Programlama Temelleri

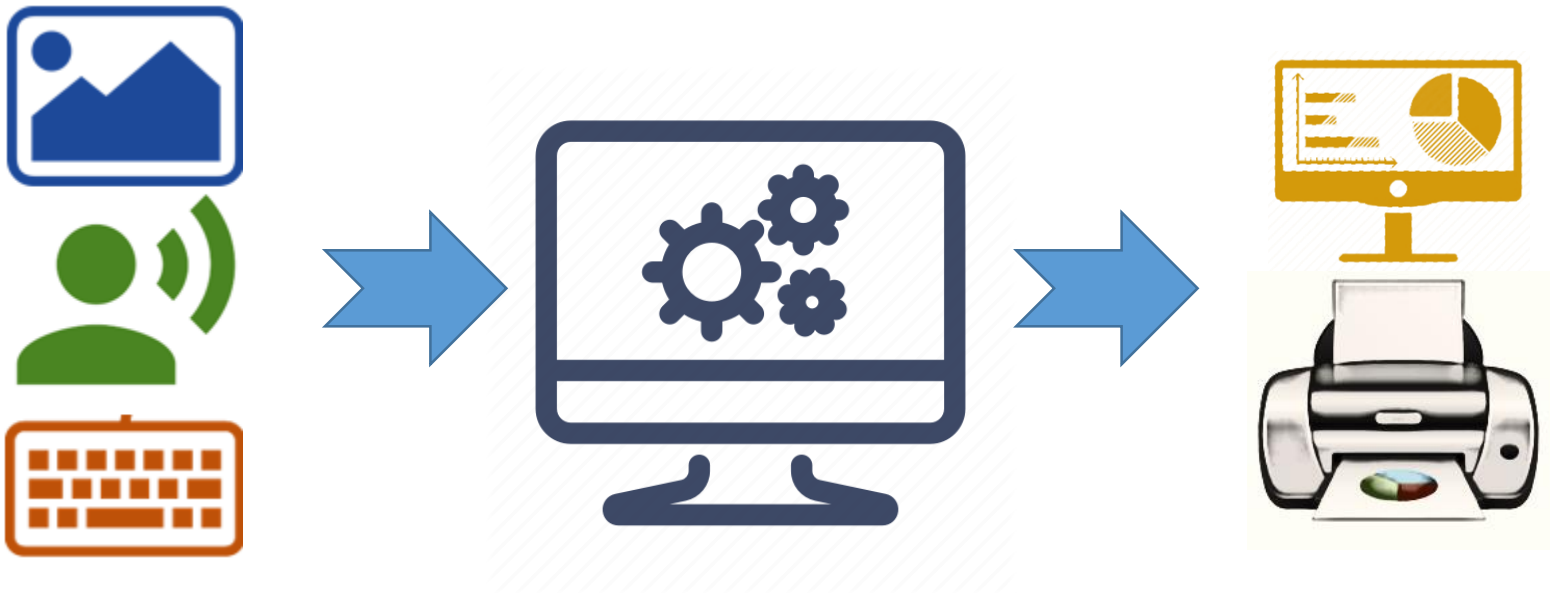
Bu derste neler öğreneceğiz?

- Algoritma tasarımı ve algoritma yazma biçimleri
- Pseudo code (kaba kod) kullanımı
- Algoritmanın akış diyagramıyla gösterilmesi
- C# programlama dilinde temel işlemler

Derse başlamadan önce…

Bilgisayar Nedir?

- Bilgisayar: dış dünyadan veriler alan ve bu verilerin yönlendirildiği programa göre çıktılar üreten bir makinedir.



GİRDİ
(INPUT)

İŞLEM

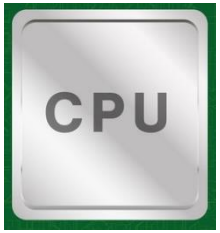
ÇIKTI
(OUTPUT)

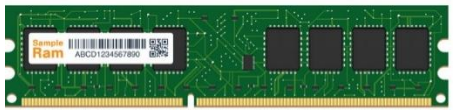
Bilgisayarın temel bileşenleri

DONANIM
(HARDWARE)

YAZILIM
(SOFTWARE)

Windows





Bilgisayar

Bilgisayarda işlemlerimizi nasıl yürütüyoruz?

- Donanımlar vasıtasıyla programları (yazılımları) kullanarak.

Program nedir?

- Bilgisayarın belirli bir işi yerine getirmesi için yazılmış olan kodlar bütünüdür.

Programlama Nedir?

- Çözülmesini istediğimiz bir problemin ya da yerine getirilmesini istediğimiz **bir işin nasıl yapılacağını** bilgisayarın anlayabileceği bir biçimde **ifade etmektir**.

Programlama

Programlama: Çözülmesini istediğimiz bir problemin ya da yerine getirilmesini istediğimiz **bir işin nasıl yapılacağını** bilgisayarın anlayabileceği bir biçimde **ifade etmektir**.

Programlama nasıl yapılır?

- Programlama dili kullanılarak (C, C++, C#, Java, PHP vs.).

Programlama mantığını ve bir programlama dilini kimler öğrenebilir?

- **Herkes!**

Programlama ve Algoritma ilişkisi

- Algoritma, kodu yazılacak olan bir programın planıdır.

Tıpkı bir binanın temeli gibi. Planı çizmeden tuğla dizmeye başlar mısınız?

- Bir program yazacaksanız:

Önce planla, sonra kodla!

Programlama ve Algoritma ilişkisi

Neden algoritma tasarlıyoruz?

- İyi planlanmış bir programın anlaşılması ve kodlanması daha kolaydır.
- Tasarım planı belli olduğu için hataların tespiti ve düzeltilmesi daha kolaydır.
- Algoritması bilinen bir problem istenilen platform için kodlanabilir. (Web uygulaması, masa üstü uygulama, mobil uygulama)

Algoritma Nedir?

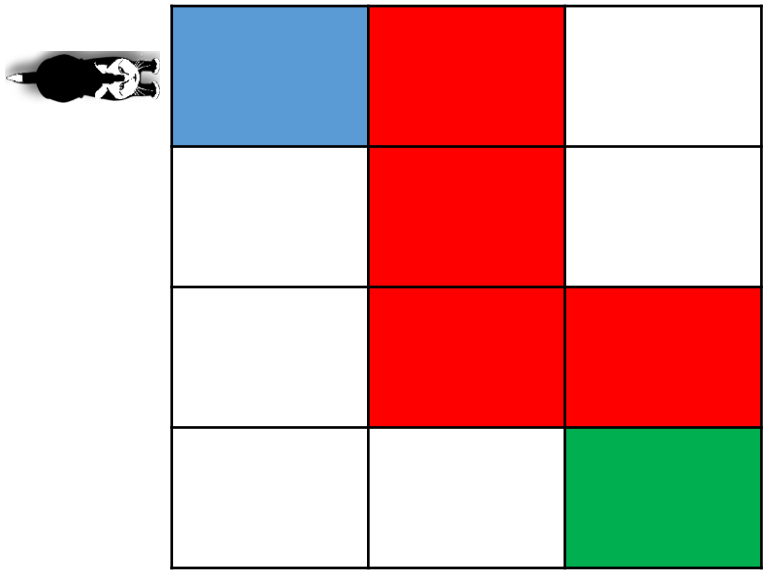
- Bir sorunu çözebilmek için gerekli olan sıralı ve mantıksal adımlar bütününe **Algoritma** denir.

Algoritma yazarken şuna dikkat edin:

- Bilgisayar sizin ne ifade ettiğinizi anlamaz, yalnızca verdiğiniz komutları uygular.
- Bilgisayar sizin anlatımınız üzerinden çıkarım yaparak sonuca varmaz, net ve kesin ifadeler kullanın!

Örnek bir problem

- Aşağıdaki haritada kedinin mavi kutudan içeri girip kırmızı yol üzerinden giderek yeşil kutudan çıkması istenmektedir. Bu problemi çözecek bir algoritma yazınız.



- Örnek bir çözüm:

1. Mavi kutudan içeri gir
2. Kırmızı yol boyunca ilerle
3. Yeşil kutudan dışarı çık

YANLIŞ!

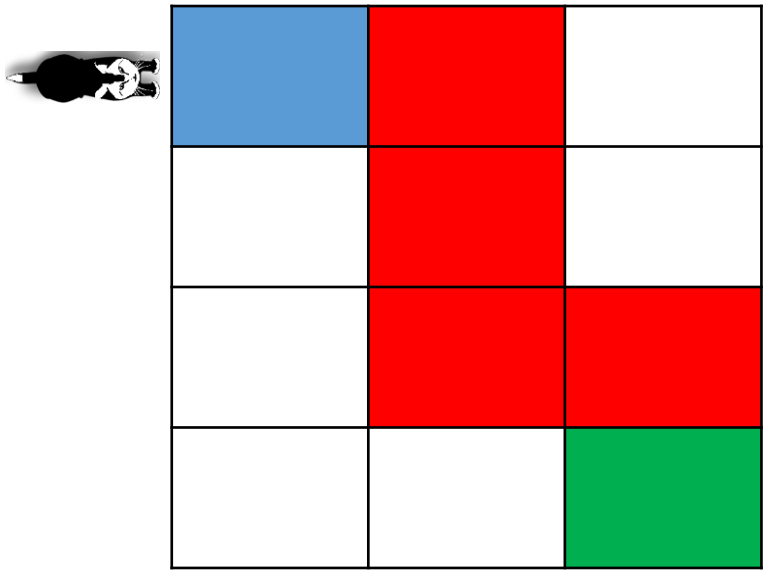
Bilgisayar sizin anlatımınız üzerinden çıkarım yaparak sonuca ulaşamaz!

Doğru algoritma nasıl tasarlanır?

- Algoritma tasarlamadan önce bilgisayarın (aslında kullandığımız programlama dilinin) neleri yapabilir olduğunu bilmemiz gerekir.
- Bu problem için varsayım:

Bilgisayar sağa veya sola dönebilir.

Bilgisayar sadece ileri gidebilir



- Çözüm:

Doğru algoritma nasıl tasarlanır?

- Algoritma tasarlamadan önce bilgisayarın (aslında kullandığımız programlama dilinin) neleri yapabilir olduğunu bilmemiz gerekir.
- Bu problem için varsayım:

Bilgisayar sağa veya sola dönebilir.

Bilgisayar sadece ileri gidebilir

- Çözüm:

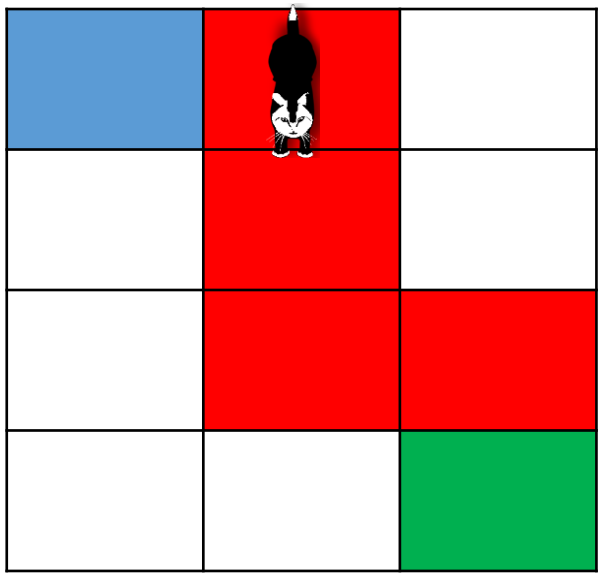
1. İki kutu ilerle

Doğru algoritma nasıl tasarlanır?

- Algoritma tasarlamadan önce bilgisayarın (aslında kullandığımız programlama dilinin) neleri yapabilir olduğunu bilmemiz gerekir.
- Bu problem için varsayım:

Bilgisayar sağa veya sola dönebilir.

