BİLGİSAYAR PROGRAMLAMA 1

Ders Notu 2– Algoritma Tasarımı ve Akış Diyagramları

Konya Teknik Üniversitesi
Elektrik – Elektronik Mühendisliği Bölümü

15.03.2024

Konya

Program Oluşum Süreci

ADIM-1) Problemin Belirlenmesi

ADIM-2) Problem Hakkında Veri Toplanması

ADIM-3) Verilerin Analiz Edilmesi

ADIM-4) Algoritma / Akış Diyagramı / Sözde Kod

ADIM-5) Herhangi bir Programlama Dilinde Kodlama

ADIM-6) Programın Test Edilmesi

ADIM-7) Lisanslama- Paketleme

Algoritma

Verilen bir problemin bilgisayar ortamında çözülmesi için yapılması gereken işlemlerin yapılış sırasının belirlenmesi ve adım adım ortaya koyulmasıdır.

Algoritma sözcüğü Ebu Abdullah Muhammed İbn Musa el Harezmi adındaki Türkistan'lı alimden kaynaklanır.

Bu alim 9. yüzyılda cebir alanındaki **algoritmik** çalışmalarını kitaba dökerek matematiğe çok büyük bir katkı sağlamıştır. "Hisab el-cebir ve el-mukabala kitabı dünyanın ilk cebir kitabı ve aynı zamanda ilk algoritma koleksiyonunu oluşturur.

Latince çevirisi Avrupa'da çok ilgi görür - alimin ismini telaffuz edemeyen Avrupalılar "algorizm" sözcüğünü "Arap sayıları kullanarak aritmetik problemler çözme kuralları" manasında kullanırlar. Bu sözcük daha sonra "algoritma"ya dönüşür ve genel kapsamda kullanılır.

https://tr.wikipedia.org/wiki/Algoritma

Algoritmada Olması Gereken Özellikler:

- **Giriş/Çıkış:** Algoritmanın üzerinde işlem yapması için veri girişi sağlanmalı ve gerçekleştirilen işlemler bir sonuç üretmelidir.

- **Etkinlik:** Gereksiz tanım ve tekrarlar içermemeli ve etkin sonuçlar üretebilmelidir. Gerektiğinde diğer algoritmalar içinde de kullanılabilmelidir.

Algoritmada Olması Gereken Özellikler:

- **Sonluluk:** Belirli sayıda adımdan oluşmalı, başlangıç ve bitiş noktası belli olmalıdır. Açıkta kalan bir durum, düşünülmeyen bir ihtimal kalmamalıdır.

- **Kesinlik:** İşlem sonuçları kesin olmalı ve tekrar yürütüldüğünde aynı sonucu sağlayabilmelidir.

- **Başarım:** Başarımı iyi olacak şekilde tasarlanmalı; olabildiğince bellek gereksinimi ile çalışma süresi dengeli tutulmalıdır.

Örnek:

1-Klavyeden girilen sayıyı oku.

2-Sayının karesini hesapla.

3-Sonucu ekrana yazdır.

Örnek:

1-Klavyeden girilen N sayısını oku.

2-T=0; X=1 tanımla.

3-T=T+X işlemini yap.

4-X değerini 1 arttır.

5-Eğer X<=N ise 3'e git.

6-T değerini yazdır.

Yandaki algoritma ne işe yarar?

*(Klavyeden girilen bir N değerine kadar olan tamsayıları toplayan ve sonucu ekrana yazan bir algoritma)

Sözde kod (Pseudo Code):

- Algoritmanın yarı programlama dili kuralları yarı konuşma dili şeklinde tanımlanmasıdır.
- Sözde kodlar, programlama mantığı altında, «eğer, iken» gibi koşul kelimeleri ve >, =, < gibi ifadeler ile beraber yazılır.
- Bir algoritma, sözde kod ile verilirse daha kolay anlatılabilir ve daha zahmetsiz bir şekilde programlama diline uyarlanabilir.

Sözde kod (Pseudo Code)

- ✓ **Algoritma Adı:** Algoritmayı tanımlayacak anlamlı bir isim verilmelidir.
- ✓ **Yaptığı İş:** Algoritmanın ne iş yapacağını ve algoritmada kullanılacak değişkenlerin tür ve amacını belirten açıklayıcı bilgiler eklenebilir.
- ✓ **İşlem Adımları:** Adımlar işlem sırasına göre numaralandırılmalıdır.
- ✓ **Açıklamalar:** Her bir adımda gerçekleştirilen işlemler kısa notlarla açıklanabilir.

Tanımlayıcılar

Programdaki değişkenleri, sabitleri, kayıt alanlarını, özel bilgi tiplerini vb. adlandırmak için kullanılan kelimelerdir.

- Yerini tuttukları ifadelere çağrışım yapacak şekilde seçilmelidir.
- İngiliz alfabesindeki **A-Z** veya **a-z** arasındaki 26 harf ile **0-9** arası rakamlar kullanılabilir.
- Sembollerden sadece altçizgi «_» kullanılabilir.
- Tanımlayıcı isimleri harfle veya altçizgiyle başlayabilir.
- Tanımlayıcı ismi, rakamla başlayamaz veya sadece rakamlardan oluşamaz.

