BLGM108

1. BÖLÜM

Problem Çözme Kavramları (Algoritma ve Akış Şemaları)

Yazılım Geliştirme Adımları

1. Gereksinimlerin belirlenmesi

   Problemin ne olduğunu anlama: sorunu çözmek için ne gereklidir, çözüm ne üretmelidir, mevcut kısıtlamalar nelerdir

2. Analiz

   Girdiler üzerinde karar verme-inputs (kullanılacak veriyi belirleme),

   çıkışları belirleme-outputs (istenilen sonuçları belirleme),

   formülleri, kullanılacak denklemleri belirleme

3. Tasarım

   Bir algoritma geliştirilmesi

   (Sözdekodlar-**pseudo kod** ve akış şemaları- **flowcharts**)

4. Uygulama

   Algoritmayı kullanarak, bir programlama dilinde program yazmak

5. Test etme (doğruluğu gösterme) ve doğrulama (programın kullanıcının gereksinimlerini karşılayıp karşılamadığını belirleme)

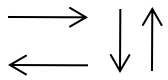
6. Bakım ve güncelleme

   Tespit edilmemiş hataları çıkarma ve programın belgeleri hazırlama

Algoritma, Pseudo Kod ve Akış Şeması (Flowchart)

- **Algoritma** Çalıştırıldığında, belirli bir sorun için çözüm üreten, belirli bir mantıksal sırayla, düzenlenmiş talimatlar kümesi.
- Algoritmayı temsil eden teknikler:
- **Pseudo Kod** Yarı resmi, sınırlı kelimeden oluşan kodlayıcı dili benzeri bir dil. Talimatlar bilgisayar diline daha yakındır.
- **Flowchart** Bir algoritmanın grafiksel gösterimidir. Geometrik şekillerin, akış hatları ile bağlanması sonucu oluşur.

Flowchart Sembolleri



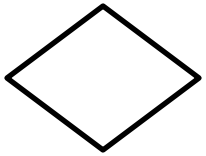
Akış hatları Blokları bağlar, akış yönünü gösterir.

Start-Begin/Stop-End: Başlangıcı ve bitişi gösterir.

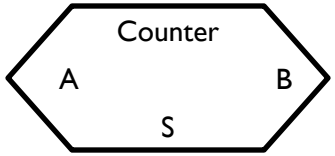
İşlem. Hesaplama gibi işlemlerin yapıldığını gösterir.

İşlem modülü Farklı bir yerde işlenen görevleri temsil eder.

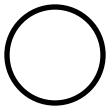
I/O: Blgisayara giriş-Input ve bilgisayardan çıkışı-output gösterir.



Karar: Karşılaştırma durumlarında kulanılır.



Atomatik-sayaçlı Döngü Sayaçlı döngüyü gösterir. Sayaç A dan başlar, B den büyük olana kadar, her seferinde S kadar artırılır.



Ayni – sayfa bağlayıcı Ayni sayfadaki flowchart kısımları birbirine bu şekille bağlanır.

Farklı-sayfa bağlayıcı Farklı sayfadaki flowchart kısımları birbirine bu şekille bağlanır.

Algoritmaların Tasarlanması

- **Yapısal Programlama**
  - Bohm ve Jacopini tarafından 1966 tanımlanmıştır. Herhangi bir algoritma sadece üç kontrol yapısıyla anlatılabilir: **sıralı(sequence), seçmeli(selection), ve yinelemeli-tekrarlama(repetition)**
- **Üstten-alta tasarım (böl ve çöz)**
  - Bir problemi küçük ana parçalara ayırıp her birini tek tek çözdükten sonra tümünü tek bir çözümde birleştirmek
  - **Programlama probleminde bir algoritmanın önemli adımları**
    1. Veriyi gir
    2. Hesaplamaları gerçekleştir
    3. Sonucu görüntüle

Yapısal Programlama

- Her algoritma üç kontrol yapısıyla oluşturulabilir. Bunlar:

  1- sıralı

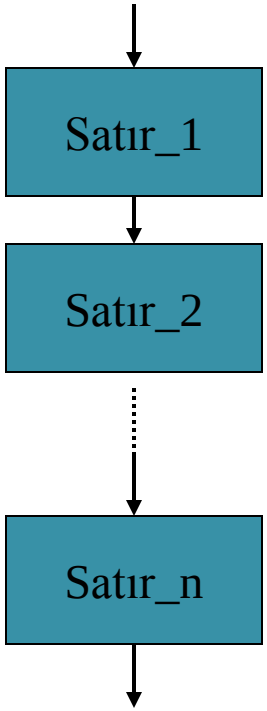
  2- seçmeli

  3- yinelemeli

Yapısal Programlama

1-Sıralı Yapı

Sıralı adımların veya satırların yazıldıkları sırada yürütülmeleri



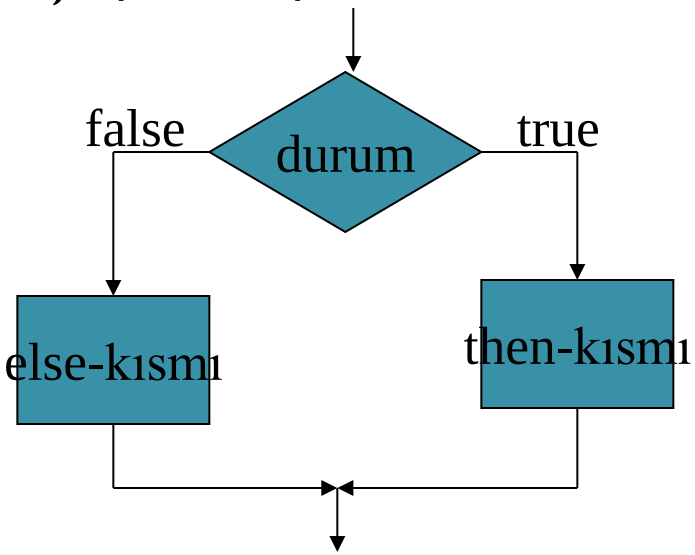

```
Satır_1
Satır_2
…
Satır_n
```

Blok

```
begin
    Satır_1
    Satır_2
    …
    Satır_n
end
```

2-Seçmeli Yapı

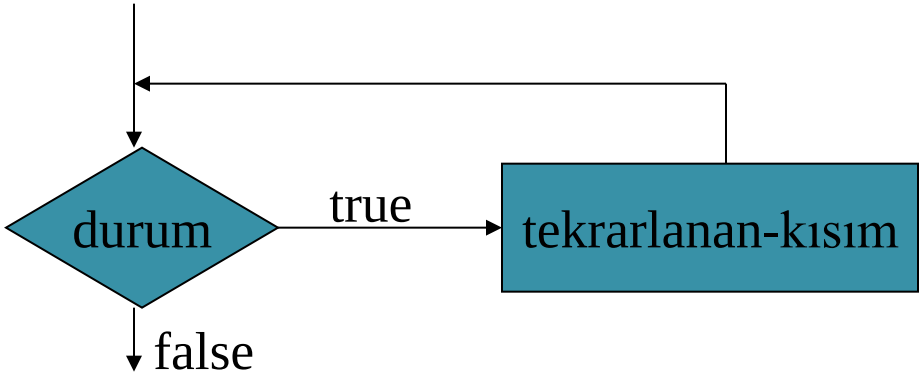
- Durumun sonucuna bağlı olarak iki farklı seçenek tanımlar
- Durum bir ifadedir (expression) değerlendirildiği zaman, Doğru (True) veya yanlış (False) sonuç verir




```
if durum
    then-kısmı
else
    else-kısmı
```

3-Yinelemeli Yapı

- Durum sağlandığı sürece bir yada daha fazla satırın tekrarlanarak yürütülmesi



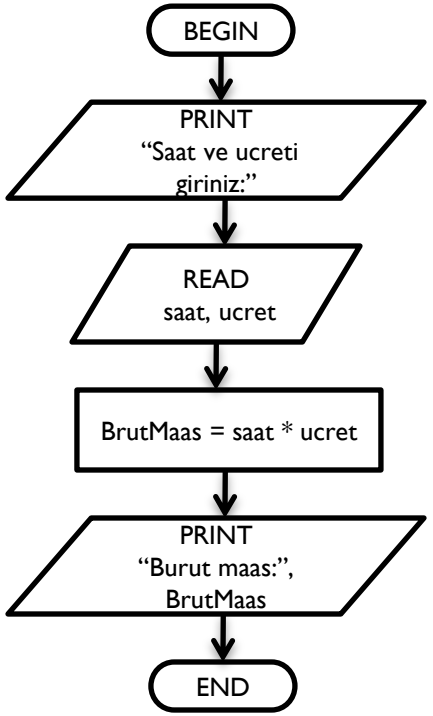

```
while durum
    tekrarlanan-kısım
end_while
```

Örnek: Maaş Hesaplama

Bir iş yerinde çalışan işçilerin brüt maaşlarını aşağıdaki formülü kullanarak hesaplayan ve ekrana yazdıran algoritmayı yazıp akış şemasını çiziniz.