Bilgisayar sadece ileri gidebilir



- Çözüm:

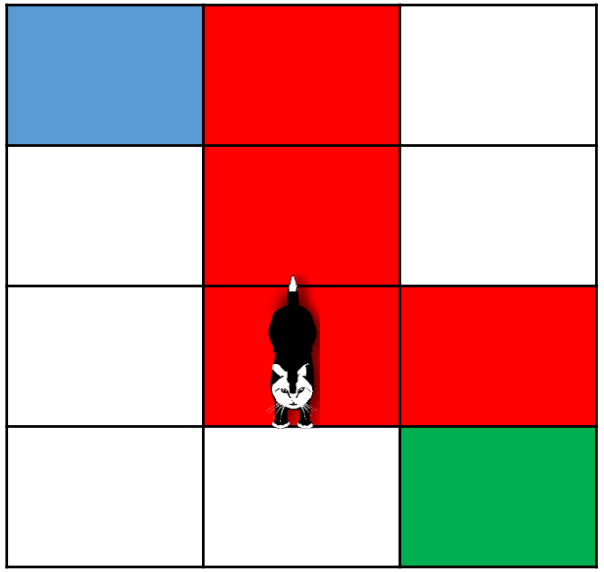
1. İki kutu ilerle
2. Sağa dön

Doğru algoritma nasıl tasarlanır?

- Algoritma tasarlamadan önce bilgisayarın (aslında kullandığımız programlama dilinin) neleri yapabilir olduğunu bilmemiz gerekir.
- Bu problem için varsayım:

Bilgisayar sağa veya sola dönebilir.

Bilgisayar sadece ileri gidebilir



- Çözüm:

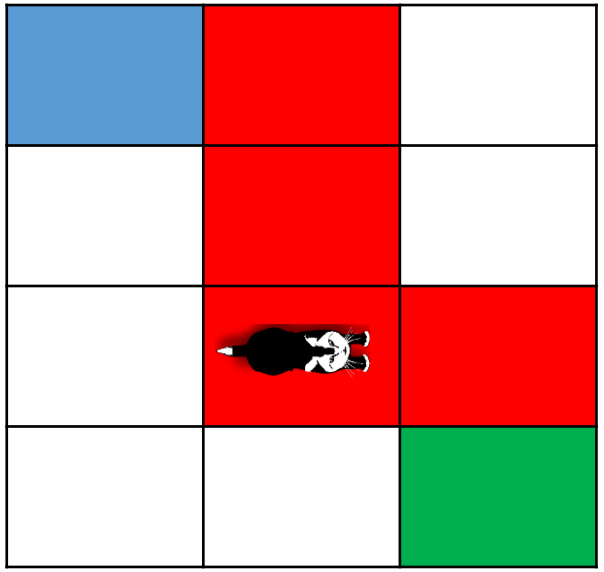
1. İki kutu ilerle
2. Sağa dön
3. İki kutu ilerle

Doğru algoritma nasıl tasarlanır?

- Algoritma tasarlamadan önce bilgisayarın (aslında kullandığımız programlama dilinin) neleri yapabilir olduğunu bilmemiz gerekir.
- Bu problem için varsayım:

Bilgisayar sağa veya sola dönebilir.

Bilgisayar sadece ileri gidebilir



- Çözüm:

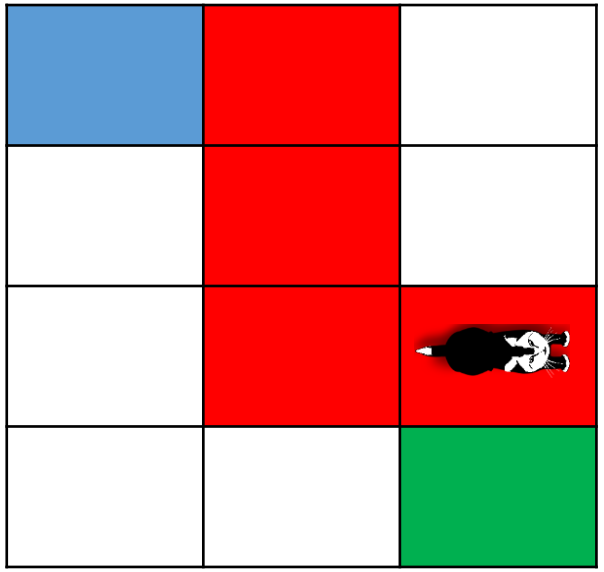
1. İki kutu ilerle
2. Sağa dön
3. İki kutu ilerle
4. Sola dön

Doğru algoritma nasıl tasarlanır?

- Algoritma tasarlamadan önce bilgisayarın (aslında kullandığımız programlama dilinin) neleri yapabilir olduğunu bilmemiz gerekir.
- Bu problem için varsayım:

Bilgisayar sağa veya sola dönebilir.

Bilgisayar sadece ileri gidebilir



- Çözüm:

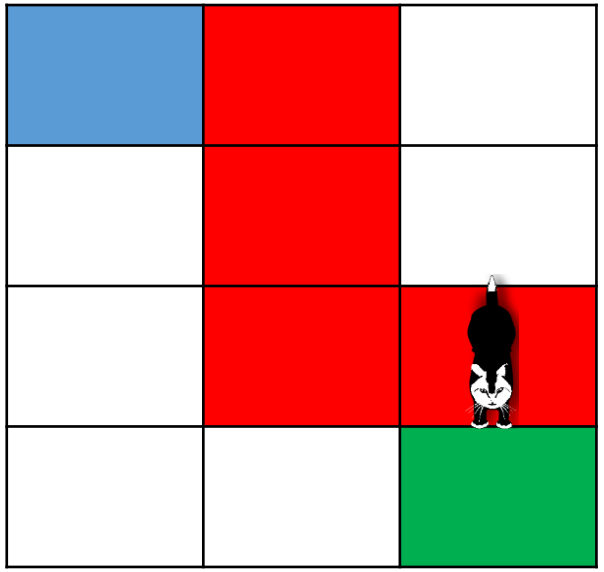
1. İki kutu ilerle
2. Sağa dön
3. İki kutu ilerle
4. Sola dön
5. Bir kutu ilerle

Doğru algoritma nasıl tasarlanır?

- Algoritma tasarlamadan önce bilgisayarın (aslında kullandığımız programlama dilinin) neleri yapabilir olduğunu bilmemiz gerekir.
- Bu problem için varsayım:

Bilgisayar sağa veya sola dönebilir.

Bilgisayar sadece ileri gidebilir



- Çözüm:

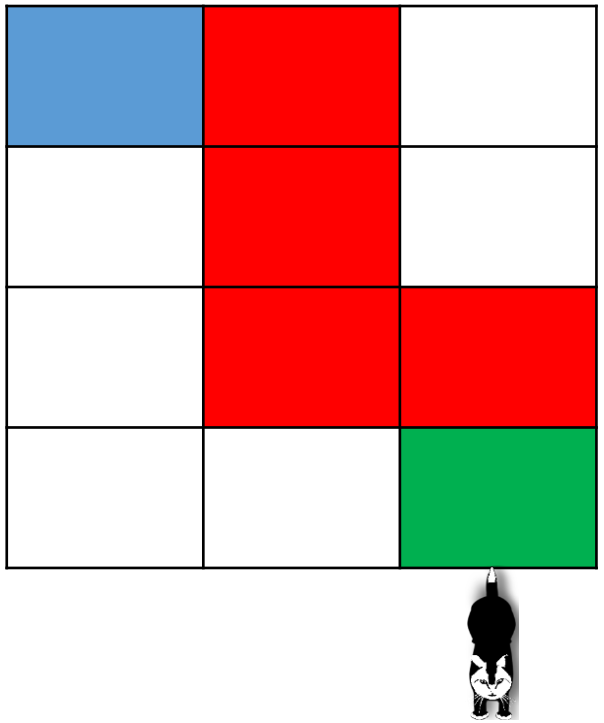
1. İki kutu ilerle
2. Sağa dön
3. İki kutu ilerle
4. Sola dön
5. Bir kutu ilerle
6. Sağa dön

Doğru algoritma nasıl tasarlanır?

- Algoritma tasarlamadan önce bilgisayarın (aslında kullandığımız programlama dilinin) neleri yapabilir olduğunu bilmemiz gerekir.
- Bu problem için varsayım:

Bilgisayar sağa veya sola dönebilir.

Bilgisayar sadece ileri gidebilir



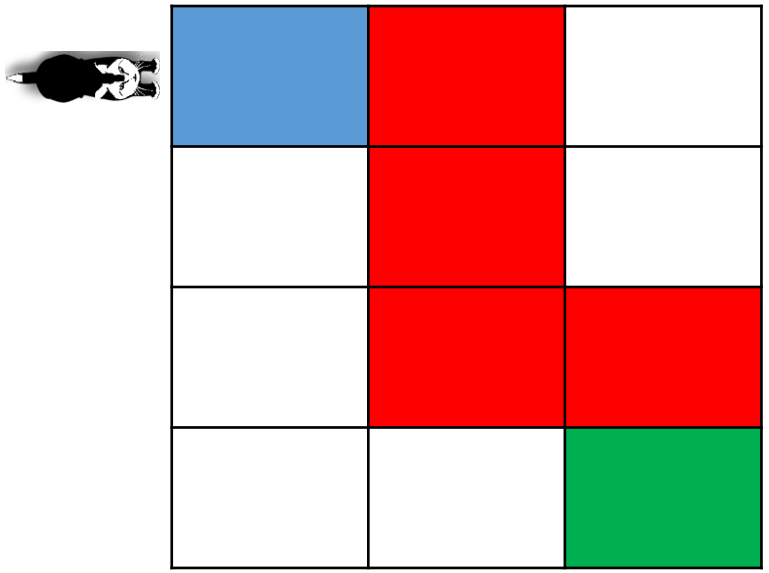
- Çözüm:

1. İki kutu ilerle
2. Sağa dön
3. İki kutu ilerle
4. Sola dön
5. Bir kutu ilerle
6. Sağa dön
7. İki kutu ilerle

Doğru algoritma nasıl tasarlanır?

- Tek doğru çözüm bu mu?
- Mantık bütünlüğü bozulmayacak ve sonuç değişmeyecek şekilde komutları birleştirmek mümkündür.

- Alternatif çözüm:

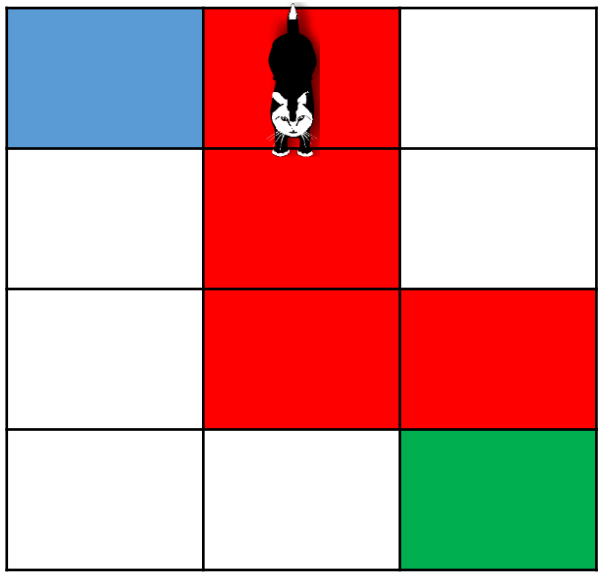


Doğru algoritma nasıl tasarlanır?

- Mantık bütünlüğü bozulmayacak ve sonuç değişmeyecek şekilde komutları birleştirmek mümkündür.

- Alternatif çözüm:

1. İki kutu ilerle ve sağa dön

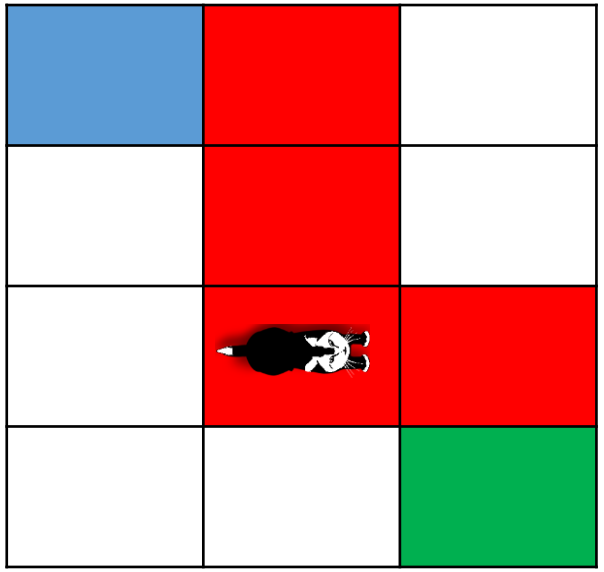


Doğru algoritma nasıl tasarlanır?