Algoritma BuyuguBul;

/* Bu algoritma iki sayı okur ve büyük olanını bulup ekrana yazdırır. */

```
1- Oku (Sayı A);              /* Birinci sayıyı oku. */
2- Oku (Sayı B);              /* İkinci sayıyı oku. */
3- Karşılaştır (A>B?)         /* Sayıları karşılaştır. */
        Evet => EB=A;
        Hayır=> EB=B;
4- Yaz(EB);                   /* Büyük sayıyı yazdır. */
5- Dur;
```

İki sayının toplamı

Algoritma

1. BAŞLA
2. Birinci sayıyı gir
3. İkinci sayıyı gir
4. İki sayıyı topla
5. Sayıların toplam değerini yaz
6. BİTİR

Sözde Kod

Toplam için T, birinci sayı için X, ikinci sayı için Y seç

1. BAŞLA
2. X değerini OKU
3. Y değerini OKU
4. T = X + Y
5. T değerini YAZ
6. BİTİR

Üçgenin alanını hesaplayan algoritma

1. BAŞLA
2. Taban değerini gir
3. Yükseklik değerini gir
4. Taban ile yüksekliği çarp ve sonucu ikiye böl
5. Çıkan sonucu yaz
6. BİTİR

Sözde kodu nasıl yazarız?

Taban için t, yükseklik için y, alan için A seç

1. BAŞLA
2. t değerini OKU
3. y değerini OKU
4. A = (t * y)/2
5. A değerini YAZ
6. BİTİR

Akış Şeması/Diyagramı:

Yapılacak bir işe ya da yazılacak bir programa ait algoritmanın şekilsel/grafiksel olarak gösterilmesidir.

Bu amaçla dikdörtgen, elips gibi simgesel işaretler kullanılmaktadır.

Akış Şeması Simgeleri

Başla/Bitir

Programın nereden başlayacağını ve nerede sonlanacağını belirtir. Her algoritmada yalnız bir başlangıç noktası olabilir.

Giriş

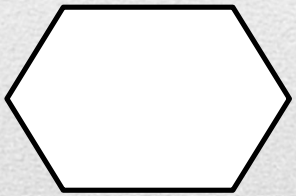
Klavye

Klavye, kart okuyucu gibi birimler üzerinden programa veri/bilgi aktarmak için kullanılır. Giriş simgesinin içine verilerin saklanacağı değişkenlerin isimleri yazılır. Böylece simgeye bakınca kaç adet veri girileceği de belli olur.

Akış Şeması Simgeleri

Çıkış/Çıktı

Ekrana yazdırmak, yazıcıdan çıktı almak gibi işlemlerde kullanılır. Çıkış simgesinin üstüne nelerin çıkış birimine gönderileceği yazılır.



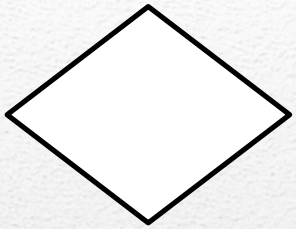
Döngü

Bir işin ya da bir grup işin tekrarlanması gerektiğinde kullanılır. Koşula bağlı döngüler yazmak da mümkündür. Döngü simgesi içine döngü sayacı, sayaç artırımı gibi değerler açıkça yazılır.

İşlem

Aritmetik işlemler, atama işlemleri ya da deyimler yazılabilir.

Akış Şeması Simgeleri

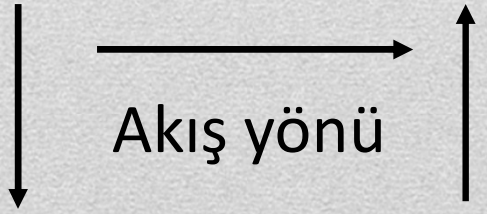


Karar/Karşılaştırma

Koşula bağlı olarak farklı işlemler yapılacağında kullanılır, koşulu sağlama durumuna göre iki farklı yöne dallandırılabilir.

Fonsiyon çağırma

Daha önce yazılmış algoritmaların, o anda kullanılan algoritma içine tamamiyle yazılmadan kullanılmasını sağlar. Simgenin içine çağırılacak fonksiyonun adı ve parametreleri yazılır.



Akış yönü

İşlem bittikten sonra program akışının nereye dallanacağını/yöneleceğini belirtir.

Bir bilgisayar programının geliştirilmesinde kullanılan programlama dili ne olursa olsun bu programların akış şemalarında genel olarak üç basit mantıksal yapı kullanılır:

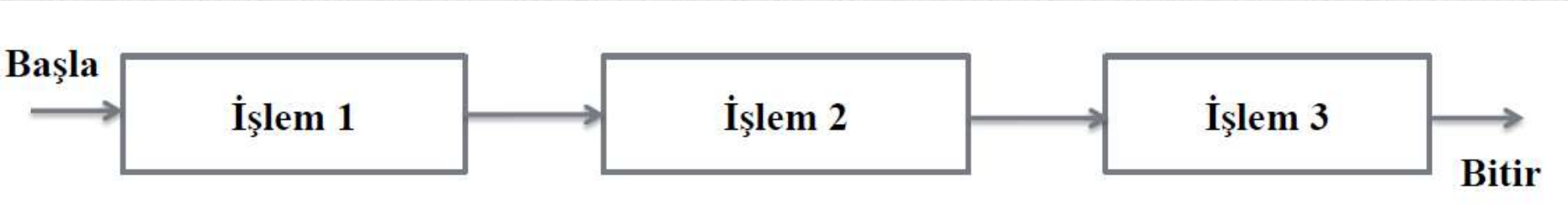
1. SıralıYapı

2. Karar Verme Yapısı

3. Tekrarlı Yapı

Sıralı Yapı

- Sıralı yapı, hazırlanacak programdaki her işlemin mantık sırasına göre nerede yer alması gerektiğini vurgular. Bu yapı sona erinceye kadar ikinci bir işlem başlayamaz.

