedeki kuklaların birbirine, kenara veya fareye değip değmediğini sorgulamak için kullanılır. Eğer ki sorgulanan kukla seçilen nesnelerden birine değiyor ise “Doğru” sonucu üretilir.
	Sahnedeki kuklanın seçilen bir renge değip değmediği sorgulanır.
	Seçilen renkteki bir nesnenin başka renkte bir nesneye değip değmediği sorgulanır.
	Kuklanın fareye veya diğer kuklalara olan mesafesini sorgular
	Kullanıcı ile soru cevap şeklinde etkileşime girmek için kullanabileceğimiz bir komuttur. Kullanıcıya sahnede bir soru sorulur ve kullanıcının verdiği yanıt cevap değişkenine aktarılır.
	Üstteki komutta sorulan sorunun cevabı bu değişken içine atılır.
	Klavyeden basılan tuşu sorgular.
	Klavyeden basılan tuşu sorgular.
	Farenin sahne üzerindeki X konumunu sorgular.
	Farenin sahne üzerindeki Y konumunu sorgular.

	Süreklenebilir modu değiştirir.
<input type="checkbox"/> ses yüksekliği	Mikrofondan alınan ses şiddetini sorgular.
<input type="checkbox"/> zamanlayıcı	Zamanlayıcı değerini sorgular.
	Zamanlayıcıyı sıfırlar.
	Sahnede bulunan diğer kuklaların X konumu, Y konumu, yönü, kılık numarası gibi bilgileri elde etmek için kullanılır.
<input type="checkbox"/> şimdi yıl	Bilgisayarınızdaki saat bilgisinin yıl, ay, gün, saat, dakika ve saniye bilgisini almak için kullanılır.
	2000 yılından itibaren geçen gün sayısını gösterir.
<input type="checkbox"/> kullanıcı adı	Online editörde sisteme giriş yapan kullanıcının adını gösterir.

Scratch 3 Operatör Blokları

Operatörler kategorisinde matematiksel işlemler, şart ifadelerin sonuçları, metin türünden ifadeler gibi birçok işlemi yaparken faydalanacağımız kod blokları mevcuttur.

	Bloklar sırasıyla iki sayıyı toplamak, çıkarmak, çarpmak ve bölmek için kullanılır.
	Belirtilen iki sayı değeri arasında rastgele bir sayı oluşturur.
	İki ifade arasında soldaki ifadenin sağdaki ifadede büyük olup olmadığını sorgular. Buraya girilecek değer sayısal ifadeler olabileceği gibi metin ifadeler de olabilir. z>a ifadesi doğru (true) sonuç üretecektir. e>k ifadesi ise (false) sonuç üretir.
	İki ifade arasında soldaki ifadenin sağdaki ifadede küçük olup olmadığını sorgular. Buraya girilecek değer sayısal ifadeler olabileceği gibi metin ifadeler de olabilir. a<b ifadesi doğru (true) sonuç üretecektir. c</b
	İki ifade arasında soldaki ifadenin sağdaki ifadeye eşit olup olmadığını sorgulanır. Buraya girilecek değer sayısal ifadeler olabileceği gibi metin ifadeler de olabilir.
	VE operatörü iki şart ifadenin de doğru (true) sonuç üretmesini bekler. Her iki şart ifadede doğru(true) sonuç üretiyorsa VE operatörü de doğru sonuç üretir. Şart ifadelerden birisi yanlış(false) sonuç üretiyorsa VE operatörü de yanlış sonuç üretir.