- Mantık bütünlüğü bozulmayacak ve sonuç değişmeyecek şekilde komutları birleştirmek mümkündür.

- Alternatif çözüm:

1. İki kutu ilerle ve sağa dön
2. İki kutu ilerle ve sola dön

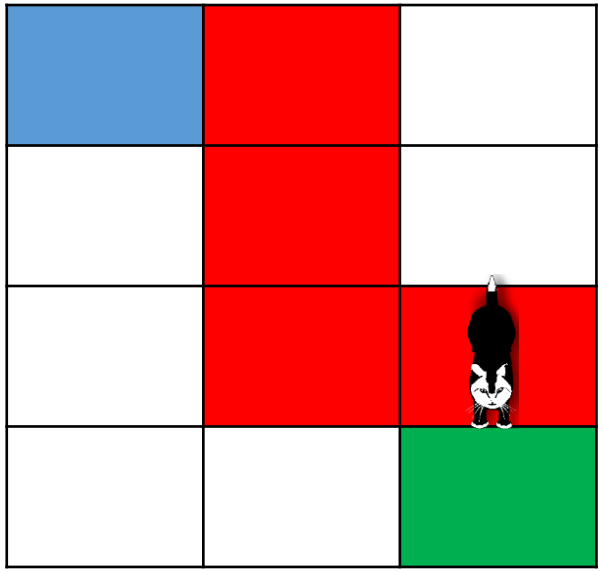


Doğru algoritma nasıl tasarlanır?

- Mantık bütünlüğü bozulmayacak ve sonuç değişmeyecek şekilde komutları birleştirmek mümkündür.

- Alternatif çözüm:

1. İki kutu ilerle ve sağa dön
2. İki kutu ilerle ve sola dön
3. Bir kutu ilerle ve sağa dön

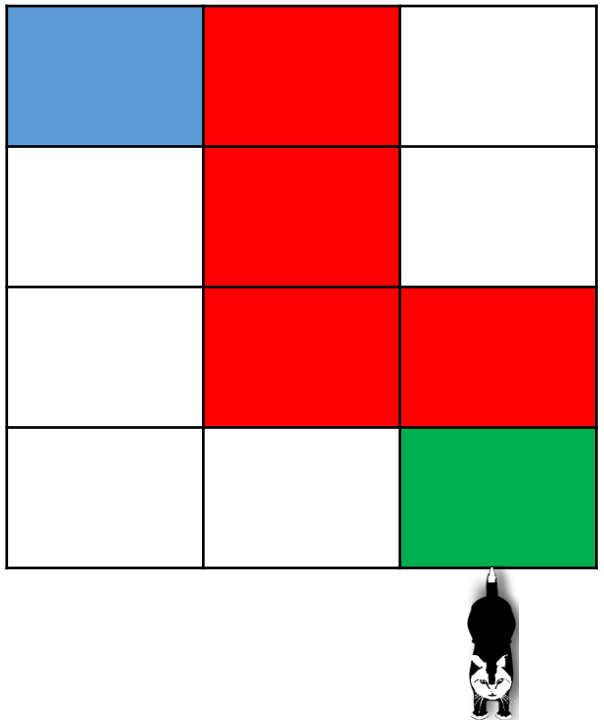


Doğru algoritma nasıl tasarlanır?

- Mantık bütünlüğü bozulmayacak ve sonuç değişmeyecek şekilde komutları birleştirmek mümkündür.

- Alternatif çözüm:

1. İki kutu ilerle ve sağa dön
2. İki kutu ilerle ve sola dön
3. Bir kutu ilerle ve sağa dön
4. İki kutu ilerle

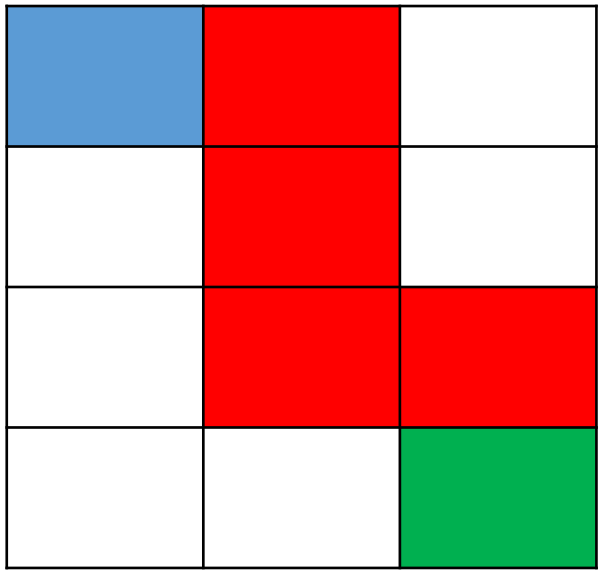


Doğru algoritma nasıl tasarlanır?

- Başlangıç pozisyonu bu şekilde olsaydı algoritma nasıl değişirdi?

1. adım: Geriye dön ? ☒ Yanlış

Bilgisayar "geriye dön" komutunu biliyor mu?



- Çözüm:

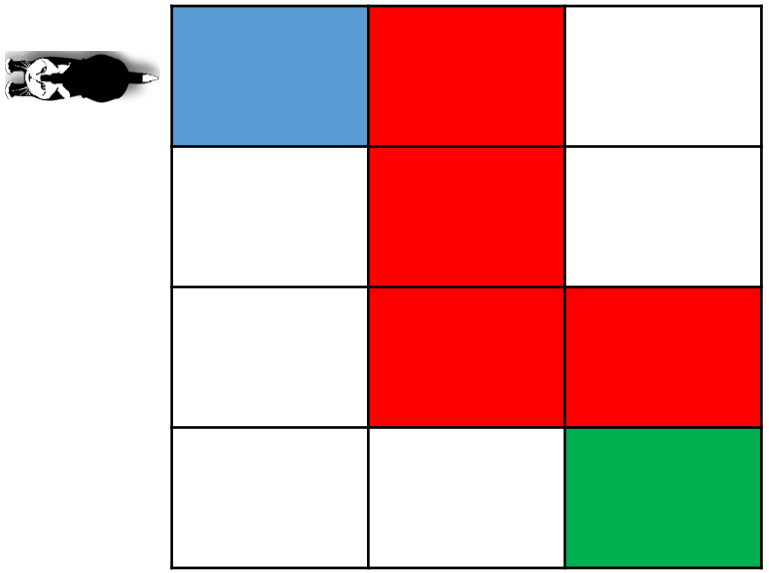
1. Sağa dön
2. Sağa dön
3. İki kutu ilerle ve sağa dön
4. İki kutu ilerle ve sola dön
5. Bir kutu ilerle ve sağa dön
6. İki kutu ilerle

Doğru algoritma nasıl tasarlanır?

- Aynı problemi çözen birden fazla doğru algoritma olabilir.

- İkinci doğru çözüm:

1. Sola dön
2. Sola dön
3. İki kutu ilerle ve sağa dön
4. İki kutu ilerle ve sola dön
5. Bir kutu ilerle ve sağa dön
6. İki kutu ilerle

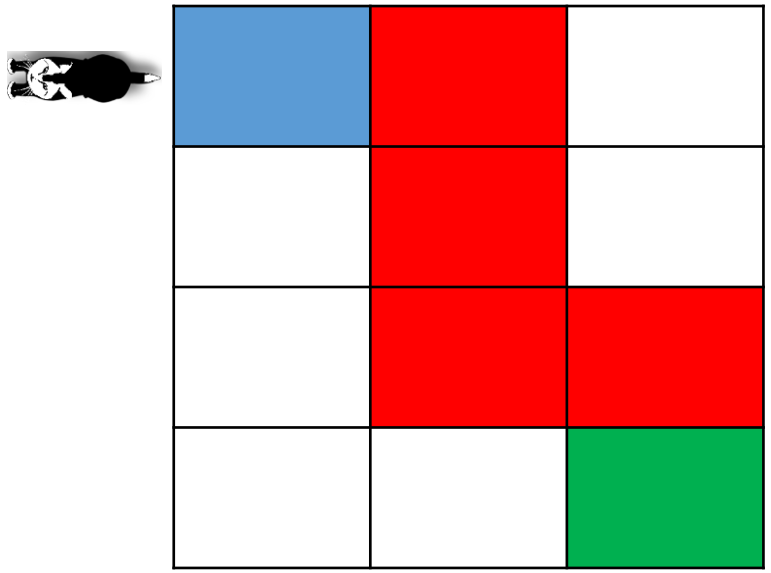


Doğru algoritma nasıl tasarlanır?

- Aynı problemi çözen birden fazla doğru algoritma olabilir.

- Üçüncü doğru çözüm:

1. İki kere sola dön
2. İki kutu ilerle ve sağa dön
3. İki kutu ilerle ve sola dön
4. Bir kutu ilerle ve sağa dön
5. İki kutu ilerle



Dördüncü bir doğru çözüm var mı?

Günlük hayattan bir örnek

- Programlama Temelleri dersi için Sanal Sınıf'a girme algoritması:

- Çözüm:

1. ue.mku.edu.tr adresine gir
2. Kullanıcı adı ve şifreni gir, sisteme giriş yap
3. Ders listesinden Programlama Temelleri dersini seç
4. Dersi işliyor olduğumuz haftanın içeriklerini bul
5. Sanal Sınıf'a tıkla ve derse katıl.

Bu adımlardan bazılarının yerini değiştirirsek ne olur?

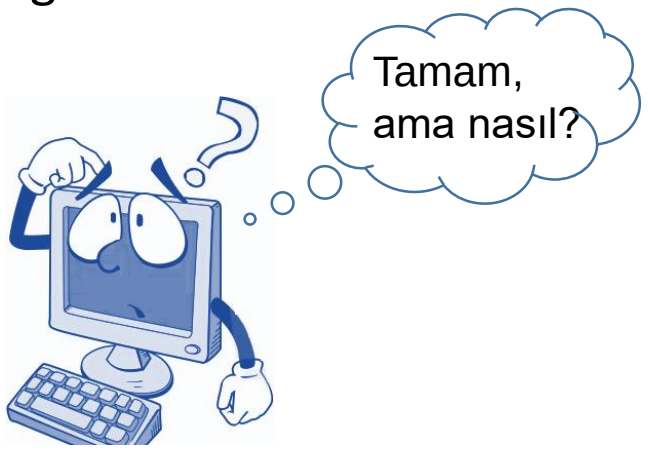
Başka bir örnek

- Kullanıcıdan iki adet sayı girmesini isteyen ve bu sayıları toplayıp sonucu ekranda gösteren bir program yazılması isteniyor. Bu programın algoritmasını yazınız.

- Çözüm:

1. Kullanıcıdan bir sayı iste ve kullanıcı sayıyı girsin
2. Kullanıcıdan başka bir sayı iste ve kullanıcı sayıyı girsin
3. Bu iki sayıyı topla
4. Bulduğun sonucu ekrana yaz.

Bilgisayar bizi tam olarak anladı mı?



Algoritma yazım şekilleri

- Doğal konuşma diline yakın bir yaklaşımla yazılabilir
- Pseudo code (kaba kod, sahte kod, sözde kod) kullanılabilir
- Akış diyagramı kullanılabilir

- Şu ana kadar doğal bir dil kullandık
- Pseudo kod ile programlama diline daha yakın olan daha net komutlar verilebilir.

Pseudo kod nedir?

- Herhangi bir programlama diline bağlı kalmadan, ama bir programlama dilinin yerine getirebileceği işlerden faydalanarak bir algoritmanın tasarlanmasıdır.

- Bir programlama dili temel olarak neler yapabilir?

Kullanıcıdan giriş verisi alabilir (klavye, kamera, mikrofon)

Aldığı verileri kalıcı veya geçici olarak saklayabilir

Aldığı verileri kullanarak işlemler gerçekleştirebilir

Çıkış cihazlarında gösterilecek sonuçlar üretebilir (ekran, yazıcı)

Veriler nerede saklanır?

- Veriler bir programın çalışması sırasında kalıcı veya geçici olarak saklanıyor olabilir.

- Programın dışında kalıcı olarak nerelerde saklanabilir?

Sabit disk, USB bellek, harici depolama diski

- Programın içinde geçici olarak nerelerde saklanır?

Değişkenlerde!

- Programcı, nerede saklandığını bilmediği bir veriyi işleyecek komutlar veremez.
- Bilgisayar da işlemek istediğimiz veriyi kendisine net olarak işaret etmezsek bunu tarif ile bulamaz.