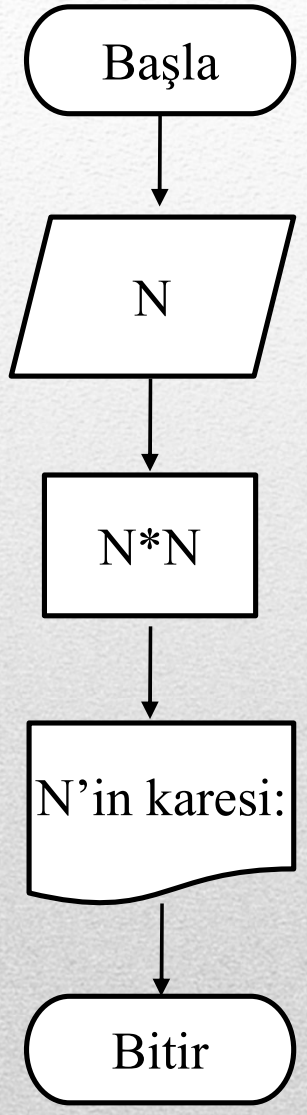


Örnek:

1-Klavyeden girilen sayıyı oku.

2-Sayının karesini hesapla.

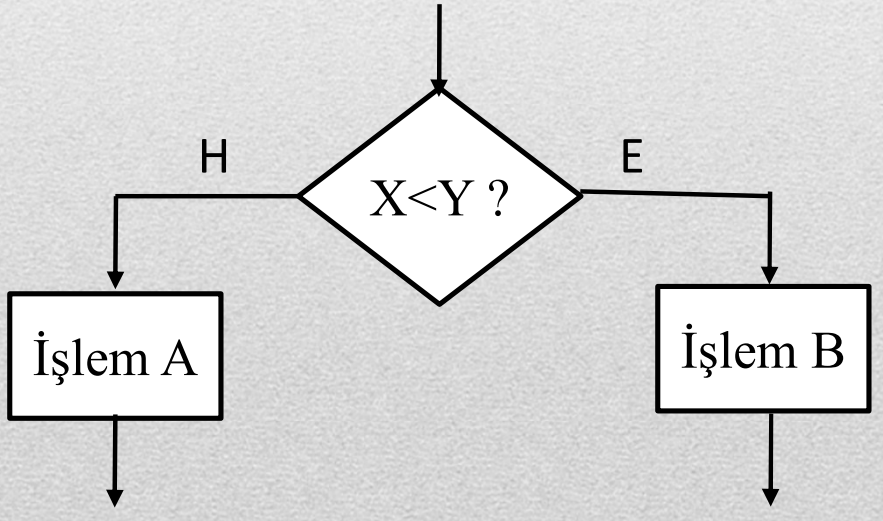
3-Sonucu ekrana yazdır.