Brüt Maaş = Saat X Ücret

Akış Şeması:



Algoritma:

1. BEGIN
2. PRINT "Saat ve ücreti giriniz: "
3. READ saat, ucret
4. BrutMaas = saat * ucret
5. PRINT "BrutMaas:", BrutMaas
6. END

Örnek:

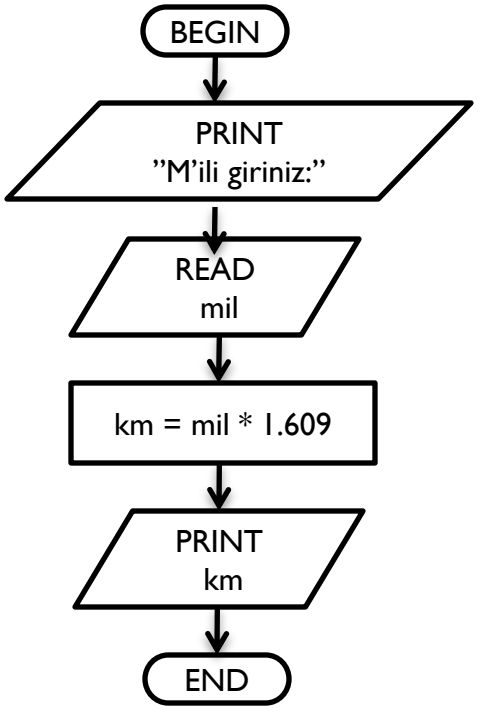
Kullanıcı tarafından girilen mil'i kilometreye çeviren algoritmayı yazıp akış şemasını çiziniz.

Formül: 1 mil = 1.609 kilometre

Algoritma:

1. BEGIN
2. PRINT "Mili giriniz:"
3. READ mil
4. km=mil* 1.609
5. PRINT km
6. END

Akış Şeması



Örnek:

Aşağıdaki problem için algoritma yazıp, akış şemasını çiziniz.

Fahrenheit sıcaklık derecesini klavyeden okuyunuz. Eğer sıcaklık derecesi -459.7 (mutlak sıfır (absolute zero)) altındaysa kullanıcıya sıcaklık derecesinin mutlak sıfırın altında olduğunu söyleyen bir hata mesajı verilecektir. Değilse, Fahrenheit sıcaklık derecesi aşağıdaki formül kullanılarak Santigrat sıcaklık derecesine çevrilecektir.