Değişken nedir?

- Bir program çalışırken dışarıdan klavye veya başka aygıtlar kullanılarak alınan, ya da bir işlem/hesaplama sonucunda elde edilen veriler olur.

- Bu verilerin program çalışken saklandığı ve program tarafından kullanılan bellek (hafıza) birimlerine **değişken** denilir.

- Bu bellek birimlerine (değişkenlere) belirli bir isim verilir. Daha sonra değişkenler kendilerine verilen isim ile çağrılarak sakladıkları değerler kullanılabilir.

- Programcı, programın çalışması esnasında aygıtlardan alınan ya da üretilen veriye değişkenler kullanarak erişir.

Değişken

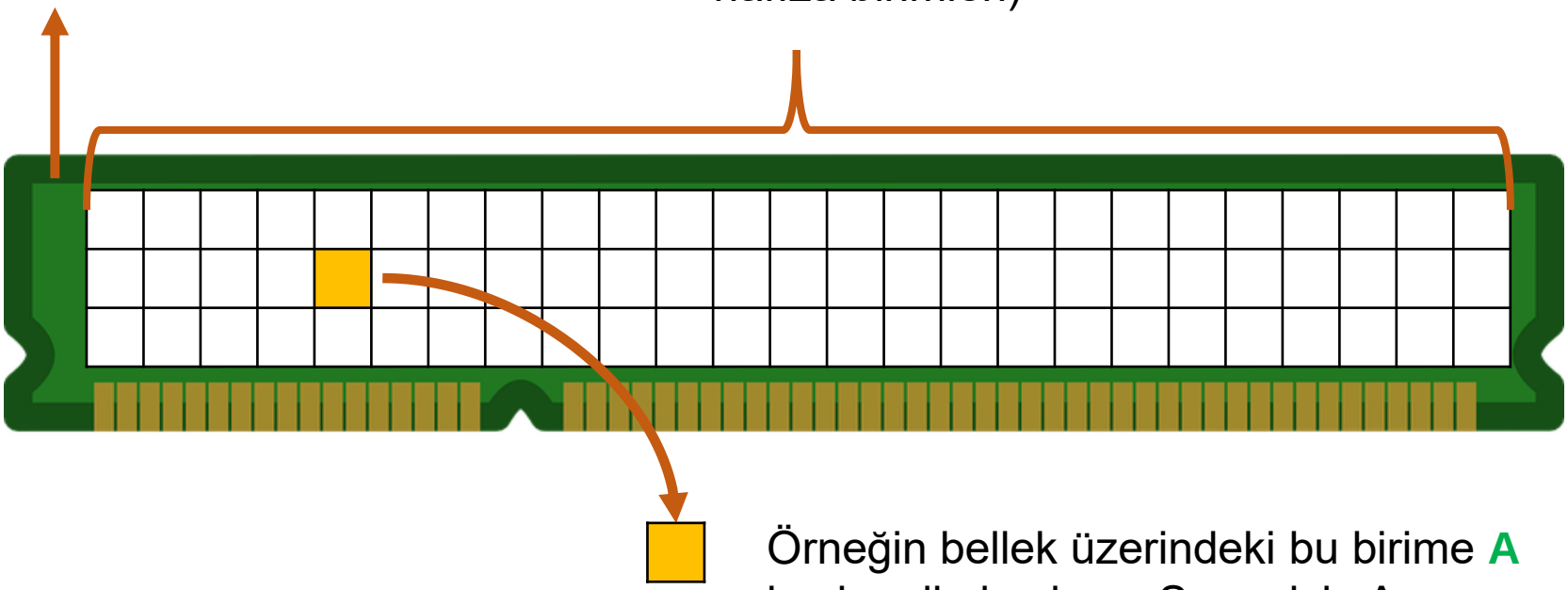
- Değişkenlerin içinde çoğunlukla sayı ya da metin tipinde veriler bulunabilir.
- Fakat veri tipleri sayı ya da metin ile sınırlı değildir!

- Değişken isimleri verilirken şu kurallara dikkat edilmelidir:
  - İngiliz alfabesindeki A-Z veya a-z arası 26 harf kullanılabilir.
  - 0-9 rakamları kullanılabilir.
  - Sembollerden alt çizgi kullanılabilir. (_)
  - Değişken isimleri harf veya alt çizgi ile başlayabilir ama rakamla başlamaz.
  - İsimler oluşturulurken boşluk kullanılmamalıdır. Örneğin “ogenci no” değil “ogrenci_no” ya da "ogrenciNo" gibi

Değişken

Bellek (Hafıza)

Bellek birimleri (verilerin saklandığı hafıza birimleri)



Örneğin bellek üzerindeki bu birime **A** ismi verilmiş olsun. Şu an için A değişkeninde bir değer yoktur. (Bellek birimi boş)

Değişkenlere değer atama

- Değişkenlere değer atama işlemleri için **=** operatörü kullanılır.
- Örneğin A = 5 denildiğinde bu, "5 değerini A değişkenine ata" anlamına gelir.

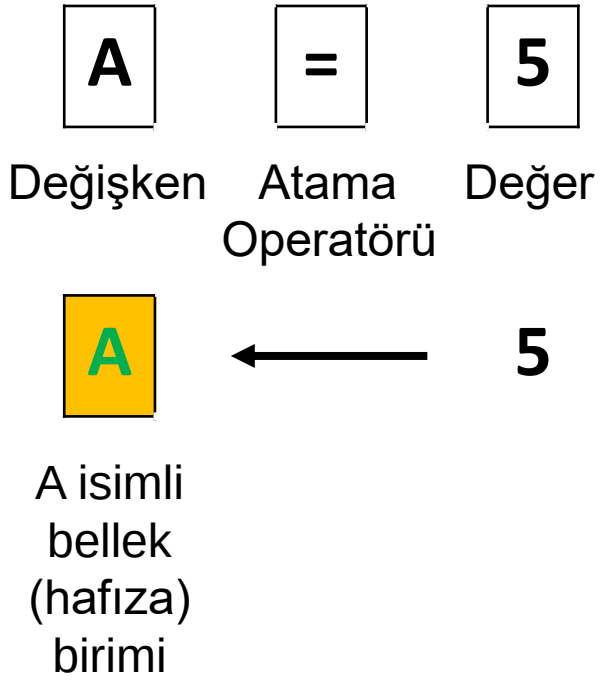
- Atama işleminde eşitliğin (=) sağındaki değer sol taraftaki değişkene aktarılır. Eşitliğin sağ tarafında aritmetik bir işlem var ise bu işlemin sonucu hesaplanır ve elde edilen değer sol taraftaki değişkene aktarılır.

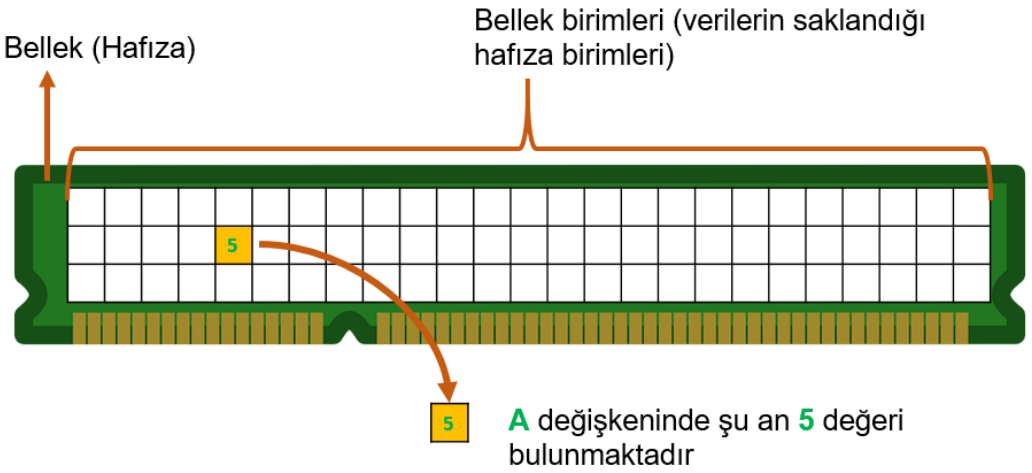
- Bir değişkene sayı tipindeki değerler atanırken değerlerin kendisi olduğu gibi yazılır. Örneğin: A = 5

- Bir değişkene metin tipindeki değerler atanırken değerler **" "** işaretleri arasında yazılır. Örneğin: B = "merhaba"

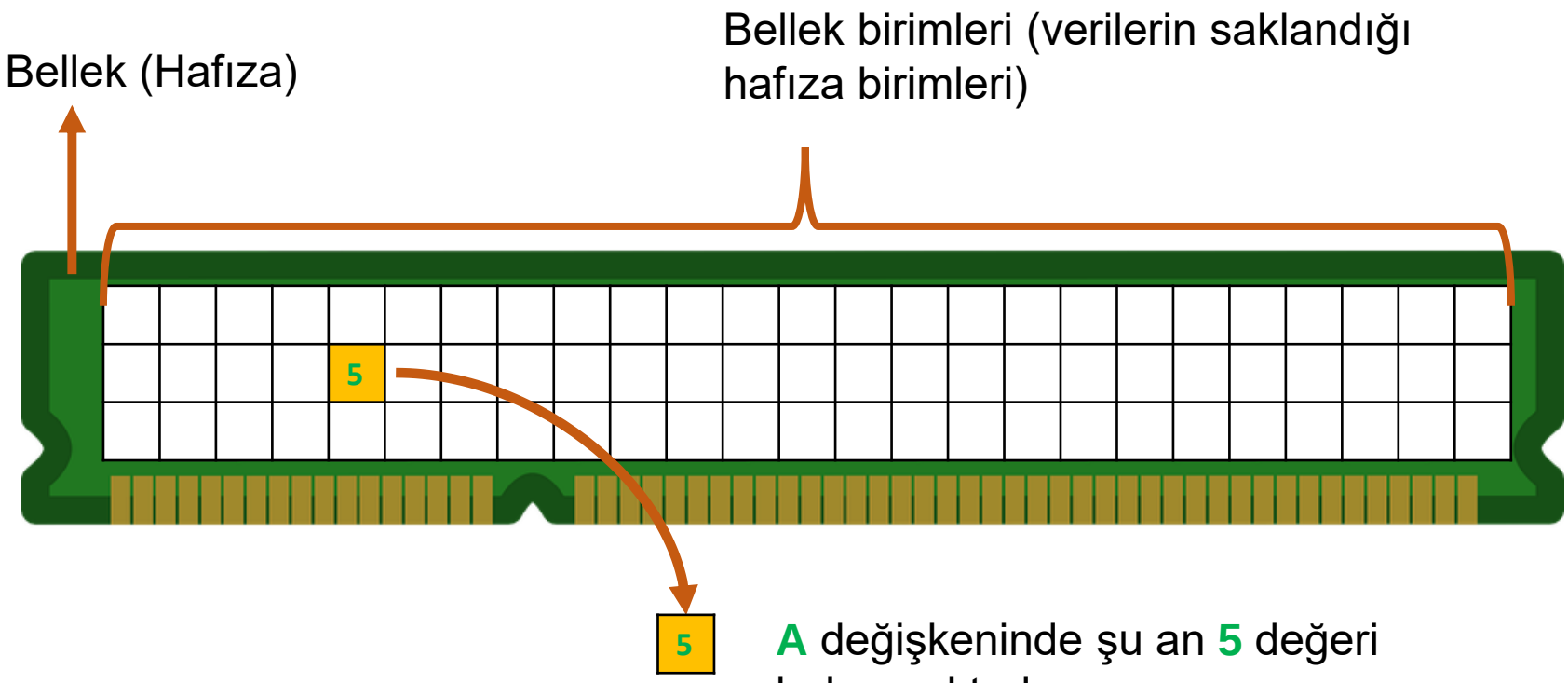
Değişkenlere değer atama

- Atama işleminde eşitliğin (=) sağındaki değer sol taraftaki değişkene aktarılır.
- Aşağıdaki şekilde, sağdaki 5 değeri atama (=) operatörü ile soldaki değişkene aktarılmaktadır. Bu işlem sonucunda artık A değişkeninin içinde 5 değeri vardır.