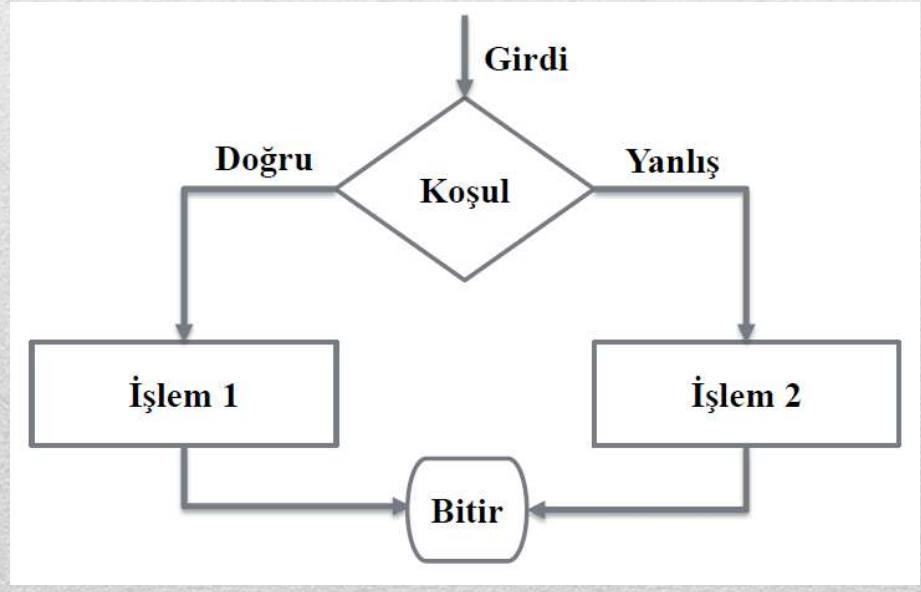


Karar Verme Yapısı / Koşullu İfadeler

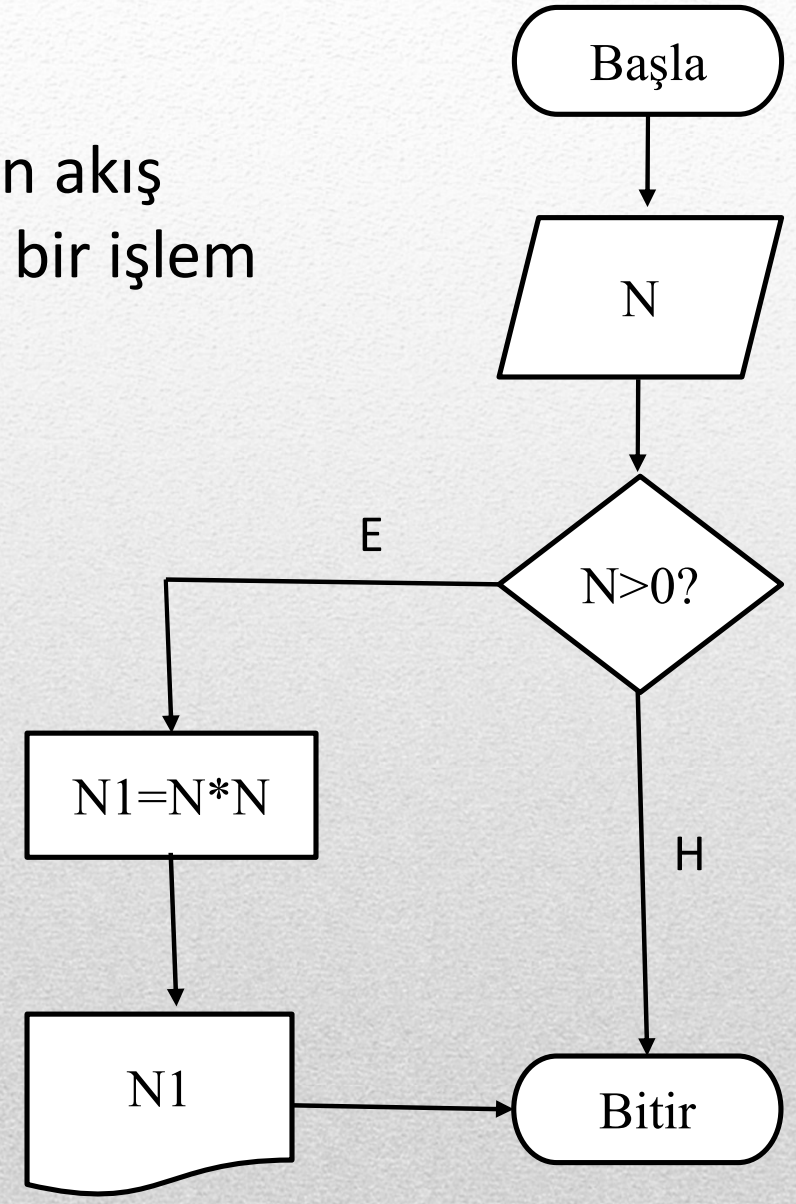
- Algoritmada verilen adımlar genellikle sırayla işlenir. Ancak bazı koşullara göre bu sıralamanın değiştirilmesi gerekebilir. Böyle durumlarda koşullu dallanma kullanılır.

- Baklava dilimi şeklindeki koşullu dallanma simgesinin içine koşul yazılır ve bu koşulun sonucuna göre iki farklı yöne dallanma söz konusu olur.





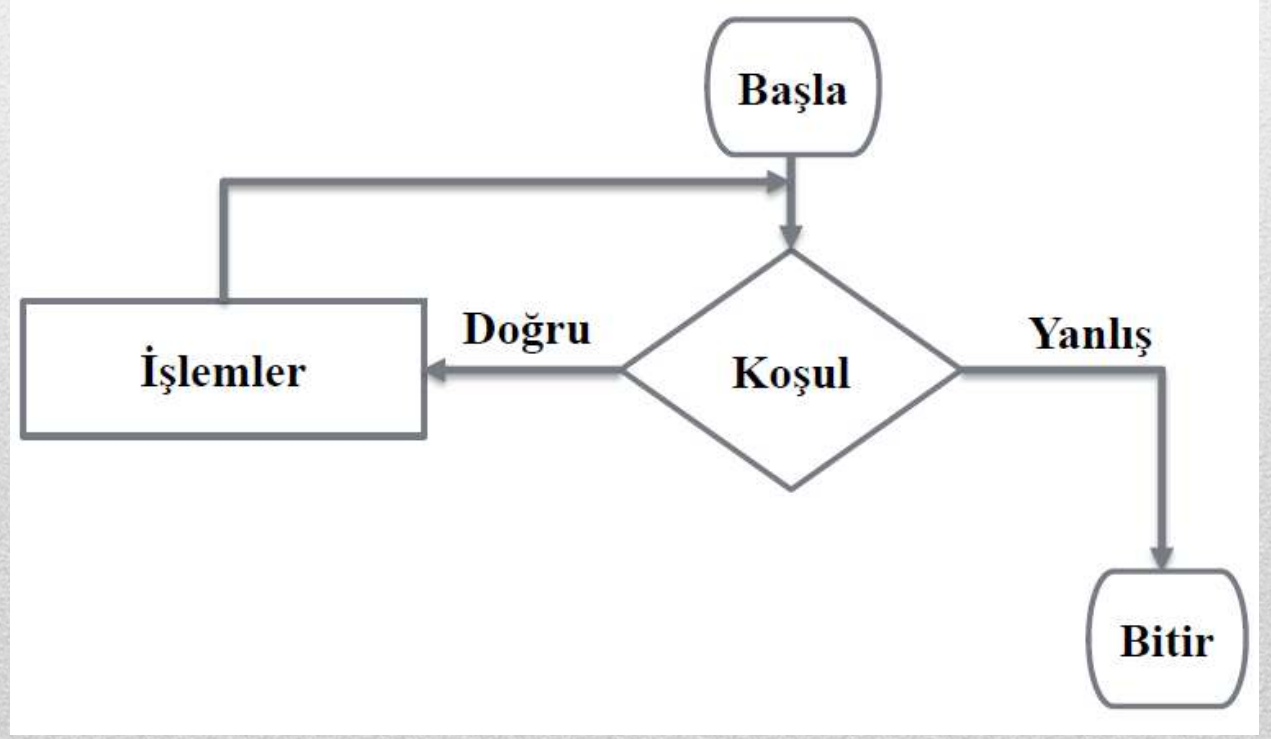
Yanda verilen akış şeması nasıl bir işlem sağlar?