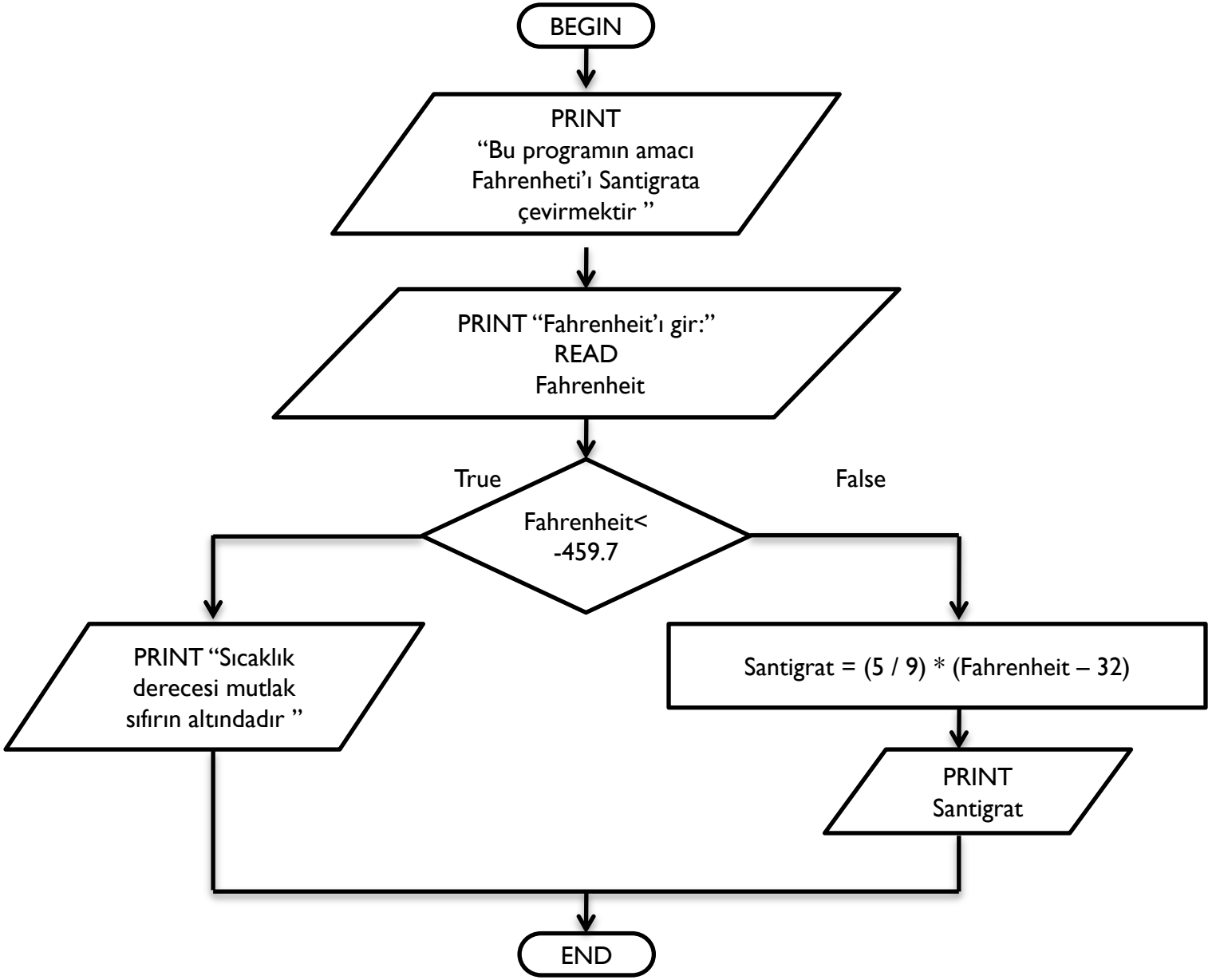
Santigrat = (5 / 9) X (Fahrenheit – 32)

Algoritma:

1. BEGIN
2. PRINT "Bu programın amacı Fahrenheit'ı Santigrata çevirmektir"
3. PRINT "Fahrenheit'ı gir:"
4. READ Fahrenheit
5. IF Fahrenheit < -459.7
   PRINT "Sıcaklık derecesi mutlak sıfırın altındadır"
   ELSE
   Santigrat = (5/9) * (Fahrenheit – 32)
   PRINT Santigrat
5. END

Programın amacını yazdırmak isteğe bağlıdır (Optional). Programı kullanacak kullanıcılar için açıklayıcı bilgi içerir.

Akış Şeması


BEGIN
PRINT
"Bu programın amacı
Fahrenheti'ı Santigrata
çevirmektir "
PRINT "Fahrenheit'ı gir:"
READ
Fahrenheit
True
False
Fahrenheit<
-459.7
PRINT "Sıcaklık
derecesi mutlak
sıfırın altındadır "
Santigrat = (5 / 9) * (Fahrenheit – 32)
PRINT
Santigrat
END