Değişkenlere değer atama



Aritmetik Operatörler

- Toplama a + b
- Çıkarma a - b
- Çarpma a * b
- Bölme a / b
- Üs alma a ^ b
- Mod alma a % b

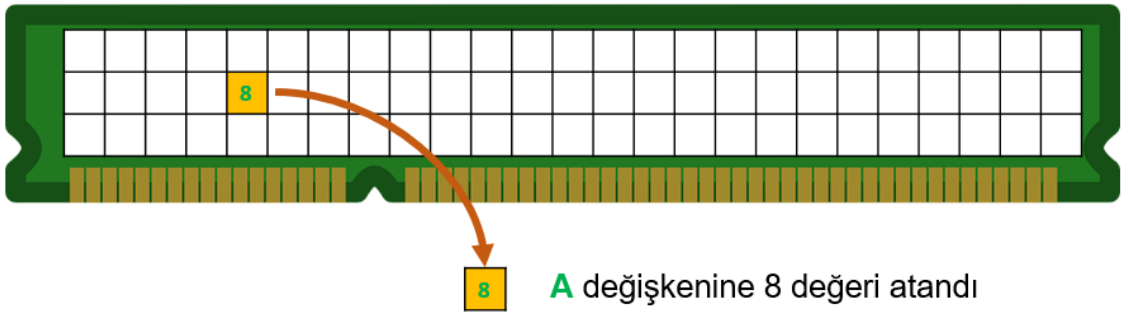
Aritmetik işlem operatörleri yalnızca sayı tipindeki veriler ile birlikte kullanılırlar!

Metin tipindeki veriler ile aritmetik işlemler yapılamaz!

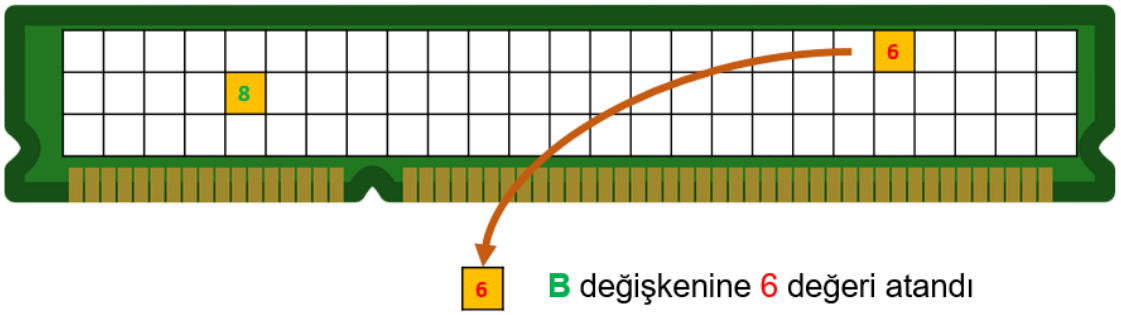
Aritmetik işlem sonuçlarını değişkene atama

- Eşitliğin (atama operatörünün) sağ tarafında aritmetik bir işlem var ise bu işlemin sonucu hesaplanır ve elde edilen değer sol taraftaki değişkene aktarılır.

A = 5 + 3



B = A - 2



Değişkenlere değer atama

- Programcılıkta eşittir (=) ifadesi değer atama işlemi için kullanılır ve matematikteki eşittir ile karıştırılmamalıdır. Atama işleminde eşittir ifadesinin sağ tarafındaki işlemler yapılır ve bulunan sonuç soldaki değişkene aktarılır.
- Bazı değer atama örnekleri:

Aşağıdaki değer atama işlemlerinin aynı program dosyası içinde olduğunu ve bu programın çalıştırıldığını varsayalım.

SAYI = 2 [SAYI değişkeninin içine 2 değerini ata]

SAYI = 5 [SAYI değişkeninin içine 5 değeri atandı. Bir önceki değer olan 2 silindi]

SAYI = SAYI + 1 [SAYI değişkenin şu anki değeri 5'tir. Eşitliğin sağ tarafındaki toplama işlemi yapılırsa 5 + 1 = 6 sonucu elde edilir. Elde edilen toplama işlem sonucu yine SAYI değişkenine atanmıştır. SAYI değişkeninin yeni değeri 6 olmuş ve eski değer olan 5 silinmiştir.]

TOPLAM = SAYI + 5 [Eşitliğin sağ tarafındaki işlem sonucu 6 + 5 = 11 olarak bulunur. Elde edilen 11 sonucu TOPLAM adlı değişkene aktarılır. TOPLAM değişkeninin değeri artık 11 olmuştur.]

A = 8 * 3 [Eşitliğin sağ tarafındaki işlem sonucu 8 x 3 = 24 olarak bulunur. Elde edilen 24 sonucu A adlı değişkene aktarılır. A değişkeninin yeni değeri artık 24 olmuştur.]

İşlem önceliği

- Matematiksel işlemler içeren kümenizin içindeki matematiksel işaretlerin bir işlem önceliği vardır.
- İşlem Önceliği Sırası:
  1. Parantezler ( )
  2. Üs alma a ^ b
  3. Çarpma ve bölme a * b veya a / b
  4. Toplama ve çıkarma a + b veya a - b

- Aynı önceliğe sahip işlemler yan yana bulunuyorsa işlem önceliği soldan sağa doğrudur. X = 2 * 3 / 6 + 2

Son örneğe geri dönersek…

- Kullanıcıdan iki adet sayı girmesini isteyen ve bu sayıları toplayıp sonucu ekranda gösteren bir program yazılması isteniyor. Bu programın algoritmasını yazınız.

- Klavyeden sayı okumak aritmetik bir işlem değil!
- Klavyeden bir sayı okunacağını algoritma yazımında nasıl ifade edebiliriz?

Klavyeden bir sayı oku ve bunu A değişkenine ata

A sayısını/değişkenini klavyeden oku

OKU: A (sayı)

OKU: A, B, C (birden fazla değişkenin klavyeden okunacağı söyleniyor)

Son örneğe geri dönersek…

- Kullanıcıdan iki adet sayı girmesini isteyen ve bu sayıları toplayıp sonucu ekranda gösteren bir program yazılması isteniyor. Bu programın algoritmasını yazınız.

- Verilerin ya da başka bilgilerin ekrana yazılması da aritmetik değil!
- Ekrana bir şey yazılacağını algoritma yazımında nasıl ifade edebiliriz?
- Bir değişkenin değerini yazdırmak istiyorsak:

  A değişkeninin değerini ekrana yaz

  YAZ: A
- Bir metinsel ifadeyi ekrana yazmak istiyorsak:

  Ekrana "merhaba" yaz

  YAZ: "merhaba"

Son örneğe geri dönersek…

- Kullanıcıdan iki adet sayı girmesini isteyen ve bu sayıları toplayıp sonucu ekranda gösteren bir program yazılması isteniyor. Bu programın algoritmasını yazınız.

- Verilerin ya da başka bilgilerin ekrana yazılması da aritmetik değil!
- Metinsel bir ifade ile bir değişkenin değeri birlikte nasıl ekrana yazılır?

  Ekrana "A değeri: " ve A değişkeninin değerini yaz

  YAZ: "A değeri: ", A

  YAZ: "A değeri: " + A (buradaki + toplama anlamında değil!)

  YAZ: "A değeri: ", A, " B değeri: ", B

Sorunun çözümü

- Kullanıcıdan iki adet sayı girmesini isteyen ve bu sayıları toplayıp sonucu ekranda gösteren bir program yazılması isteniyor. Bu programın algoritmasını yazınız.

- Bütün algoritmalar BAŞLA adımı ile başlar ve BİTİR adımı ile sona erer!
- Komutlar sıralı adımlar ile gösterilir.
- Algoritma, belirli sayıda adımdan sonra mutlaka sona ermelidir! (Adım sayısının kaç olduğu önemli değil)

- Çözüm:

1. BAŞLA
2. OKU: A, B
3. TOPLAM = A + B
4. YAZ: TOPLAM
5. BİTİR (DUR, SON)

Örnek

- Bir dikdörtgenin kısa ve uzun kenar bilgileri klavyeden girilecektir. Bu dikdörtgenin alanını ve çevresini hesaplayıp sonucu ekrana yazan bir programın algoritmasını yazın.

1. BAŞLA
2. OKU: KISA, UZUN // Kısa ve Uzun kenar bilgisini tutan değişkenler
3. ALAN = UZUN * KISA
4. CEVRE = 2 * (KISA + UZUN)
5. YAZ: "HESAPLANAN ALAN = ", ALAN
6. YAZ: "HESAPLANAN CEVRE = ", CEVRE
7. BİTİR

Çözüme dair not…

- Değişken adları yapılan iş ile ilişkili olması amacıyla KISA, UZUN, ALAN, CEVRE şeklinde seçilmiştir.
- İşi yapan değişkenlerin adları değil, yazdığımız algoritmadır!

Değişken adını CEVRE yapıp işlemi "CEVRE = KISA * UZUN" şeklinde belirtmek bir işe yarar mı?

- "**//**" nedir?

Genellikle // ifadesi kendisinden sonra gelen içeriğin açıklama/yorum olduğunu bildirir.

Algoritmayı işletirken bu içerikler dikkate alınmaz.

Açıklama satırları sadece okuyucuyu bilgilendirme amacı taşır.

Farklı programlama dillerinde yorum satırları çeşitli belirteçler ile gösterilebilir (%, # vb.)

Örnek

- Klavyeden bir sayı girilecektir. Bu sayının karesini hesaplayıp ekrana yazacak programın algoritmasını yazın.

1. BAŞLA
2. OKU: X
3. KARE = X^2 // KARE = X * X şeklinde de olabilirdi.
4. YAZ: "GİRİLEN SAYININ KARESİ = ", KARE
5. BİTİR

Örnek

- Klavyeden bir sayı ve üs bilgisi girilecektir. Girilen bilgilere göre sayının üssünü hesaplayıp sonucu ekrana yazan bir programın algoritmasını yazın.

1. BAŞLA
2. OKU: SAYI, US
3. X = SAYI ^ US
4. YAZ: "HESAPLANAN DEĞER: ", X
5. BİTİR

Örnek

- Klavyeden girilen 3 sayının ortalamasını hesaplayıp sonucu ekrana yazan bir programın algoritmasını yazın.

1. BAŞLA
2. OKU: A, B, C
3. ORTALAMA = (A+B+C) / 3
4. YAZ: "ORTALAMA DEĞER = ", ORTALAMA
5. BİTİR

Örnek

- Bir öğrencinin aldığı bir ders için vize ve final notları girilecektir. Girilen notlara göre bu öğrencinin not ortalamasını hesaplayıp sonucu ekrana yazan programın algoritmasını yazın.

Bilgi: Not ortalaması vize notunun %40'ı ile final notunun %60'ı toplanarak hesaplanır.

1. BAŞLA
2. OKU: V, F // V: Vize notu, F: Final notu
3. ORTALAMA = V*0.4 + F*0.6
4. YAZ: "HESAPLANAN ORTALAMA = ", ORTALAMA
5. BİTİR

Örnek

- Bir öğrencinin aldığı bir ders için vize ve final notları girilecektir. Girilen notlara göre bu öğrencinin not ortalamasını hesaplanacaktır. Not ortalamasına göre bu öğrencinin dersten geçtiğini ya da kaldığını hesaplayıp bunu ekrana yazan programın algoritmasını yazın.

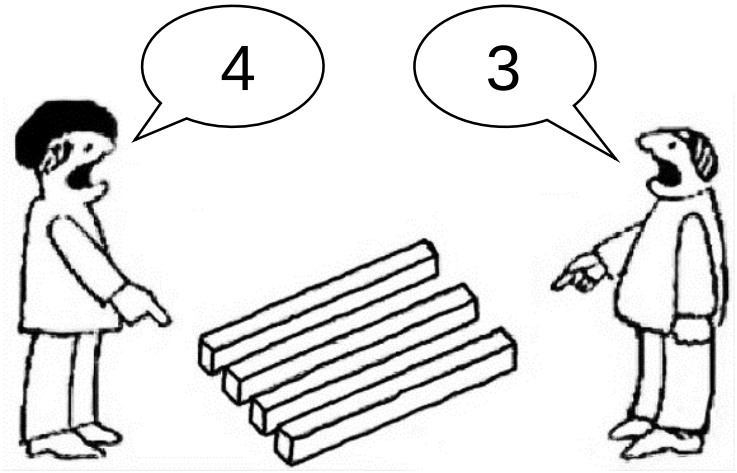
Bilgi: Not ortalaması 60 veya üzerinde ise öğrenci bu dersten geçmiş sayılır.

