Bilgisayar Programlama-I

Ahmet Haşim Yurttakal

İçerik

- Algoritma Giriş
  - Algoritma Nedir?
  - Algoritma Tasarımı
  - Algoritmaların Temel Özellikleri
- Bazı Terimler
  - Makine Dili
  - Derleyici-Yorumlayıcı
- Algoritma Gösterimi
  - Konuşma Dili
  - Sözde Kod
  - Akış Diyagramları
- Uygulamalar

Bu sunum YÖK dersleri platformu üzerindeki, Anadolu Üniversitesi Bilgisayar ve Programlamaya Giriş, Anadolu Üniversitesi Algoritmalar ve Programlama, Atatürk Üniversitesi Programlama Temelleri dokümanlarından hazırlanmıştır.

ALGORİTMAYA NEDİR?

Algoritma

- Algoritma, bir problemin çözümü için net olarak tanımlanmış iş adımları listesidir.
- Algoritmanın her adımı programlama dilinde yazıldığında bilgisayar tarafından yürütülebilen açık bir talimat olmalıdır.
- Adımların sırası önemlidir; çünkü çoğu programlama dilinde işlemler programda bulundukları sıraya uygun olarak çalıştırılır.

Örnek

- Hesabın bulunduğu bankaya ait bir ATM'ye gidilir.
- ATM önündeki bekleme kuyruğunu girilir.
- İşlem sırası gelene kadar kuyrukta beklenir.
- İşlem sırası geldiğinde, bankamatik kartı ATM'nin kart haznesine takılır.
- Bankamatik kartına ait şifre girilir ve "Giriş" tuşuna basılır.
- Para çekme menüsüne erişilir.
- Çekilecek nakit tutarı belirlenir ve "Devam" tuşuna basılır.
- ATM, bankamatik kartını kart haznesinden çıkartır.
- Bankamatik kartı ATM'den geri alınır.
- ATM, nakit parayı para haznesine doldurur.
- Nakit para ATM'den alınır.
- Para çekme işlemi tamamlanarak, işlem kuyruğundan çıkılır.

Algoritma Tasarımı

- Algoritma tasarımında, öncelikle sorunu anlamak önemlidir. Aşağıdaki sorulara cevap bulunması gerekir
  - Nihai hedef nedir?
  - Hedefe giden yolda hangi işlemlerin yapılması gerekir?
- Sorunu Anlamak;
  - Problemin girdileri neler?
  - Çözüm çıktıları ne olacak?
  - Talimatlar hangi sırayla gerçekleştirilecek?
  - Problemde hangi kararların alınması gerekiyor?
  - Çözümde tekrarlayan kısımlar var mı?

Soru

- Çay demlemesini bilmeyen birisi için anlayacağı şekilde çay demleme algoritmasını yazınız.

Algoritmaların Yapısı

- Değer Atama
- Aritmatiksel İşlemler (Toplama, Çıkarma, Çarpma, Bölme, Mod)
- Mantıksal İşlemler (Büyüktür, Küçüktür, Büyük Eşit, Küçük Eşit gibi)