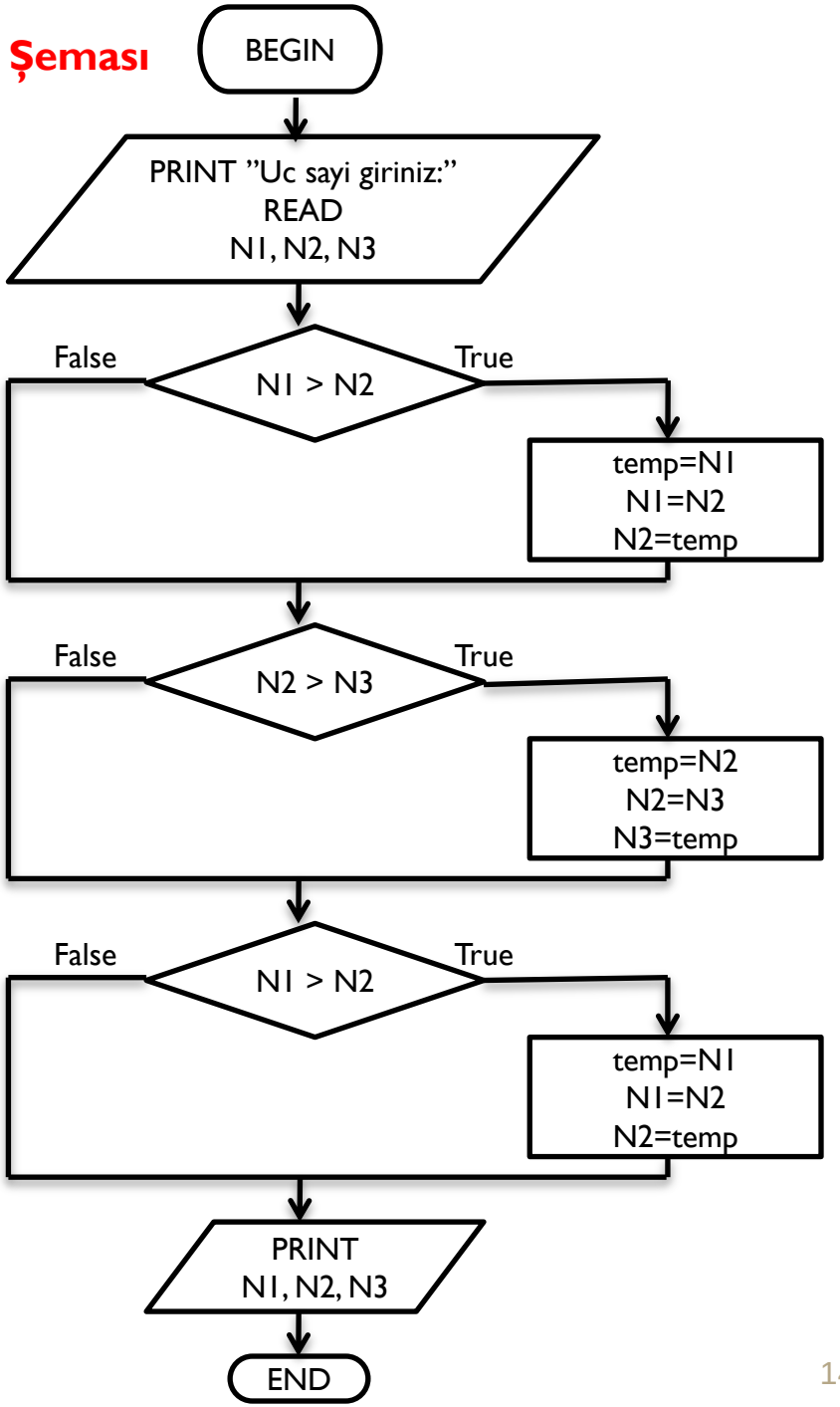
Örnek:

Üç sayıyı N1, N2, N3 olarak okuyup onları N1<=N2<=N3 şeklinde sıralayan algoritmayı yazıp akış şemasını çiziniz.

Algorithm

1. BEGIN
2. PRINT "Uc sayi giriniz:"
3. READ N1, N2, N3
4. IF N1>N2
   - temp=N1
   - N1=N2
   - N2=temp
4. IF N2>N3
   - temp=N2
   - N2=N3
   - N3=temp
5. IF N1>N2
   - temp=N1
   - N1=N2
   - N2=temp
6. PRINT N1, N2, N3
5. END

Akış Şeması



Örnek:

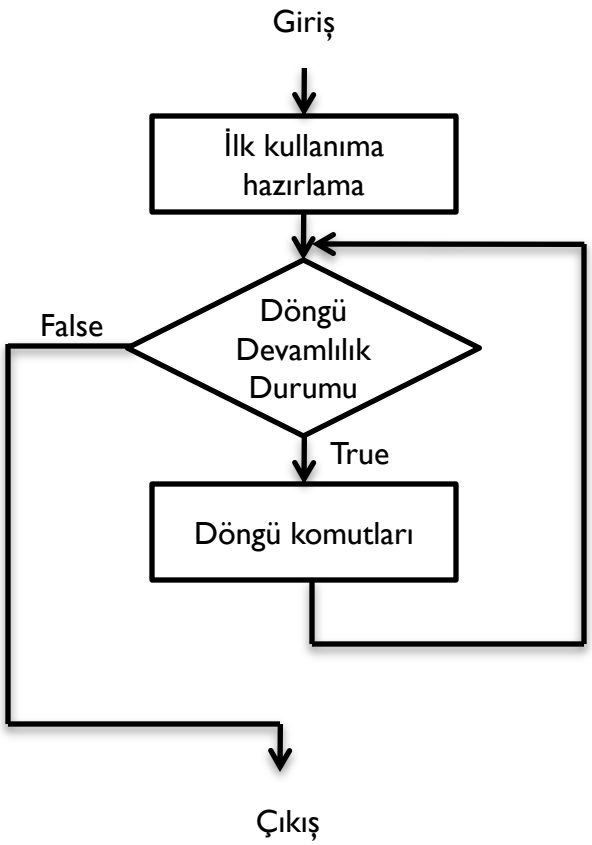
Verilen iki sayının maksimun olanını bulup ekrana yazdıran problemin algoritmasını yazıp akış şemasını çiziniz.