Nasıl çözeriz?

- Bir öğrencinin aldığı bir ders için vize ve final notları girilecektir. Girilen notlara göre bu öğrencinin not ortalamasını hesaplanacaktır. Not ortalamasına göre bu öğrencinin dersten geçtiğini ya da kaldığını hesaplayıp bunu ekrana yazan programın algoritmasını yazın.

- Bilgisayar sayısal değerleri birbirine göre büyüklük-küçüklük ilişkisi yönünden karşılaştırabilir mi?

Evet!



Karşılaştırma Operatörleri

- Eşittir a == b
- Eşit değildir (farklıdır) a != b
- Büyüktür a > b
- Küçüktür a < b
- Büyük eşittir a >= b

(büyük veya eşit)

- Küçük eşittir a <= b

(küçük veya eşit)

Karşılaştırma Operatörleri

- Nasıl çalışır?

Bu operatörler kullanılarak değişkenler ya da sabit değerler arasında yapılan karşılaştırmalar doğru ya da yanlış şeklinde bir sonuç döndürür.

- Örneğin :

5 > 8 ifadesi yanlış sonucunu döndürür

3 <= 6 ifadesi doğru sonucunu döndürür

7 <= 7 ifadesi doğru sonucunu döndürür

7 < 7 ifadesi yanlış sonucunu döndürür

Karşılaştırma Operatörleri

- Değişkenler kullanarak da karşılaştırmalar yapmak mümkündür
- Örneğin:

A > 5 ifadesi eğer A değişkeninde tutulan değer 5'ten büyük ise doğru, değil ise yanlış sonucunu döndürecektir

A<B ifadesi eğer A değişkeninde tutulan değer B değişkeninde tutulan değerden daha küçücük ise doğru, değil ise yanlış sonucunu döndürecektir

A >= A ifadesi doğru sonucunu döndürecektir

Karşılaştırma operatörleri yalnızca sayılar ya da sayı tipinde değer tutan değişkenler ile birlikte kullanılır!

Karşılaştırma Operatörleri

- Nerede kullanılır?

KOŞULLU İŞLEM İFADELERİ içinde

Koşullu İşlem İfadeleri "Karar Yapıları/İfadeleri", "Kontrol Yapıları/İfadeleri" olarak da anılabilir.

- Koşullu İşlem İfadelerinin yapısı:

EĞER [KOŞUL BELİRTİLİR] belirtilen koşul DOĞRU İSE

> Burada belirtilen işlem ya da işlemleri yap.

DEĞİL İSE (Yukarıdaki koşul doğru değil ise anlamında)

> Burada belirtilen işlem ya da işlemleri yap.

"DEĞİL İSE" kısmını kullanmak zorunlu değil, opsiyoneldir.

Koşullu İşlemler

- Kısaca

EĞER [koşul] İSE

> Burada belirtilen işlem ya da işlemleri yap.

DEĞİL İSE

> Burada belirtilen işlem ya da işlemleri yap

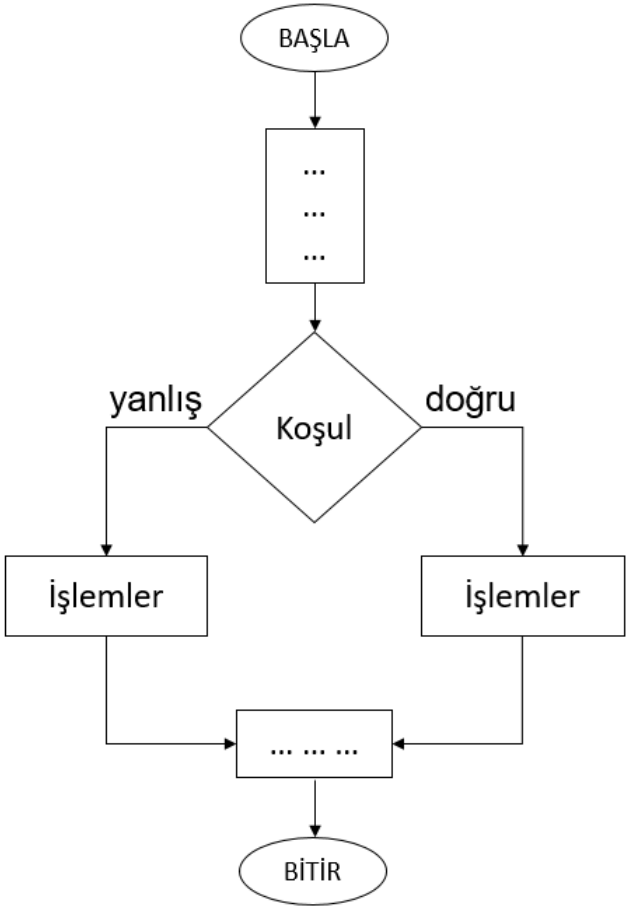
- Aslında ifade edilmek istenen:

EĞER DOĞRU İSE

> Burada belirtilen işlem ya da işlemleri yap.

YANLIŞ İSE

> Burada belirtilen işlem ya da işlemleri yap



Koşullu İşlemler

- **DEĞİL İSE** kısmında başka bir koşul ifadesi kullanılmaz.

- **EĞER [koşul] İSE**:

  Koşuldan yanlış değeri dönerse buradaki işlemler gerçekleştirilmez ve **DEĞİL İSE** kısmındaki işlemler başka bir koşula bakılmadan gerçekleştirilir.

- **EĞER [koşul] İSE**:

  Koşuldan doğru değeri dönerse burada belirtilen işlemler gerçekleştirilir ve bu durumda **DEĞİL İSE** kısmındaki işlemler gerçekleştirilmez.

Koşullu İşlemler

- **EĞER [koşul] İSE**:

  Koşul doğru olduğunda kaç işlem yapılacak olursa olsun bu ifade ve kendisine bağlı olan bütün işlemler sıralı algoritma adımları arasında tek bir adım olarak işletilir.

- **EĞER [koşul] İSE** işlemler **DEĞİL İSE** işlemler:

  Koşul doğru da yanlış da olsa, yapılacak işlemlerin kaç adet olduğuna bakılmaksızın bu ifade bütün olarak tek bir algoritma adımı olarak işletilir.

Soruya geri dönersek…

- Bir öğrencinin aldığı bir ders için vize ve final notları girilecektir. Girilen notlara göre bu öğrencinin not ortalamasını hesaplanacaktır. Not ortalamasına göre bu öğrencinin dersten geçtiğini ya da kaldığını hesaplayıp bunu ekrana yazan programın algoritmasını yazın.

Bilgi: Not ortalaması 60 veya üzerinde ise öğrenci bu dersten geçmiş sayılır.

1. BAŞLA
2. OKU: V, F
3. ORTALAMA = V*0.4 + F*0.6
4. EĞER ORTALAMA >= 60 İSE
   YAZ: "DERSTEN GEÇTİ"
   DEĞİL İSE // ORTALAMA 60'A EŞİT YA DA BÜYÜK DEĞİL İSE ANLAMINDA
   YAZ: "DERSTEN KALDI"
5. BİTİR

Başka bir örnek

- Klavyeden A ve B sayıları girilecektir. Bu sayıların büyük olanından küçük olanı çıkarılacak ve hesaplanan sonuç ekrana yazılacaktır. Bu işlemleri gerçekleştiren programın algoritmasını yazın.

1. BAŞLA
2. OKU: A, B
3. EĞER X > Y İSE
   FARK = X – Y
   DEĞİL İSE
   FARK = Y – X
4. YAZ: "HESAPLANAN DEĞER: ", FARK
5. BİTİR

Örnek

- Klavyeden bir sayı girilecektir. Girilen sayının tek ya da çift sayı olduğunu bulup bunu ekrana yazan programın algoritmasını yazın.

Bilgi: Bütün çift sayılar 2 ile tam bölünür

1. BAŞLA
2. OKU: X
3. EĞER X % 2 == 0 İSE YAZ: X, " ÇİFT SAYIDIR"
   DEĞİL İSE YAZ: X, "TEK SAYIDIR"
4. BİTİR

Örnek

- Klavyeden bir tam sayı girilecektir. Girilen sayının sıfırdan büyük ya da küçük olduğunu bulup sonucu ekrana yazacak olan programın algoritmasını yazın.

1. BAŞLA
2. OKU: X
3. EĞER X > 0 İSE YAZ: X, " SIFIRDAN BÜYÜKTÜR"
   DEĞİL İSE YAZ: X, " SIFIRDAN KÜÇÜKTÜR"
4. BİTİR

X sayısı sıfır olarak girilirse ekranda ne yazar?

Örnek

- Klavyeden bir tam sayı girilecektir. Girilen sayının sıfırdan büyük ya da küçük olduğunu bulup sonucu ekrana yazacak olan programın algoritmasını yazın.

Alternatif çözüm:

1. BAŞLA
2. OKU: X
3. EĞER X > 0 İSE
   YAZ: X, " SIFIRDAN BÜYÜKTÜR"
   5. ADIMA GİT // Böyle bir komut verebilir miyim? Evet!
4. EĞER X < 0 İSE YAZ: X, "SIFIRDAN KÜÇÜKTÜR"
   DEĞİL İSE YAZ: X, " SIFIRA EŞİTTİR"
5. BİTİR

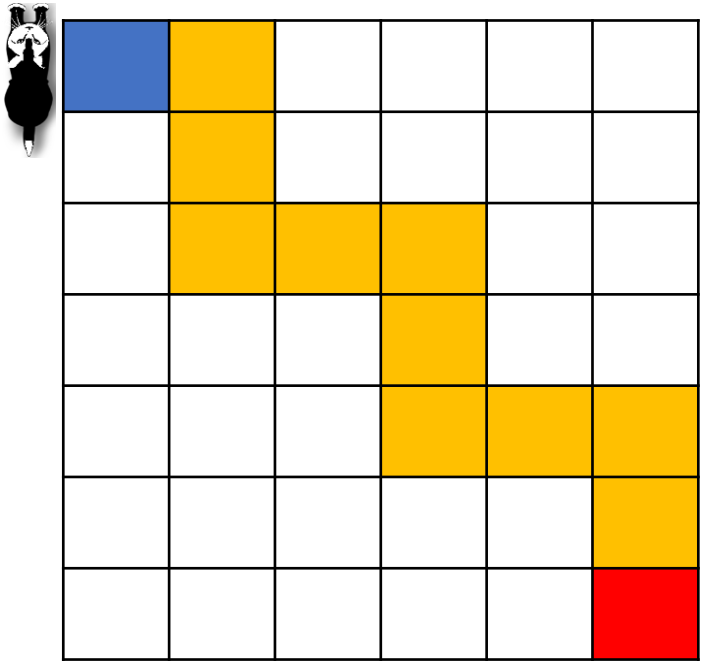
Soru

- Verilen haritada mavi kutudan içeri girip sarı yolda ilerleyerek kırmızı kutu üzerinde durulması isteniyor. Buna uygun olan algoritmayı yazınız.
- Bilgisayar her adımında aşağı, yukarı, sağa ya da sola doğru gidebilir.

- Çözüm:

1. İki adım sağa git
2. İki adım aşağı git
3. İki adım sağa git
4. İki adım aşağı git
5. İki adım sağa git
6. İki adım aşağı git

Adımların benzerliğini fark ettiniz mi?



Soru

- Verilen haritada mavi kutudan içeri girip sarı yolda ilerleyerek kırmızı kutu üzerinde durulması isteniyor. Buna uygun olan algoritmayı yazınız.
- Bilgisayar her adımında aşağı, yukarı, sağa ya da sola doğru gidebilir.