Algoritmaların Temel Özellikleri

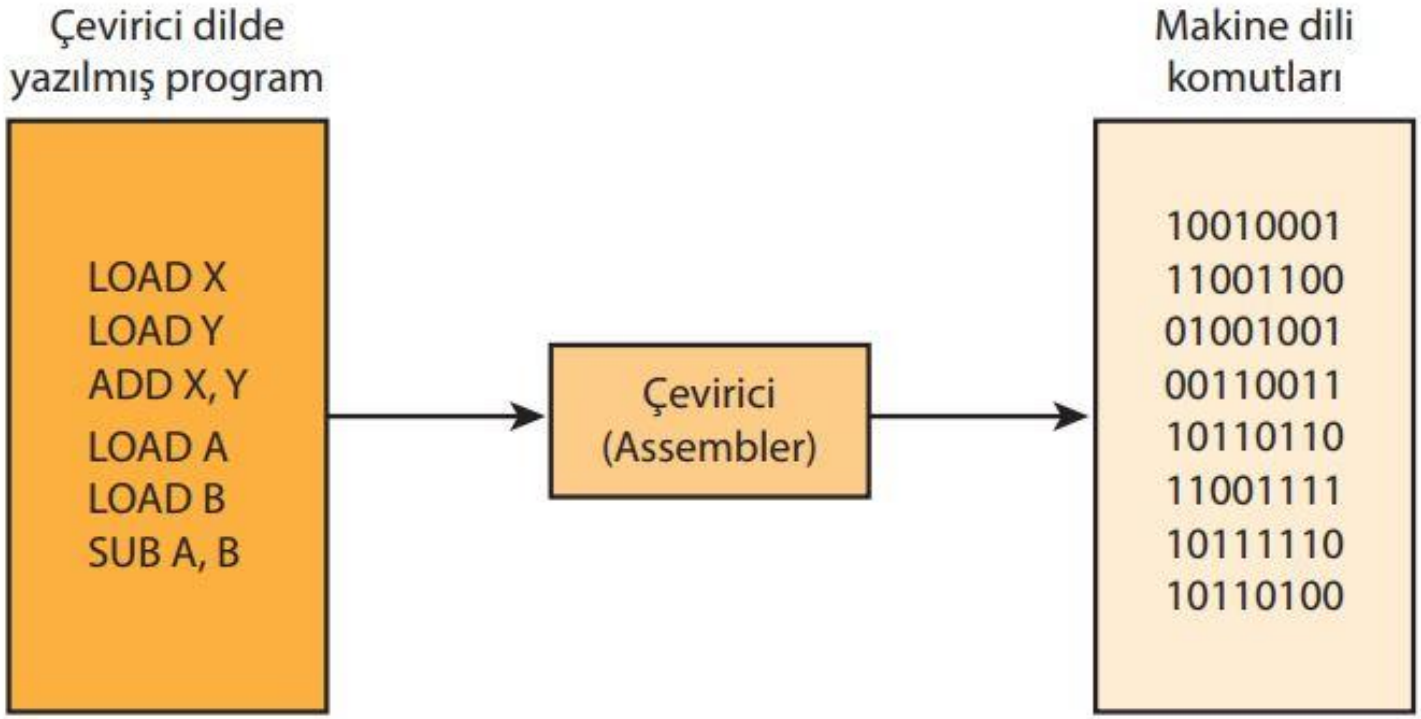
- *Girdi ve Çıktı Bilgisi:* Algoritmalarda girdi ve çıktı bilgileri olmalıdır. Girdi bilgisi algoritmaya dışarıdan verilirken, çıktı bilgisi ise algoritma içerisinde üretilir.
- *Açıklık:* Algoritmayı oluşturan adımlar doğru ve kesin bir şekilde tanımlanmalıdır.
- *Doğruluk:* Farklı girdi bilgileri ile çalışabilen algoritmalar, her girdi için doğru bir çıktı üretmelidir.
- *Sonluluk:* Algoritmaların daima bir sonu olmalıdır. Girilen veri boyutundan bağımsız bir şekilde, algoritma adımları farklı bir aşamaya geçebilmeli veya sonlanmalıdır. Algoritma adımları gerçekleştirilirken, algoritma sonsuz döngüye girmemelidir.
- *Verimlilik:* Algoritmayı oluşturan adımlar, yapılan iş için kabul edilebilir bir süre içerisinde tamamlanmalıdır.
- *Genellik:* Bir algoritma, aynı türdeki problemlerin hepsine uygulanabilir olmalıdır.

BAZI TERİMLER

Makine Dili

- Bilgisayarlar, yalnızca makine dilinde yazılmış programları çalıştırabilir.
- Makine dili komutları 0 ve 1 değerlerinden oluşan, insan tarafından okunması ve anlaşılması kolay olmayan komutlardır.
- Bu komutları kullanarak binlerce satırlık programlar yazmak ve gerektiği durumlarda bu programları incelemek, oldukça zahmetli, zaman gerektiren ve hataya müsait bir iştir
- Bu zorluğun üstesinden gelebilmek ve programcıların işini kolaylaştırmak için çevirici diller (assembly) geliştirildi.
- Bu dillerde makine dili komutlarının doğrudan kullanılması yerine, komutları ifade eden kısa kelimelere geçiş yapıldı.
- Örnek olarak, toplama işlemi için ADD, çıkarma işlemi için SUB, çarpma işlem için MUL, veriyi bellekte farklı bir alana taşımak için MOV gibi kısa komutlar, makine dili komutlarının yerine geçti.

Assembly


Çevirici dilde yazılmış program
LOAD X
LOAD Y
ADD X, Y
LOAD A
LOAD B
SUB A, B
Çevirici (Assembler)
Makine dili komutları
10010001
11001100
01001001
00110011
10110110
11001111
10111110
10110100