* Klavyeden girilen pozitif sayıların karesini alıp ekrana yazdırır.

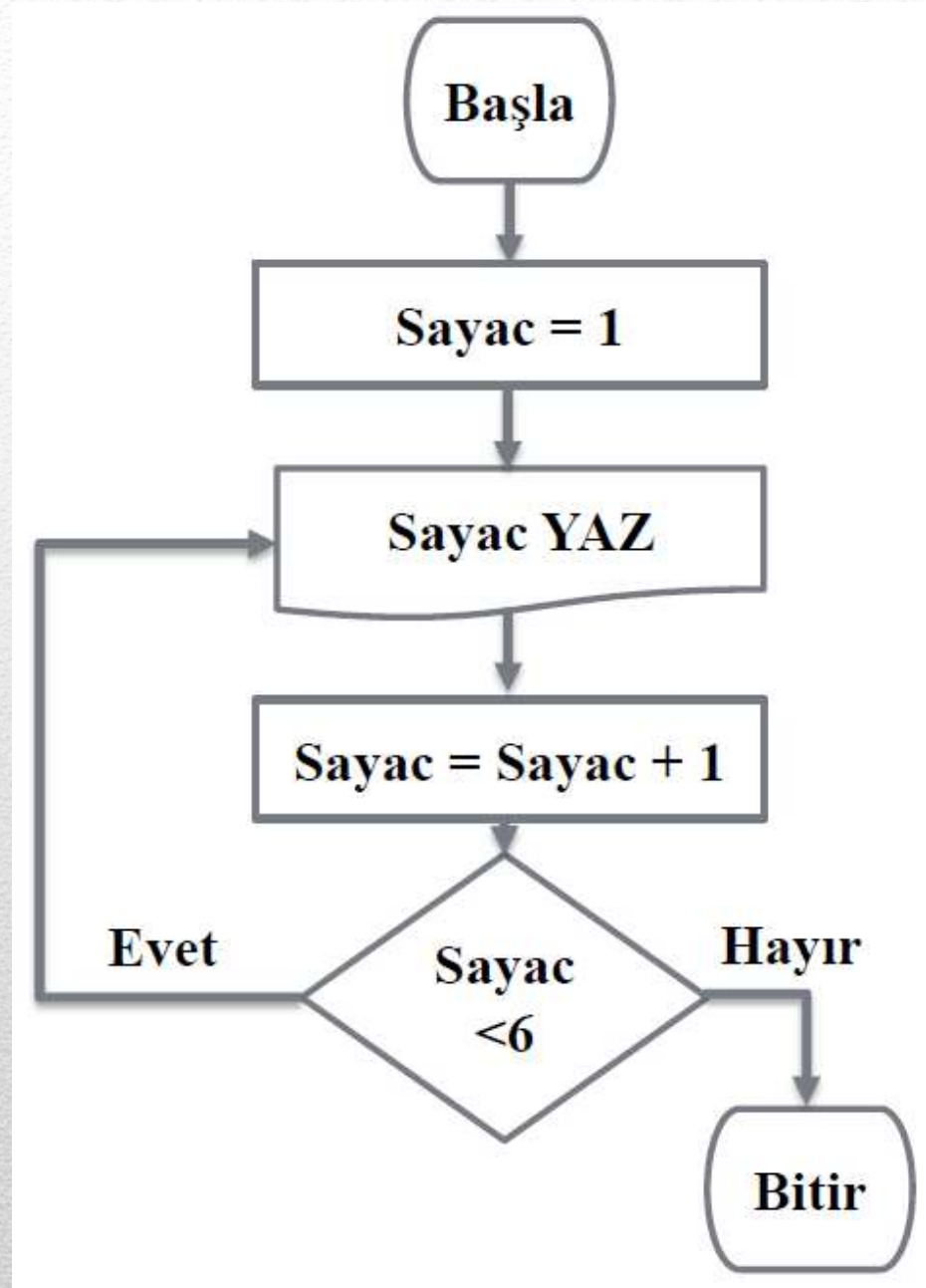
Tekrarlı Yapı / Döngü

- Algoritma içinde, bazı satırlar tekrarlı şekilde işlem görüyorsa, bir döngü söz konusudur. Döngülere belirli bir koşul geçerli olduğu sürece devam eden eylemleri tanımlamak için başvurulur.