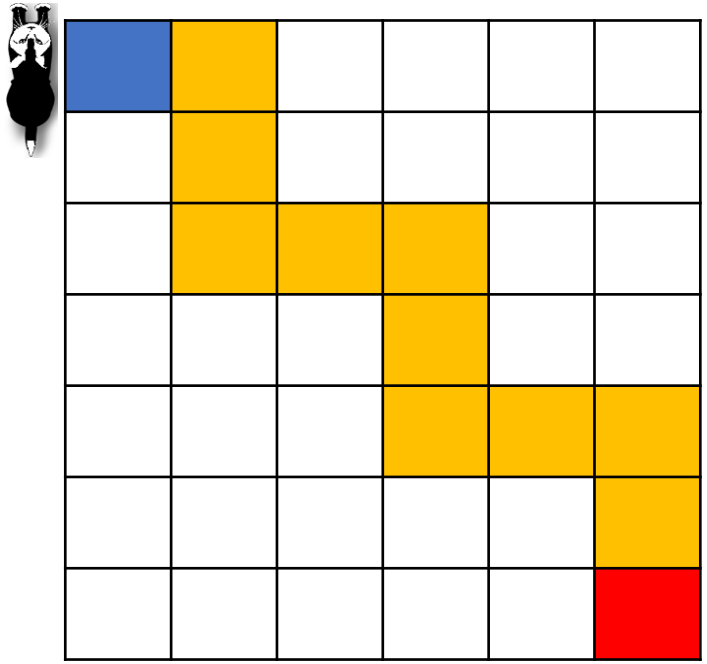
- Alternatif çözüm:

3 kere şu komutu tekrar et:

İki adım sağa git

İki adım aşağı git

Bilgisayar komutları tekrar etmeyi biliyor mu?



Nasıl çözeriz?

- Bilgisayar kaç defa işlem yaptığını sayabilir, ama işlemleri nasıl tekrar edeceğini bilmez.
- İşlemleri nasıl tekrar edeceğini biz tarif edebilir miyiz? Nasıl?

- Önce ilk çözümde işlemlerin kaç defa tekrar ettiğini sayalım:

1. BAŞLA
2. SAYAC = 0 // Sayı saymak için bir değişken tanımladık

SAYAC
0

Nasıl çözeriz?

- Bilgisayar kaç defa işlem yaptığını sayabilir, ama işlemleri nasıl tekrar edeceğini bilmez.
- İşlemleri nasıl tekrar edeceğini biz tarif edebilir miyiz? Nasıl?

- Önce ilk çözümde işlemlerin kaç defa tekrar ettiğini sayalım:

1. BAŞLA
2. SAYAC = 0 // Sayı saymak için bir değişken tanımladık
3. İki adım sağa git, İki adım aşağı git
4. SAYAC = SAYAC + 1

SAYAC
1

Nasıl çözeriz?

- Bilgisayar kaç defa işlem yaptığını sayabilir, ama işlemleri nasıl tekrar edeceğini bilmez.
- İşlemleri nasıl tekrar edeceğini biz tarif edebilir miyiz? Nasıl?

- Önce ilk çözümde işlemlerin kaç defa tekrar ettiğini sayalım:

1. BAŞLA
2. SAYAC = 0 // Sayı saymak için bir değişken tanımladık
3. İki adım sağa git, İki adım aşağı git
4. SAYAC = SAYAC + 1
5. İki adım sağa git, İki adım aşağı git
6. SAYAC = SAYAC + 1

SAYAC
2

Nasıl çözeriz?

- Bilgisayar kaç defa işlem yaptığını sayabilir, ama işlemleri nasıl tekrar edeceğini bilmez.
- İşlemleri nasıl tekrar edeceğini biz tarif edebilir miyiz? Nasıl?

- Önce ilk çözümde işlemlerin kaç defa tekrar ettiğini sayalım:

1. BAŞLA
2. SAYAC = 0 // Sayı saymak için bir değişken tanımladık
3. İki adım sağa git, İki adım aşağı git
4. SAYAC = SAYAC + 1
5. İki adım sağa git, İki adım aşağı git
6. SAYAC = SAYAC + 1
7. İki adım sağa git, İki adım aşağı git
8. SAYAC = SAYAC + 1
9. BİTİR

SAYAC
3

SAYAC bilgisinden nasıl faydalanabilirim?

Sayaç bilgisini kullanmak

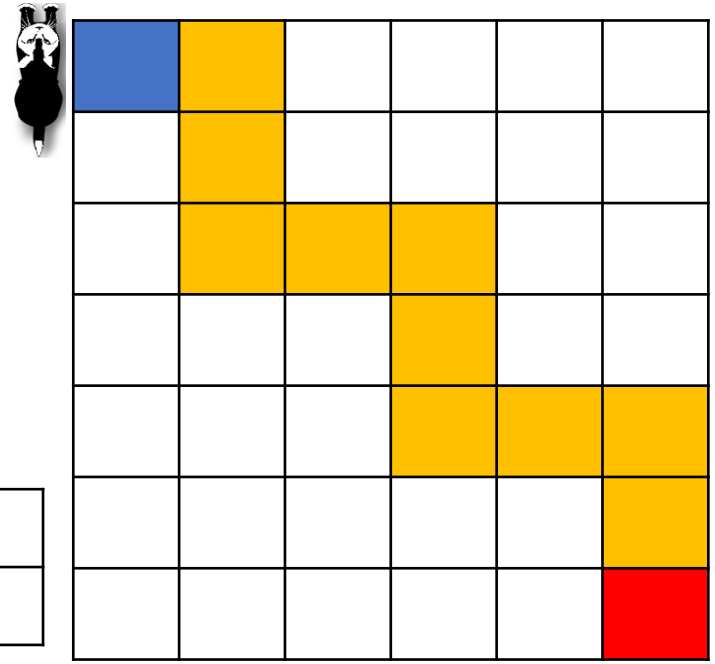
- Bir işlemin tekrarlı olarak kaç defa yapıldığını sayabiliyoruz.
- Elde etmek istediğimiz tekrar adedine ulaşılmadıkça bilgisayarı önceki adımlara geri döndürüp işlemleri tekrar ettirebiliriz.

1. BAŞLA
2. SAYAC = 0 // Sayı saymak için bir değişken tanımladık
3. İki adım sağa git, İki adım aşağı git
4. SAYAC = SAYAC + 1
5. EĞER SAYAC < 3 İSE 3. ADIMA GİT
6. BİTİR

Sayaç bilgisini kullanmak

- Bir işlemin tekrarlı olarak kaç defa yapıldığını sayabiliyoruz.
- Elde etmek istediğimiz tekrar adedine ulaşılmadıkça bilgisayarı önceki adımlara geri döndürüp işlemleri tekrar ettirebiliriz.

1. BAŞLA
2. SAYAC = 0
3. İki adım sağa git, İki adım aşağı git
4. SAYAC = SAYAC + 1
5. EĞER SAYAC < 3 İSE 3. ADIMA GİT
6. BİTİR



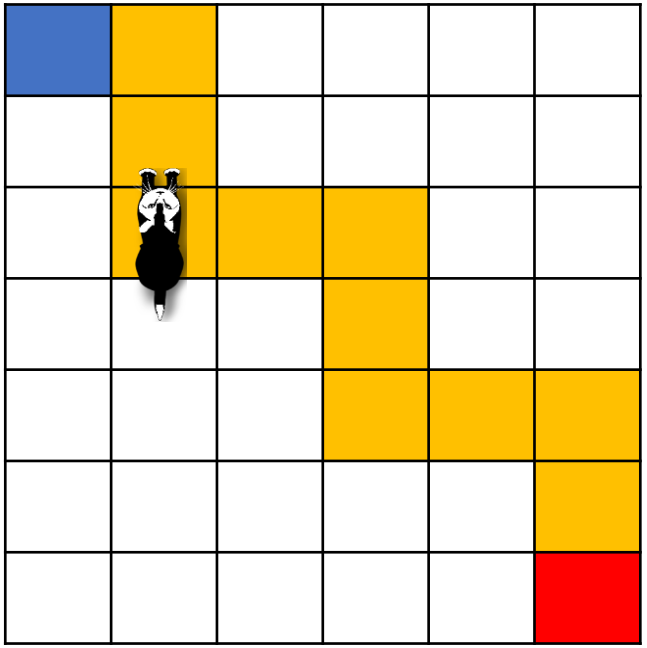
SAYAC
0

Sayaç bilgisini kullanmak

- Bir işlemin tekrarlı olarak kaç defa yapıldığını sayabiliyoruz.
- Elde etmek istediğimiz tekrar adedine ulaşılmadıkça bilgisayarı önceki adımlara geri döndürüp işlemleri tekrar ettirebiliriz.

1. BAŞLA
2. SAYAC = 0
3. İki adım sağa git, İki adım aşağı git
4. SAYAC = SAYAC + 1
5. EĞER SAYAC < 3 İSE 3. ADIMA GİT
6. BİTİR

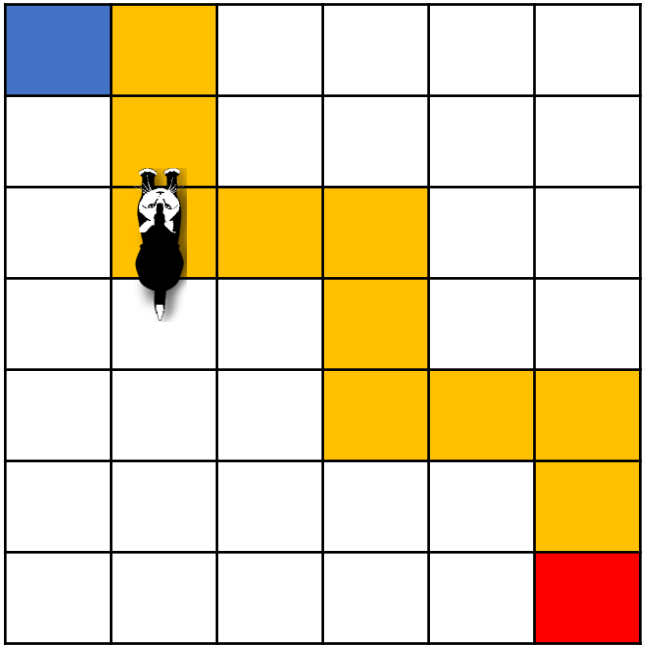
SAYAC
0



Sayaç bilgisini kullanmak

- Bir işlemin tekrarlı olarak kaç defa yapıldığını sayabiliyoruz.
- Elde etmek istediğimiz tekrar adedine ulaşılmadıkça bilgisayarı önceki adımlara geri döndürüp işlemleri tekrar ettirebiliriz.

1. BAŞLA
2. SAYAC = 0
3. İki adım sağa git, İki adım aşağı git
4. SAYAC = SAYAC + 1
5. EĞER SAYAC < 3 İSE 3. ADIMA GİT
6. BİTİR

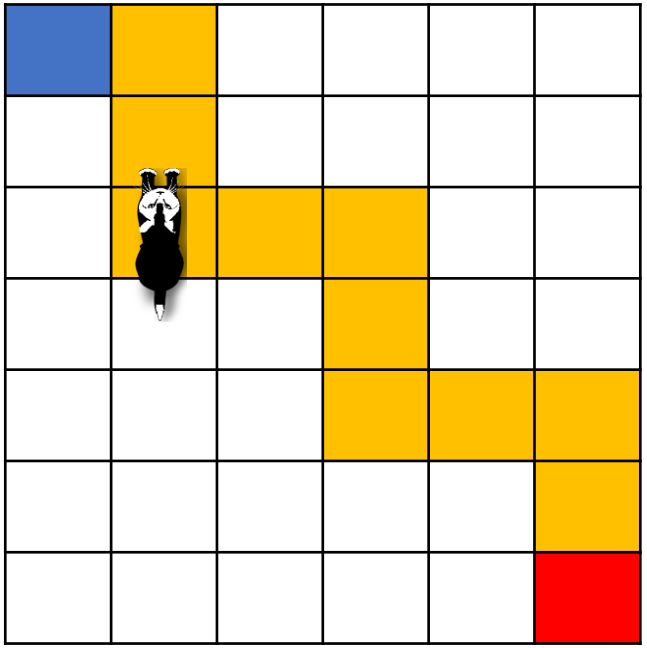


SAYAC
1

Sayaç bilgisini kullanmak

- Bir işlemin tekrarlı olarak kaç defa yapıldığını sayabiliyoruz.
- Elde etmek istediğimiz tekrar adedine ulaşılmadıkça bilgisayarı önceki adımlara geri döndürüp işlemleri tekrar ettirebiliriz.