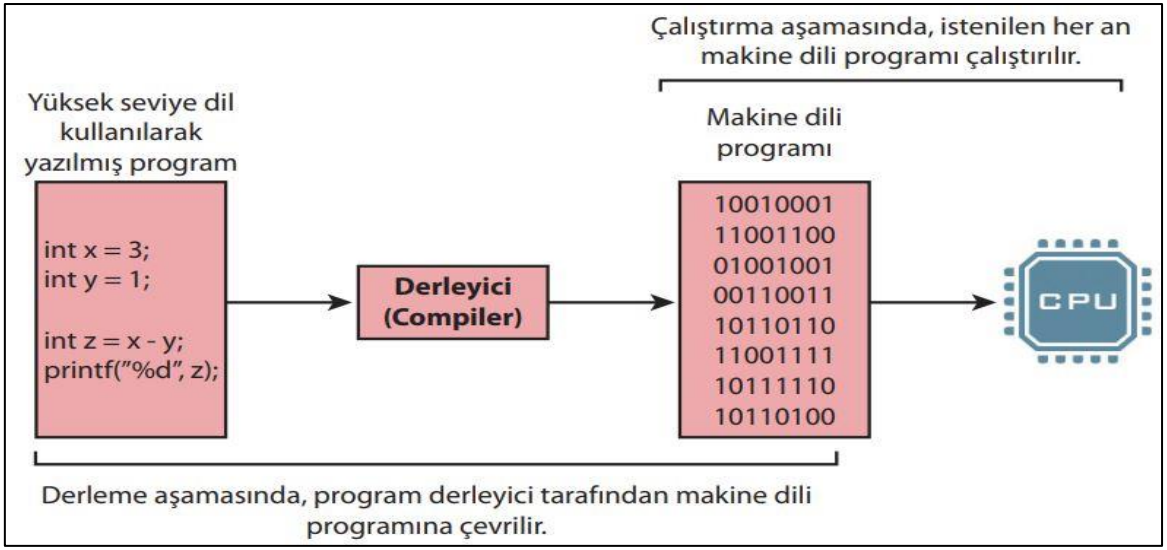
Yüksek Seviye Diller

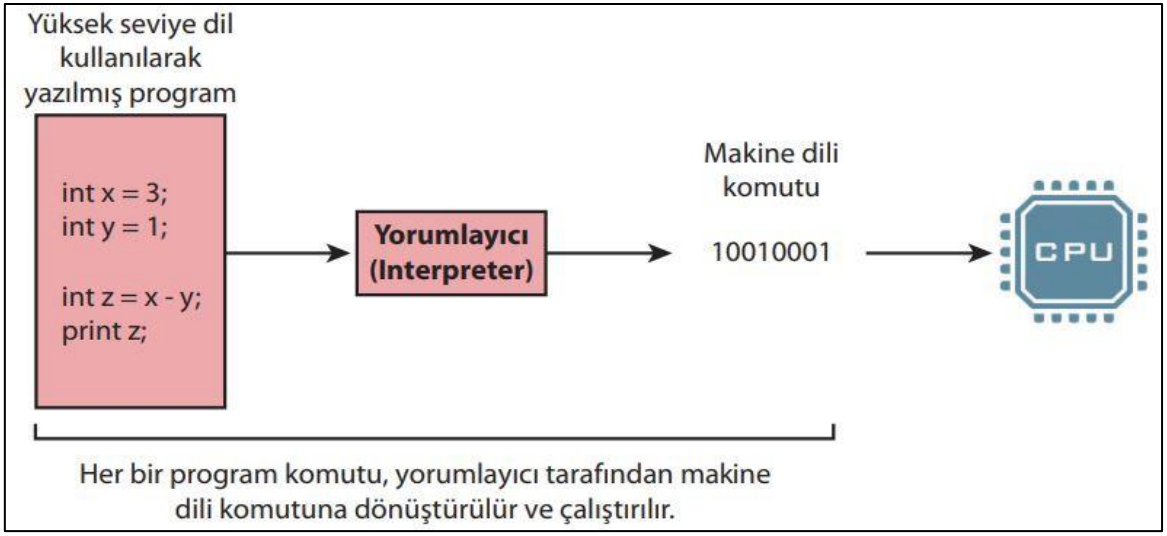
- Yapı itibariyle makine diline oldukça benzeyen çevirici diller, alt seviye programlama dilleri kategorisine girer.
- Alt seviye programlama dillerinde yaşanan zorlukların önlenebilmesi için yüksek seviye programlama dilleri oluşturulmuştur.
- C, C++, C#, Java, Python

Derleyiciler-Yorumlayıcılar

- Bilgisayar sistemlerinde merkezî işlem birimi, yalnızca makine dili komutlarını algılayabilir. Dolayısıyla, yüksek seviye programlama dili kullanılarak yazılan programların makine diline çevrilmesi gerekmektedir.
- **Derleyici:** Bir programlama dilinde yazılmış kodu, genellikle makine diline çevirmek için kullanılan bilgisayar programıdır. Derleyicilerin temel görevi, çalışır bir program elde etmektir.
- **Yorumlayıcı:** Bir programlama dilinde yazılmış kodu, doğrudan çalıştıran bilgisayar programıdır. Yorumlayıcılarda dönüştürme ve çalıştırma işlemleri geçekleştirilir.

Derleyiciler-Yorumlayıcılar





Derleyici	Yorumlayıcı
• Programın tamamını makine diline çevirir. • Kaynak kodun analizi daha uzun sürer. • Çalışma hızı yüksektir. • Programın tamamını tarandıktan sonra hatalar gösterilir. • Orta seviyeli nesne kodları üretilir. Bu yüzden bellek ihtiyacı daha fazladır. • Kodların sürekli derlenmesine gerek yoktur.	• Program komutlarını ayrı ayrı makine diline çevirir. • Kaynak kodun analizi kısa sürer. • Çalışma hızı düşüktür. • Hata ile karşılaşana dek dönüştürme devam eder. • Orta seviyeli nesne kodları üretilmez. Bu yüzden bellek ihtiyacı daha düşüktür. • Kodlar her seferinde dönüştürülmelidir.