ÖRNEK: 1’ den 5’ e kadar sayıların ekrana yazdırılması

1. BAŞLA
2. Sayac = 1
3. Sayac değerini YAZ
4. Sayac = Sayac + 1
5. Eğer Sayac < 6, GİT 3
6. BİTİR



1. BAŞLA
2. Sayac = 1
3. Toplam = 0
4. Eğer Sayac > 10, GİT 8
5. Toplam = Toplam + Sayac
6. Sayac = Sayac + 2
7. GİT 4
8. BİTİR

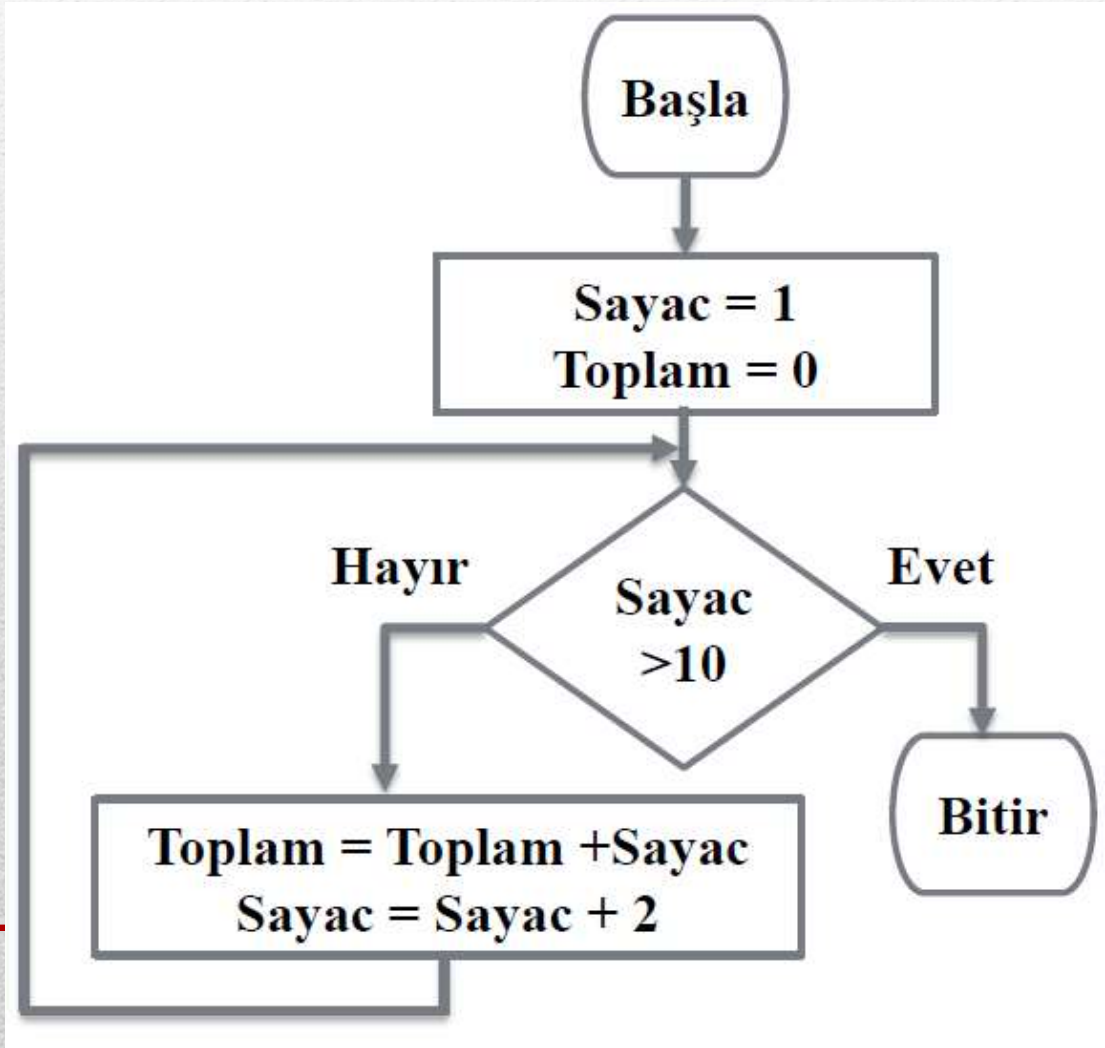
ÖRNEK: 1-10 arasındaki tek sayıları toplayan algoritma

Değişken İzleme Tablosu

Eski Sayac	Eski Toplam	Yeni Toplam	Yeni Sayac
1	0	1	3
3	1	4	5
5	4	9	7
7	9	16	9
9	16	25	11