Algoritma:

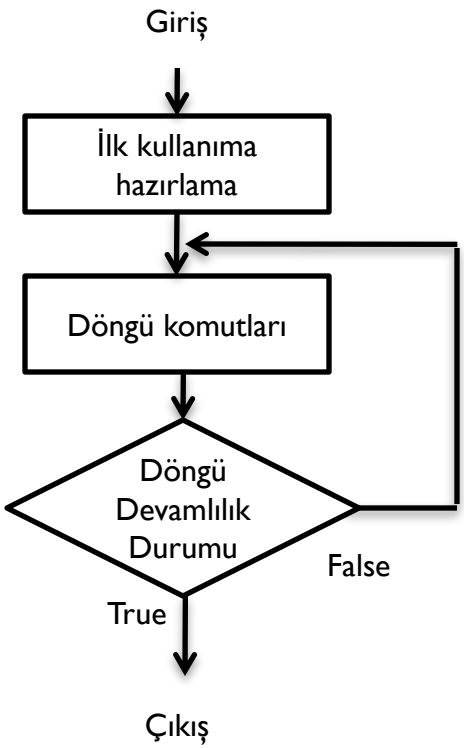
1. BEGIN
2. PRINT "Iki sayı giriniz:"
3. READ N1, N2
4. IF N1> N2
   PRINT "Maksimum:", N1
   ELSE
   PRINT "Maksimum:", N2
5. END

WHILE/WHILE-END DÖNGÜSÜ, REPEAT-UNTIL (DO-WHILE) DÖNGÜSÜ ve OTOMATİK-SAYAÇLI DÖNGÜ (FOR)

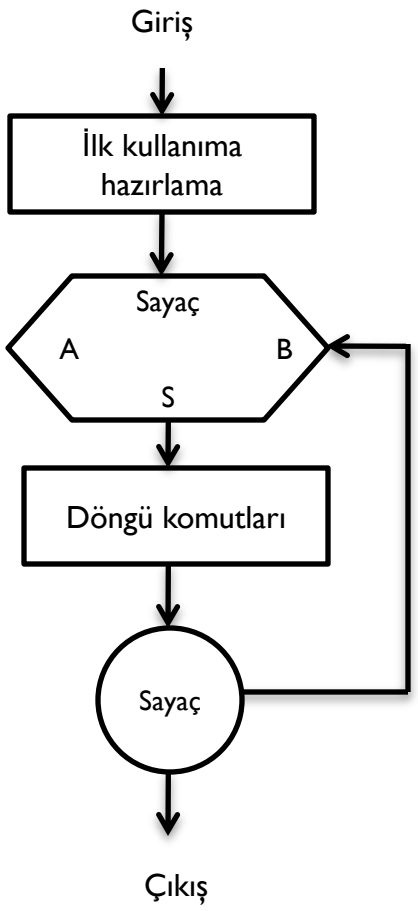
WHILE/WHILE-END



REPEAT-UNTIL (DO-WHILE)



OTOMATİK-SAYAÇLI DÖNGÜ (FOR)



Örnek:

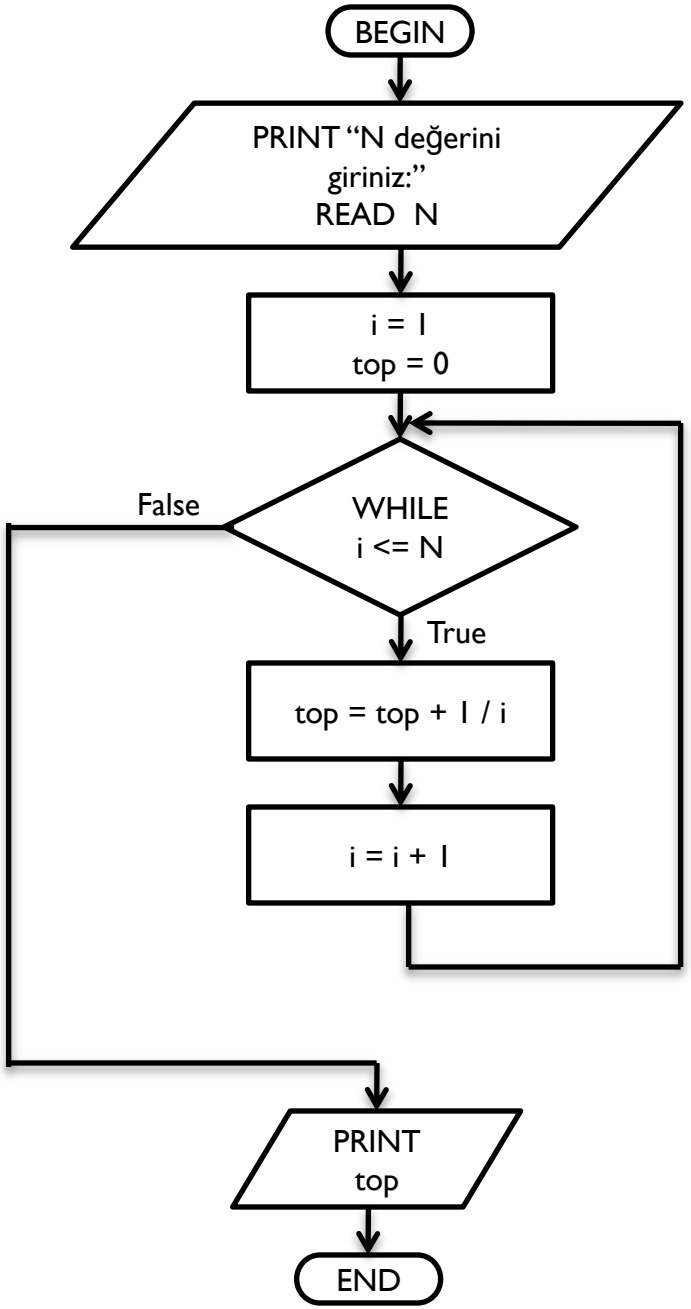
Aşağıdaki hesaplamayı yapan algoritmayı yazıp akış şemasını çiziniz.
Not: N kullanıcı tarafından girilecektir.

1/1 +1/2 + 1/3 + ... + 1/N

Algoritma:

1. BEGIN
2. PRINT "N değerini giriniz:"
3. READ N
4. i=1
5. top=0
6. WHILE i<=N
   top = top + 1 / i
   i = i + 1
   WHILE-END
6. PRINT top
7. END

Akış Şeması:



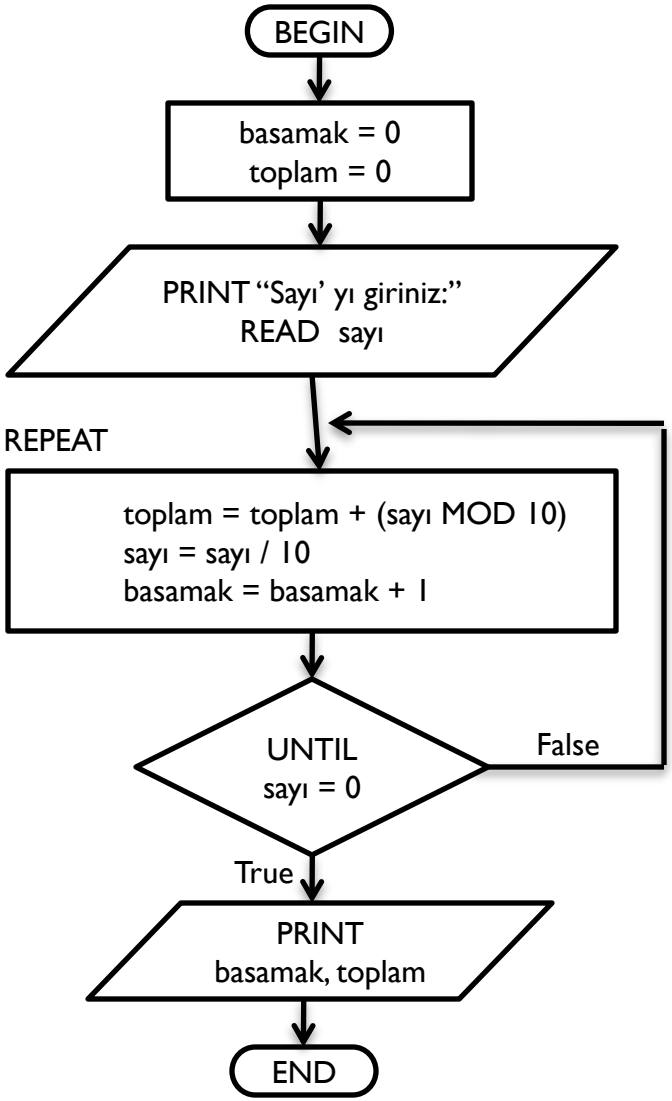
Örnek:

Kullanıcı tarafından girilen pozitif bir sayının basamaklarının toplamını ve basamak sayısını bulan algoritmayı yazıp akış şemasını çiziniz.