Arasındaki farklar

ALGORİTMA GÖSTERİMİ

Konuşma Dili

- Bir algoritmanın açıklaması ve algoritmada yer alan adımlar, konuşma dili kuralları çerçevesinde ifade edilebilir.
- Algoritma açık ve kesin bir dille tanımlanır. Algoritmada yer alan adımlar liste halinde yazılır

Örnek

- Elimizde iki adet pozitif tamsayı vardır. Bu iki sayının ortak bölenlerinin en büyüğü bulunacaktır.

Elimizdeki sayılar 8 ve 12 olsun.

Başla
Adım 1. A = 12, B = 8
Adım 2. A % B = 4, K = 4
Adım 3. K = 4 olduğundan K != 0, A = 8, B = 4, Adım 2'ye dön
Adım 2. A % B = 2, K = 0
Adım 3. K == 0 olduğundan ortak bölenlerin en büyüğü B'nin değeri, yani 4'tür.
Bitir

SÖZDE KOD (PSEUDOCODE)

- Bir algoritma veya program oluşturulurken kullanılan, konuşma diline benzeyen ve programlama dillerinin detaylarından uzak anlatımlardır.
- Algoritmaların sözde kod ile gösterilmesinde, bir programlama diline benzeyen ifadeler kullanılır, ancak bu ifadeler bilgisayarın anlayabileceği ifadeler değildir.
- Sözde kodu okuyan bir kişi, programlama dillerinin detaylarına takılmadan, algoritmanın çalışma mantığını kavrayabilir ve koda kolayca aktarabilir.

Sözde Kod Deyimleri

- Sözde kod için çok katı kurallar listesi olmasa da sözde kodlamada yaygın olarak kullanılan deyimler gruplanarak aşağıda verilmiştir.
  - Aritmetik işlemlerde: +, -, *, / ,%, <,>,>=,<=, =,==, !=
  - Başla: BEGIN
  - Girdi: INPUT
  - Çıktı: PRINT
  - Başlatma: SET
  - Koşul: IF - THEN - ELSE – ENDIF
  - Döngü: WHILE - ENDWHILE, DO - WHILE, REPEAT - UNTIL
  - Bitir: END

Örnek 1

- Bir öğrencinin notu 50'ye eşitse veya büyükse bilgisayar ekranında "Geçti", 50'den küçükse "Kaldı" yazacak algoritmanın sözde kodu

```
BEGIN
INPUT ogrencininNotu
IF ogrencininNotu >= 50 THEN
        PRINT “Geçti”
  ELSE
        PRINT “Kaldı”
  ENDIF
END
```

Örnek 2

- 10 öğrencinin notlarını okuyup ortalamasını hesaplayacak algoritmanın sözde kodu

```
BEGIN
SET toplam = 0
SET sayac = 1
WHILE sayac <= 10
        INPUT ogrencininNotu
        toplam=toplam + ogrencininNotu
        sayac = sayac + 1
ENDWHILE
SET ortalama = toplam / 10
PRINT ortalama
END
```

Örnek 3

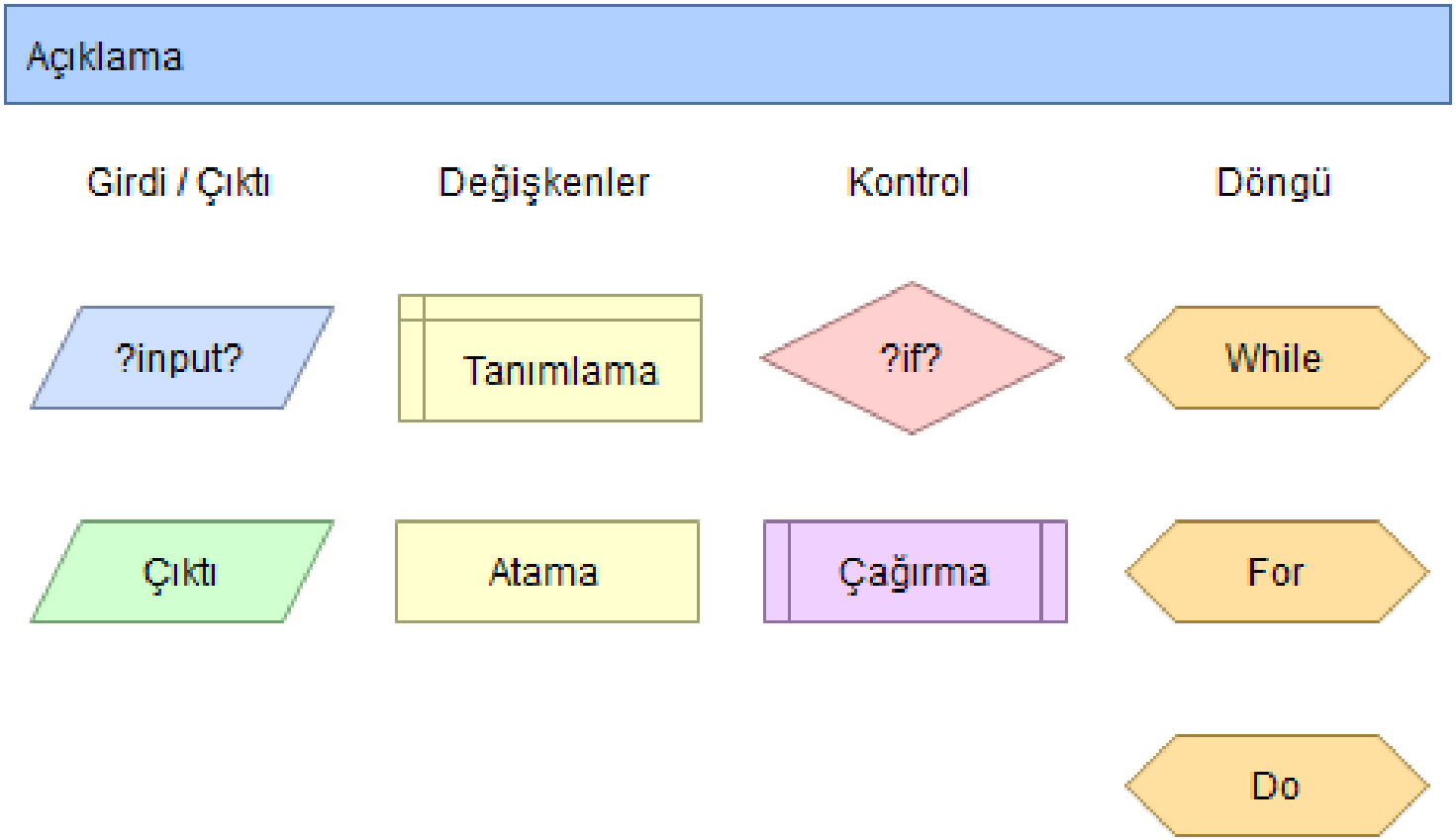
- İki pozitif tamsayının ortak bölenlerinin en büyüğünü bulmak için kullanılan Euclid algoritmasının sözde kodu;

```
begin
k = a % b
while (k ! = 0)
        a = b
        b = k
        k = a % b
endwhile
print b
end
```