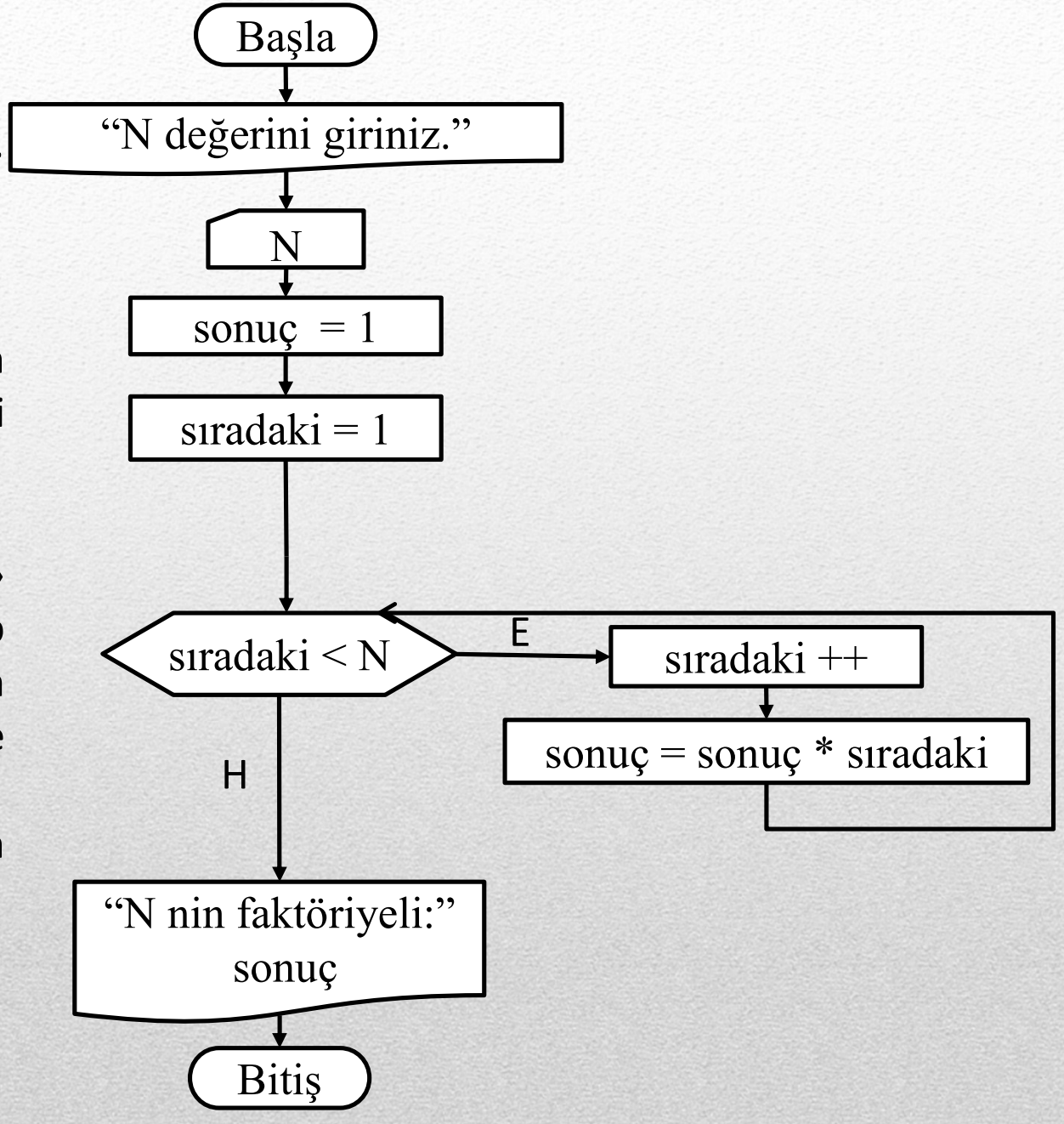
1. BAŞLA
2. Sayac = 1
3. Toplam = 0
4. Eğer Sayac > 10, GİT 8
5. Toplam = Toplam + Sayac
6. Sayac = Sayac + 2
7. GİT 4
8. BİTİR

ÖRNEK: 1-10 arasındaki tek sayıları toplayan algoritma



Klavyeden girilen bir sayının faktöriyelini hesaplama (N! = 1 * 2 * 3 *…* N)

1. Başla.
2. Kullanıcıdan bir sayı girmesini iste.
3. Girilen sayıyı oku.
4. «sonuç» değişkenine 1 ata.
5. «sıradaki» değişkenine 1 ata.
6. «sıradaki» değişkeni N sayısından küçük olduğu sürece aşağıdaki adımları tekrar et:
   1. «sıradaki» değişkenini 1 artır.
   2. «sıradaki» ve «sonuç» değişkenlerini birbiri ile çarp ve bu çarpmadan elde edilen sonucu «sonuç» değişkenine ata.
7. “sonuç” değişkeninin aldığı en son değeri ekrana yazdır.
8. Son.



$$ax^2 + bx + c = 0$$

$$\Delta = b^2 - 4ac$$

$\boldsymbol{\Delta > 0}$

$$x_1 = \frac{-b - \sqrt{\Delta}}{2a}$$

$$x_2 = \frac{-b + \sqrt{\Delta}}{2a}$$

$\boldsymbol{\Delta = 0}$

$$x_1 = x_2 = \frac{-b}{2a}$$

$\boldsymbol{\Delta < 0}$ → Reel kök yoktur.