	VEYA operatörü kendisine verilen şart ifadelerden birisi dahi doğru sonuç üretmesi durumunda doğru(true) sonuç üretir. Şart ifadelerden her ikisi de yanlış sonuç üretiyorsa VEYA operatörü de yanlış (false) sonuç üretir.
	Değil operatörü; bir şartın tersi değer üretir. Örneğin şart ifade olarak doğru(true) sonuç üreten bir ifade verilirse DEĞİL operatörü yanlış(false) sonuç üretecektir.
	İki metin ifadeyi birleştirmek amacıyla kullanılır.
	Verilen metin ifadenin belirtilen sıradaki harfini verir.
	Verilen ifadenin toplam karakter uzunluğunu belirtir
	Verilen metin içerisinde girilen harfi arar.
	Soldaki sayının sağdaki sayıya göre modunu alır. Örneğin 10 MOD 6 işleminin sonucu Bildiğiniz üzere 4'dür.
	Verilen sayıyı en yakın tamsayıya yuvarlar.
	Verilen sayının karekök, sin, aşağı yuvarlama, mutlak değer gibi birçok matematiksel fonksiyon sonucunu verir.

Scratch 3 Değişkenler Blokları

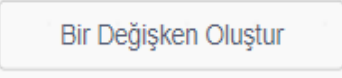
Değişkenler kategorisinde iki temel blok bulunmaktadır. Bunlar “Bir değişken oluştur” ve “Bir Liste Oluştur” kod bloklarıdır. Bu kod bloklarını incelemek için değişken ve liste kavramlarını inceleyelim.


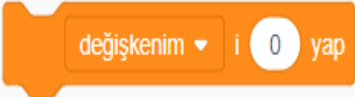

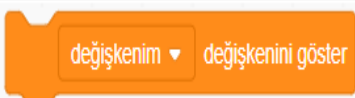

Değişken (Variable): Program yazarken kullanıcıdan alınan bilgiler, hesaplama sonrası ortaya çıkan sonuç değerleri, bir kuklanın koordinat sistemindeki yeri gibi birçok bilgiyi geçici olarak bilgisayarın belleğinde saklamak gerekir. Bunun için değişkenler kullanılır.

Bir değişken oluşturduğumuzda bilgisayarın belleğinde değişken için bir alan ayrılır.

Bu alana ulaşmak için değişkenin ismini kullanırız. Bu değişkenin adını kullandığımız her yerde değişkenin değeri geçerlidir. Bir değişkene değer atamak için “=” operatörünü kullanırız. Örnek olarak sonuç değişkenine değer atamak için `sonuc=5+5` ifadesi kullanılmıştır. Bellekteki sonuc isimli kutucuğa (alana) 10 değeri aktarılır. Bir sonraki işlemde “`sonuc=sonuc+10`” şeklinde bir atama yapsaydık sonuc değişkeninin son değeri 20 olacaktır. Bellekteki kutucukta artık 20 değeri tutulmaktadır.

Liste: Programlama dillerinde dizi(array) olarak bilinen listeler, birden fazla değişkenin tek bir isim altında birçok kutucuğa sahip değişken olarak ifade edebiliriz. “Notlar” adında oluşturulmuş bir listenin bellekte yerleştirilmiş olduğunu düşünelim. Notlar listesine 3 eleman eklendiğini varsayalım. Bu elemanların değerleri sırasıyla 60, 80 ve 90 olsun. Notlar dizisinin 1. elemanını uygulamamızda kullandığımızda 60 değerini ifade edecektir. İstersek bu elemanın değerini değişkenlerde olduğu gibi değiştirebiliriz.

	Yeni bir değişken oluşturmak için kullanılır. Değişkenin seçili kukla için mi yoksa hepsi için mi kullanılacağı oluşturulurken seçilir.
--	---

	Değişkeni ve değerini ekranda gösterir.
	Değişkene belirtilen değeri aktarır.
	Değişkenin değerini belirtilen değer kadar arttırır. Değerini azaltmak için – değeri kullanmak gerekir.
	Değişkeni sahnede göstermek için kullanılır.
	Sahnede gösterilen değişkeni gizlemek için kullanılır.

Kaynaklar

<https://blockodlama.com/scratch-3-kod-bloklari-tanitimi/>

<https://www.bilgisayarbilisim.net/konular/scratch-3-0-konu-anlatimi.169704/>