Akış Diyagramları

- Akış şeması, bir algoritmanın görsel halini ifade eder. Görsellik, algoritmaların daha kolay anlaşılabilmesine olanak sağlar.
- Görselleştirme için Flowgorithm programını kullanabilirsiniz.

Diyagramlar



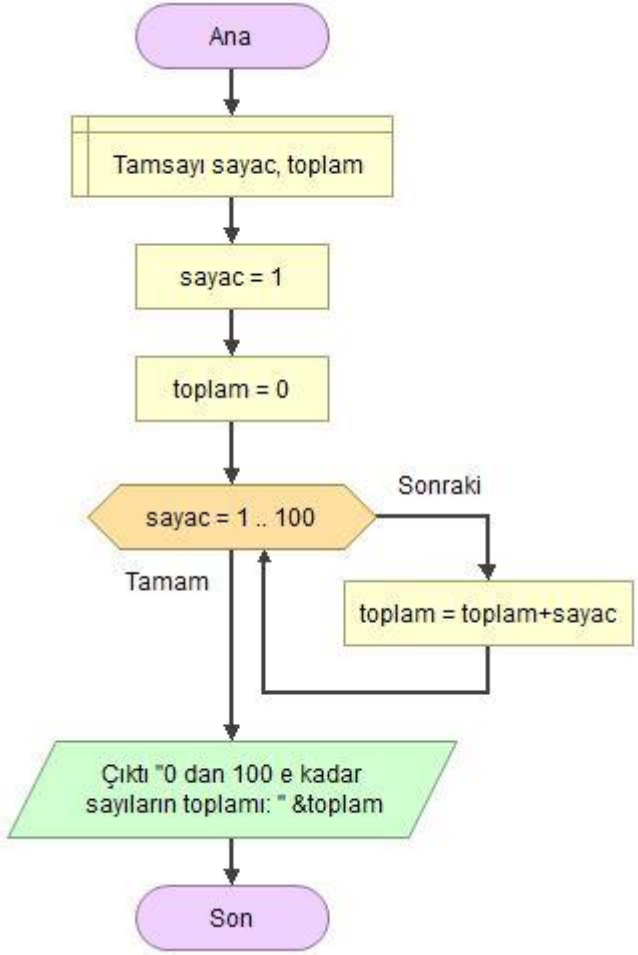
UYGULAMALAR

Örnek 1

Klavyeden girilen iki sayının toplamı