İkinci dereceden denklemlerin köklerini bulan algoritma (sözde kod)

1. Başla
2. a,b,c katsayılarını gir.
3. $\Delta = b^2 - 4ac$
4. $\Delta > 0\ ise$

$$x_1 = (-b - \sqrt{\Delta})/2a$$
$$x_2 = (-b + \sqrt{\Delta})/2a$$

7'ye git.

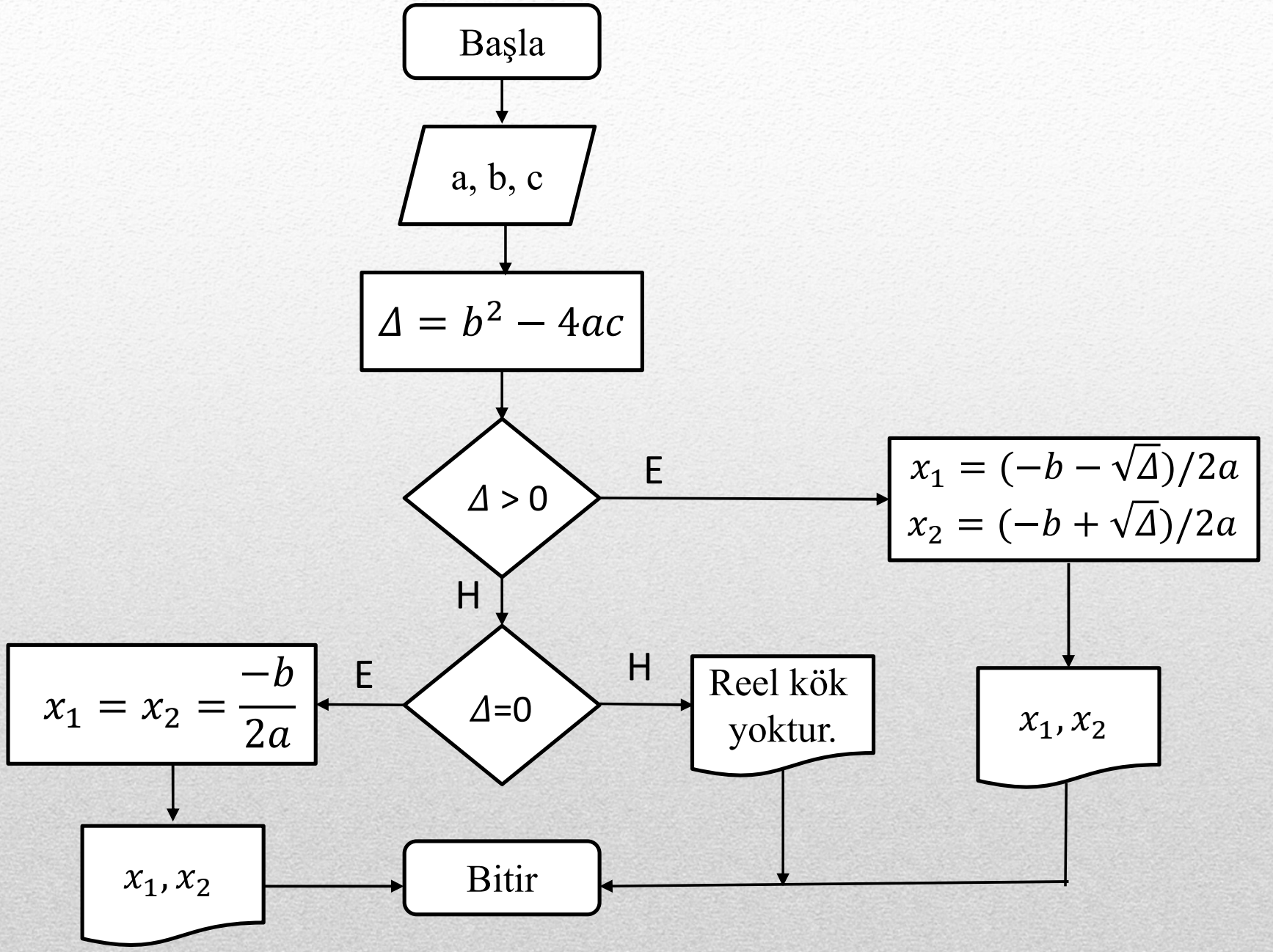
5. $\Delta = 0\ ise$

$$x_1 = x_2 = -b/2a$$

7'ye git.

6. «Reel kök yoktur» yaz , 8'e git.
7. x_1 , x_2' yi yaz.
8. Bitir

İkinci dereceden denklemlerin köklerini bulan akış şeması



ÖDEV:

30 kişilik bir sınıftaki öğrencilerin Bilgisayar Programlama dersinden aldıkları vize notlarının tek tek ekrandan girilmesini ve bu notların Konya Teknik Üniversitesi lisans yönergesine göre harf karşılığının ekrana yazılmasını sağlayan programa ait algoritmayı yazınız ve akış şemasını oluşturunuz.