Algoritma:

1. BEGIN
2. basamak=0
3. toplam=0
4. PRINT "Sayı' yı giriniz:"
5. READ sayı
6. REPEAT
   toplam = toplam + (sayı MOD 10)
   sayı = sayı / 10
   basamak = basamak + 1
   UNTIL sayı = 0
6. PRINT basamak, toplam
7. END

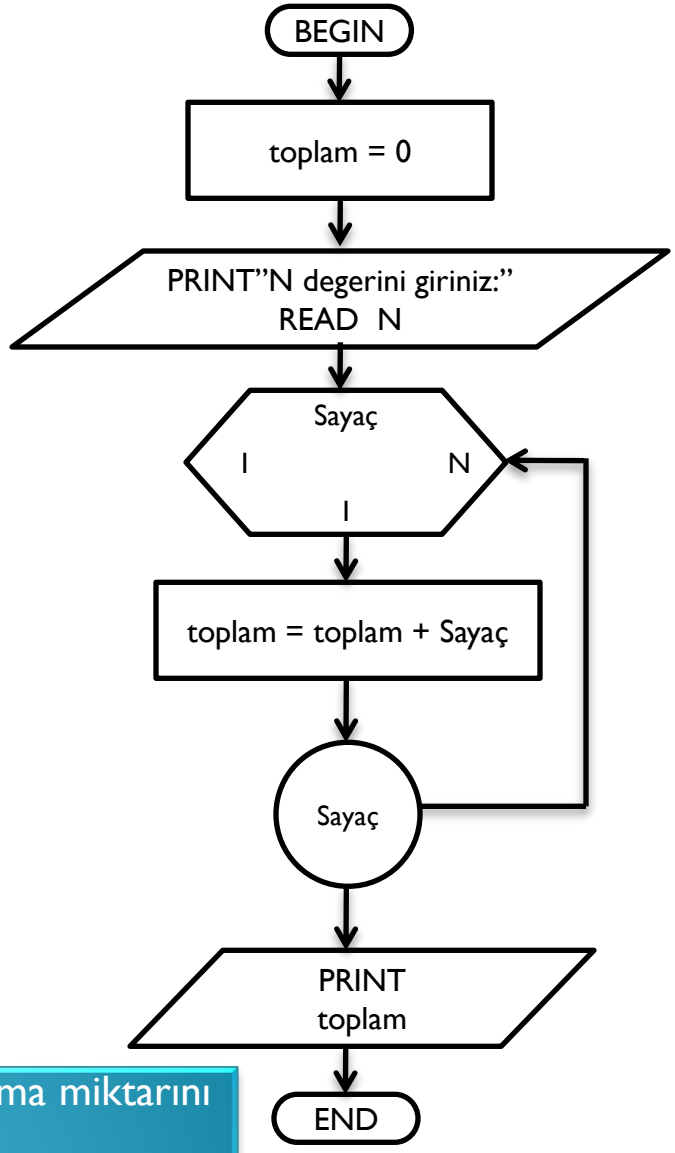
Akış Şeması:



Örnek:

1'den N'e kadar olan sayıların toplamını bulan algoritmayı yazıp akış şemasını çiziniz. N kullanıcı tarafından girilecektir.

1 + 2 + 3 + ... + N



Algoritma:

1. BEGIN
2. toplam=0
3. PRINT"N degerini giriniz:"
4. READ N
5. LOOP: Sayaç= 1 TO N STEP 1
   toplam = toplam + Sayaç
   LOOP-END: Sayaç
5. PRINT toplam
6. END

STEP 1, her dönüşte Sayaç'ın artırılma miktarını göstermektedir.

Not: Eğer her dönüşte 1 artırılacaksa STEP 1 yazılmayabilir. STEP 1 yazılmadığında sistem otomatik olarak her dönüşte 1 artırılacağını düşünerek hareket eder.