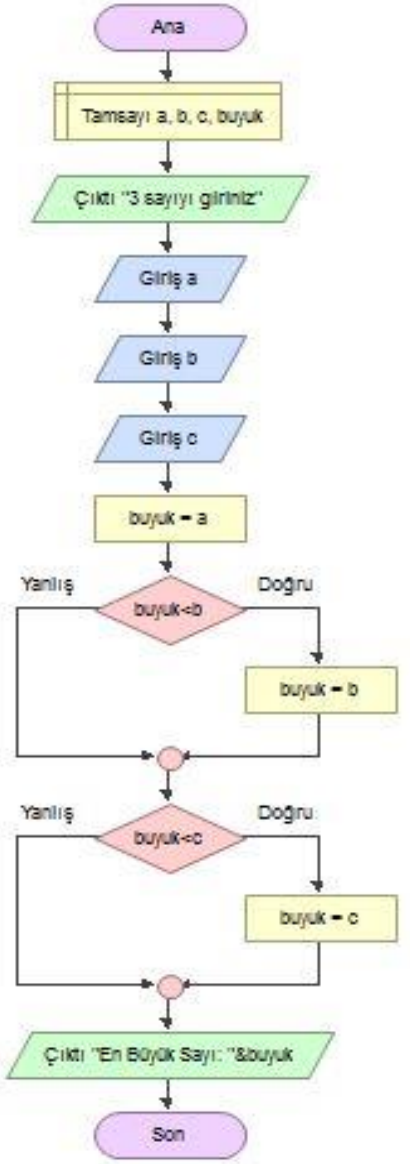
Örnek 2

0-100 arasındaki sayıların toplamı



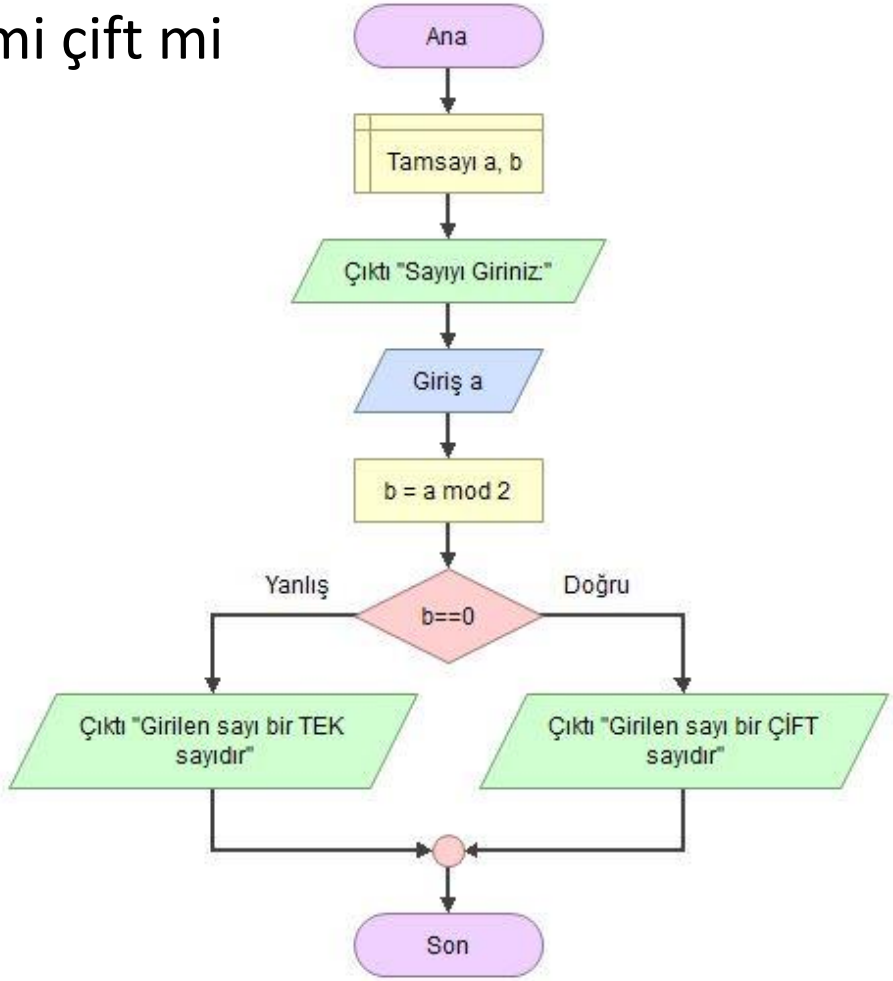
Örnek 3

Klavyeden girilen 3 sayıdan en büyüğü



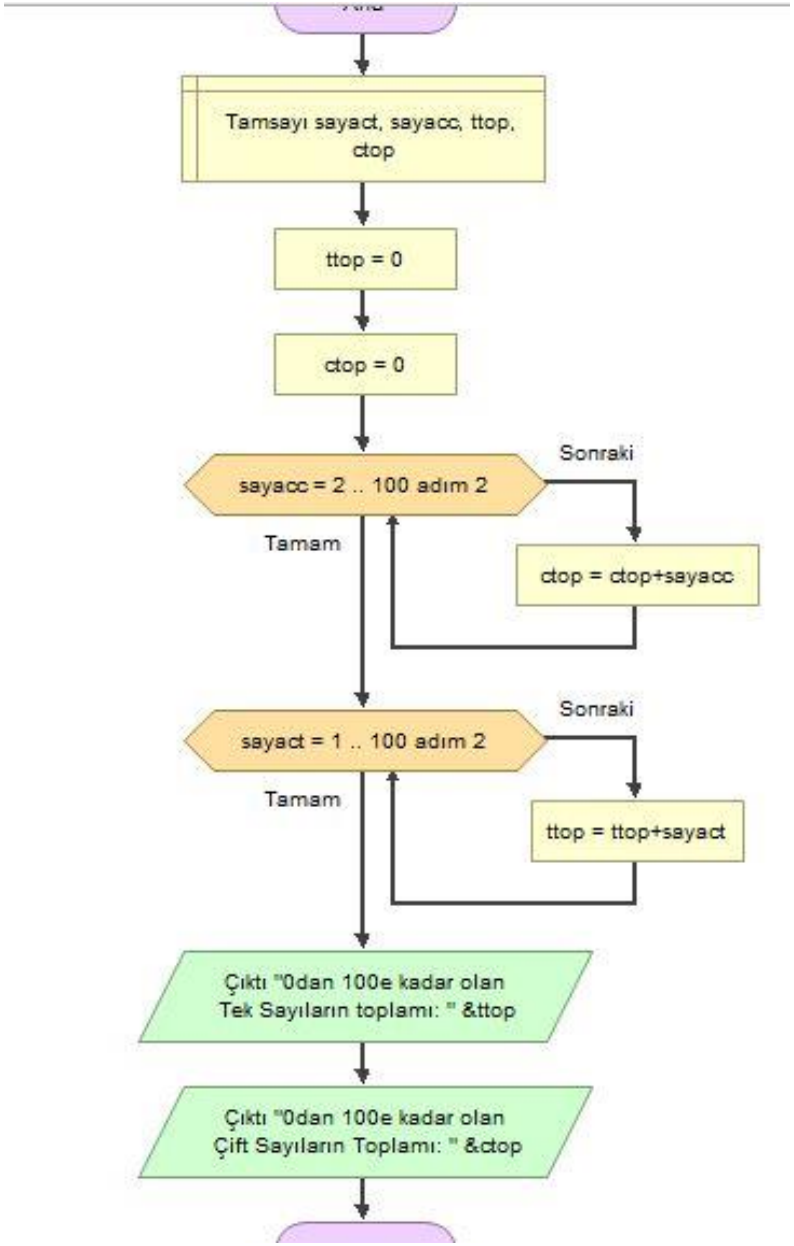
Örnek 4

Klavyeden girilen sayı tek mi çift mi



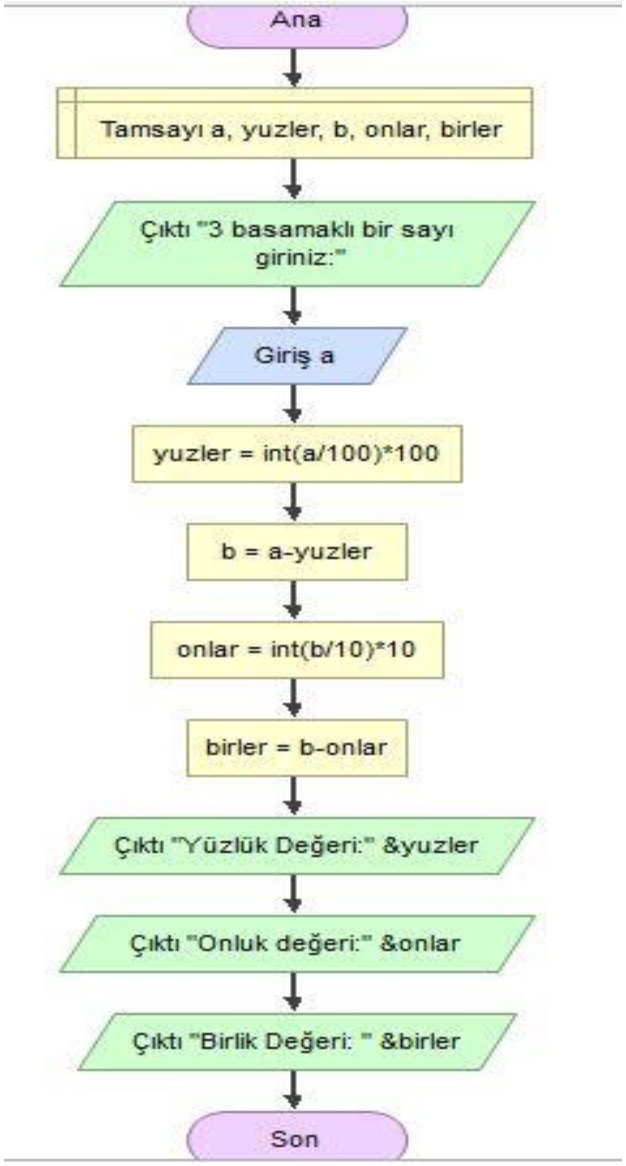