1. BAŞLA
2. SAYAC = 0
3. İki adım sağa git, İki adım aşağı git
4. SAYAC = SAYAC + 1
5. EĞER SAYAC < 3 İSE 3. ADIMA GİT
6. BİTİR



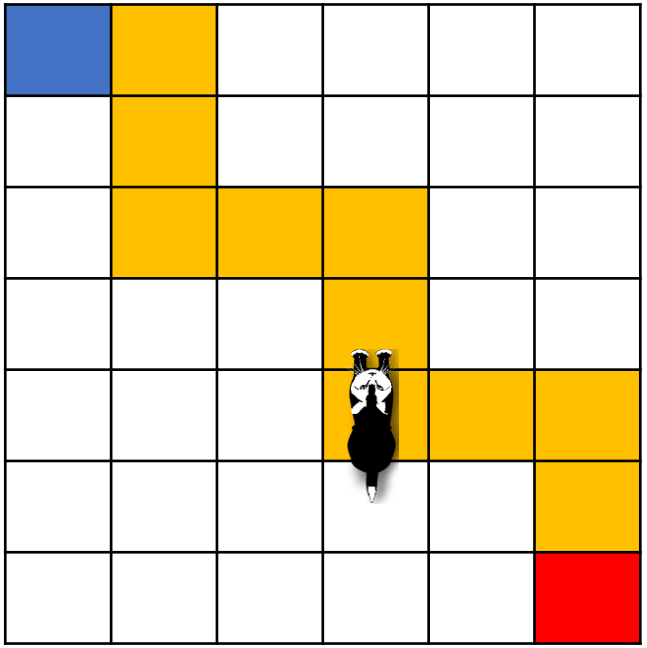
SAYAC
1

Sayaç bilgisini kullanmak

- Bir işlemin tekrarlı olarak kaç defa yapıldığını sayabiliyoruz.
- Elde etmek istediğimiz tekrar adedine ulaşılmadıkça bilgisayarı önceki adımlara geri döndürüp işlemleri tekrar ettirebiliriz.

1. BAŞLA
2. SAYAC = 0
3. İki adım sağa git, İki adım aşağı git
4. SAYAC = SAYAC + 1
5. EĞER SAYAC < 3 İSE 3. ADIMA GİT
6. BİTİR

SAYAC
1

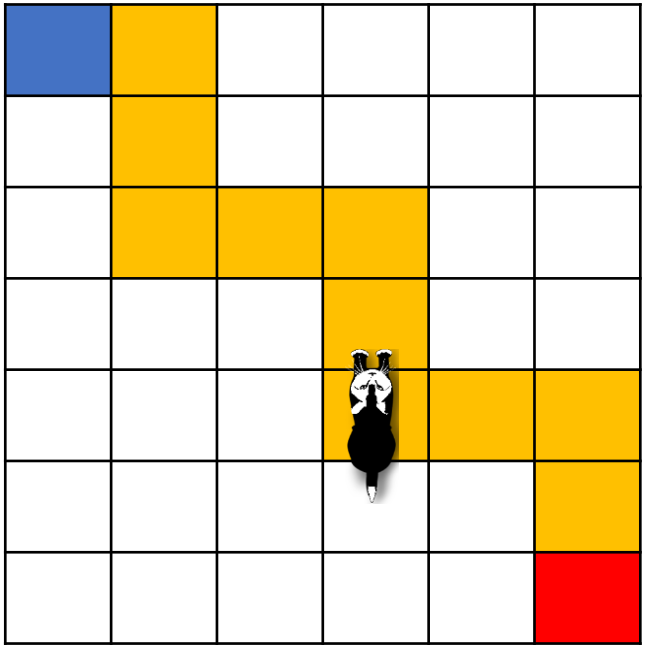


Sayaç bilgisini kullanmak

- Bir işlemin tekrarlı olarak kaç defa yapıldığını sayabiliyoruz.
- Elde etmek istediğimiz tekrar adedine ulaşılmadıkça bilgisayarı önceki adımlara geri döndürüp işlemleri tekrar ettirebiliriz.

1. BAŞLA
2. SAYAC = 0
3. İki adım sağa git, İki adım aşağı git
4. SAYAC = SAYAC + 1
5. EĞER SAYAC < 3 İSE 3. ADIMA GİT
6. BİTİR

SAYAC
2

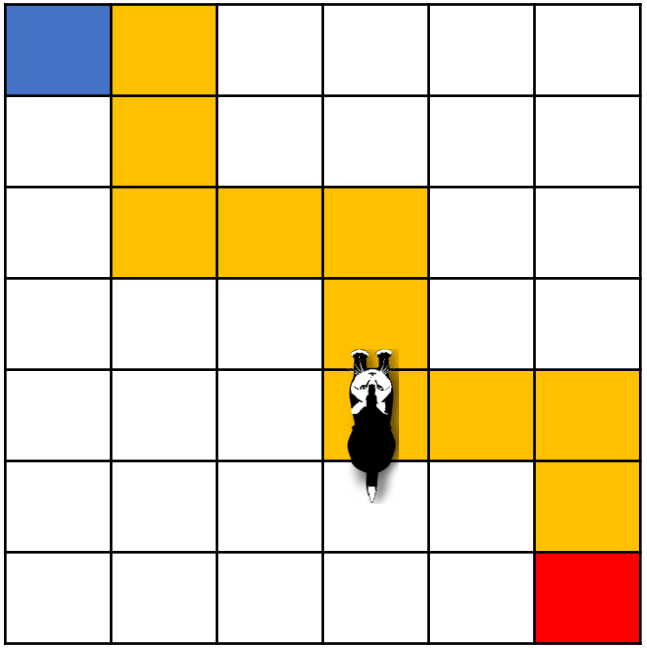


Sayaç bilgisini kullanmak

- Bir işlemin tekrarlı olarak kaç defa yapıldığını sayabiliyoruz.
- Elde etmek istediğimiz tekrar adedine ulaşılmadıkça bilgisayarı önceki adımlara geri döndürüp işlemleri tekrar ettirebiliriz.

1. BAŞLA
2. SAYAC = 0
3. İki adım sağa git, İki adım aşağı git
4. SAYAC = SAYAC + 1
5. EĞER SAYAC < 3 İSE 3. ADIMA GİT
6. BİTİR

SAYAC
2

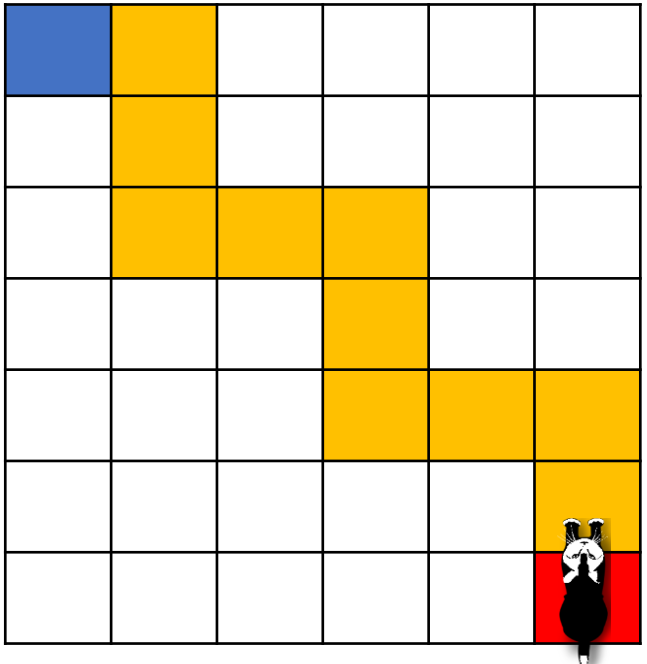


Sayaç bilgisini kullanmak

- Bir işlemin tekrarlı olarak kaç defa yapıldığını sayabiliyoruz.
- Elde etmek istediğimiz tekrar adedine ulaşılmadıkça bilgisayarı önceki adımlara geri döndürüp işlemleri tekrar ettirebiliriz.

1. BAŞLA
2. SAYAC = 0
3. İki adım sağa git, İki adım aşağı git
4. SAYAC = SAYAC + 1
5. EĞER SAYAC < 3 İSE 3. ADIMA GİT
6. BİTİR

SAYAC
2

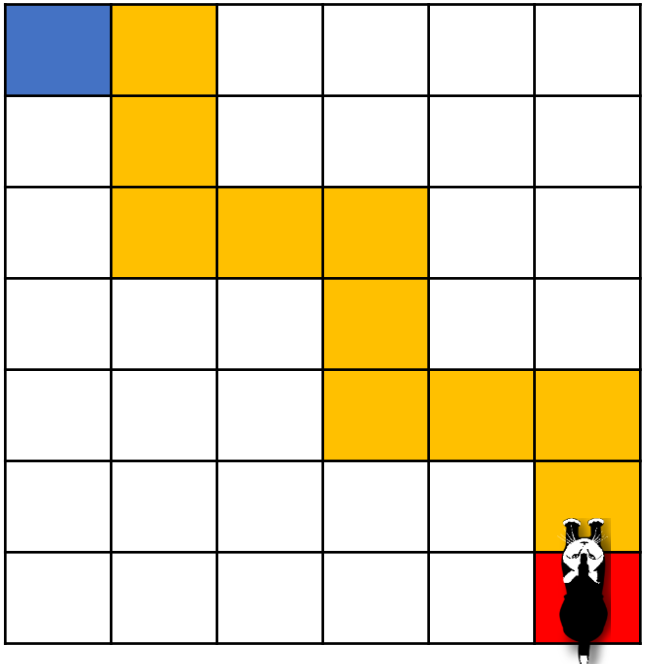


Sayaç bilgisini kullanmak

- Bir işlemin tekrarlı olarak kaç defa yapıldığını sayabiliyoruz.
- Elde etmek istediğimiz tekrar adedine ulaşılmadıkça bilgisayarı önceki adımlara geri döndürüp işlemleri tekrar ettirebiliriz.

1. BAŞLA
2. SAYAC = 0
3. İki adım sağa git, İki adım aşağı git
4. SAYAC = SAYAC + 1
5. EĞER SAYAC < 3 İSE 3. ADIMA GİT
6. BİTİR

SAYAC
3

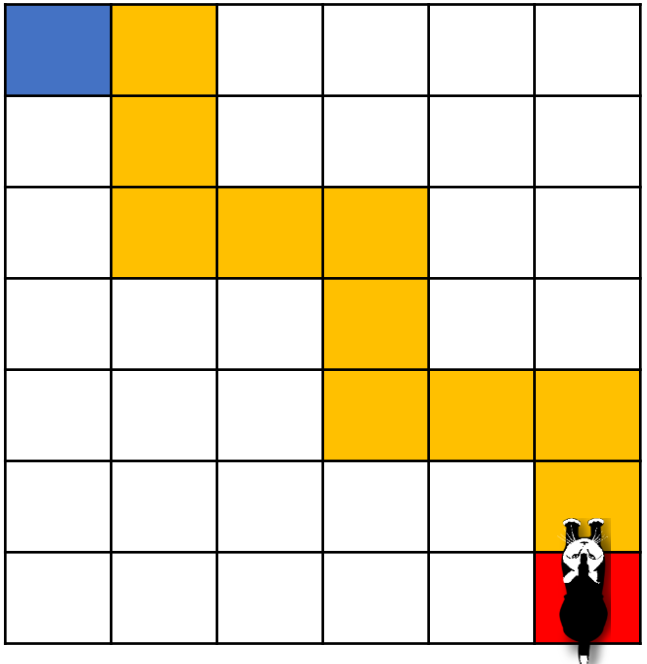


Sayaç bilgisini kullanmak

- Bir işlemin tekrarlı olarak kaç defa yapıldığını sayabiliyoruz.
- Elde etmek istediğimiz tekrar adedine ulaşılmadıkça bilgisayarı önceki adımlara geri döndürüp işlemleri tekrar ettirebiliriz.

1. BAŞLA
2. SAYAC = 0
3. İki adım sağa git, İki adım aşağı git
4. SAYAC = SAYAC + 1
5. EĞER SAYAC < 3 İSE 3. ADIMA GİT
6. BİTİR

SAYAC
3



Sayaç bilgisini kullanmak

- Bir işlemin tekrarlı olarak kaç defa yapıldığını sayabiliyoruz.
- Elde etmek istediğimiz tekrar adedine ulaşılmadıkça bilgisayarı önceki adımlara geri döndürüp işlemleri tekrar ettirebiliriz.

1. BAŞLA
2. SAYAC = 0
3. İki adım sağa git, İki adım aşağı git
4. SAYAC = SAYAC + 1
5. EĞER SAYAC < 3 İSE 3. ADIMA GİT
6. BİTİR

SAYAC
3