Örnek 5

0-100 arasındaki tek ve çift sayıların toplamı



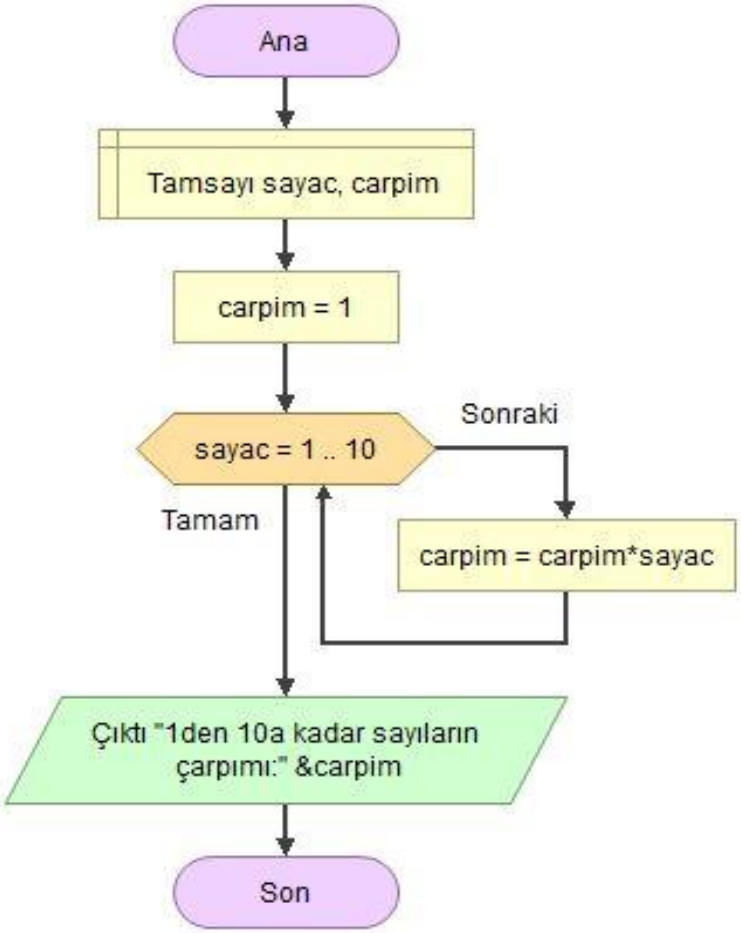
Örnek 6

Klavyeden girilen 3 basamaklı sayının basamak değerleri



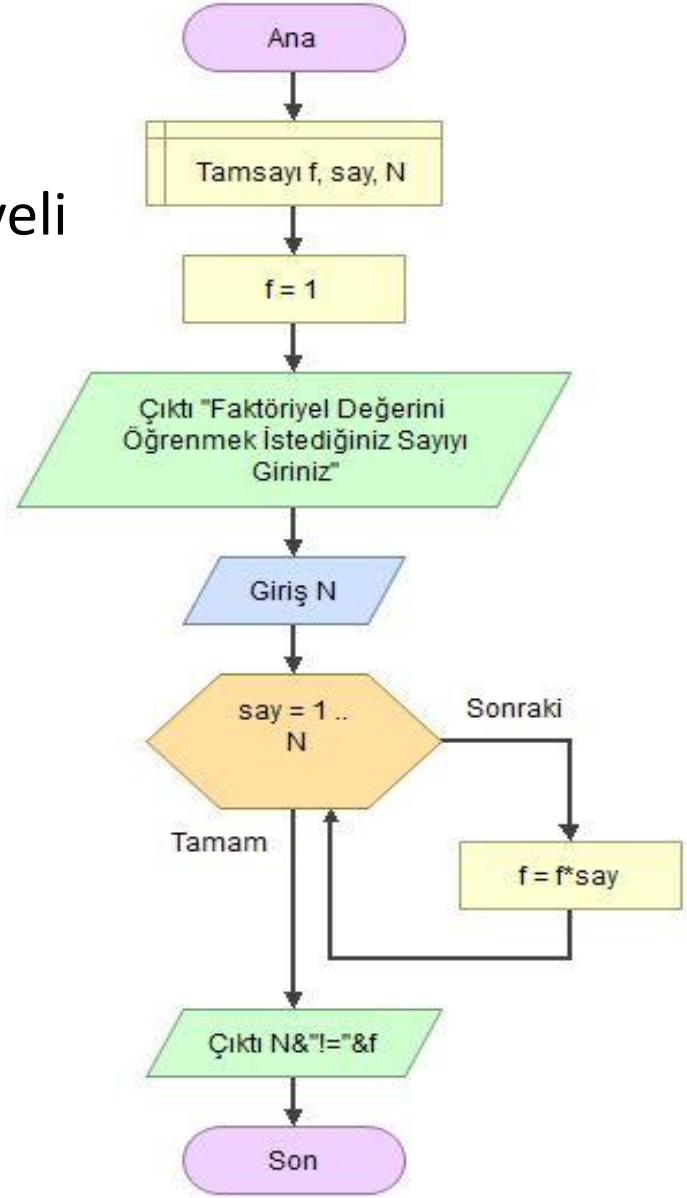
Örnek 7

1-10 arası sayıların çarpımı



Örnek 8

Klavyeden girilen bir sayının faktöriyeli



ÖDEV

SORULAR

1. İkinci dereceden iki bilinmeyeli denklemin köklerini bulan algoritmanın akış şemasını çiziniz.
2. Obeb-Okek bulan algoritmanın akış şemasını çiziniz.
3. En büyük ve en küçük dizi elemanını bulan algoritmanın akış şemasını çiziniz.
4. İki matrisin toplamını toplan algoritmanın akış şemasını çiziniz.

SON