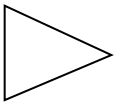
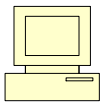
PROGRAMLAMA TEMELLERİ

Bilgisayar programlamasının amacı insan ile makine arasında ilişki kurmak ve haberleşmeyi sağlamaktır.

Yüksek Seviyeli Diller
(High Level Language)
- Pascal, Cobol,Basic,C++ -



Alçak Seviyeli Diller
(Assembly Languages)
- Assembly, Derleyiciler -



0101011

insan ***derleyici*** ***Makine Dili***

Günümüzde yüzlerce programlama dili vardır. Bunları genel olarak üç gruba ayırabiliriz.

1. ***Makine dili (Machine Languages);*** mikroişlemci komutlarının kodunu çözer, ikili sayılara dönüştürür. Yani makine dili 1 ve 0 ların dizilimlerinden oluşur.
2. ***Alçak Seviyeli diller (Assembly Languages) ; S***embolik makine dillerini ve doğal makine dillerini kapsar.

   MOV AX, S1
   MOV BX, S2
   ADD AX,BX
   MOV TOPLAM, AX

3. ***Yüksek Seviyeli Diller ;*** belirli kalıplarda belirli kurallara göre yazılmaları şartlandırılmıştır. Bu şartlandırmalar her programlama diline göre değişir.
   Yüksek Seviyeli bir programlama dilinin şu üç işlevi yerine getirmesi gerekir.
   a). Bilgisayar kullanımında uzmanlaşmamış personel için , insan diline olabildiğince yakın bir yapay dile gereksinim vardır.
   b). Belirli bir kullanım alanına özgü bir bilgi türünü , olabildiğince zahmetsiz bir şekilde işleyebilecek bir yapıya sahip olmaları gerekir.
   c). Programı yerine getirmek için kullanılan bilgisayardan bağımsız olmalıdır.

Yüksek Seviyeli bir dili Alçak Seviyeli veya Makine diline çeviren programlar, derleyici veya yorumlayıcılardır.

.

Derleyiciler (Compilers) : İşleme yönelik dilde yazılmış bir temel programı (kaynak program), makine diliyle yazılmış amaç programa çeviren programlardır. Derleyiciye çevirmesi için verilen programa " **kaynak (source) program**", makine diline çevrilen programa ise "**amaç (object) program**" denir.

Derleyiciler, yüksek seviyeli her bir programlama diline özgü yazılırlar. Yüksek Seviyeli bir dille yazılan yazılımlarda, kurallarına ters düşen hata kontrollerini yaptıktan sonra komutların ve bilgilerin makine tarafından tanınarak işlemlerin yapılmasını , sonuçların tekrar yüksek seviyeli programlama diline anlaşılır duruma getirilmesini sağlarlar.

Kaynak kodu $\xrightarrow[\text{Çevriciler}]{}$ Amaç (object) kodu

Örneğin; DOS İşletim sisteminde bir PASCAL programının hangi aşamalardan geçerek çalışabilir bir kod haline geldiği aşağıdaki şemada gösterilmiştir.

Kaynak programlar, editörler kullanılarak yazılırlar. Daha sonra bu kaynak kodlar derleyiciler tarafından amaç koda çevrilirler. En sonunda ise bağlayıcı programlar ile *"çalışabilir"* hale getirilirler.

Yorumlayıcılar (Interpreters) : Yorumlayıcılar, kaynak programı satır satır ele alarak doğrudan çalıştırırlar, yorumlayıcılar standart bir amaç kod üretmezler. Yorumlama işlemi satır satır yapıldığında genellikle ilk hatanın bulunduğu yerde programın çalışması da kesilir.

- Yorumlayıcılar, derleyicilere göre çok daha kolay yazılabilir.
- Yorumlayıcıların bir programı icra zamanı (run time) derleyicilere göre oldukça uzundur.
- Yorumlayıcıların her zaman kaynak koda ihtiyaç duymaları da kaynak kodun gizliliği ve güvenirliği açısından bir dezavantajdır.
- Basic programlama dili yorumlayıcı kullanırken , Pascal ,Cobol, Fortran C++ gibi diller derleyici kullanır.

Programlama dilleri :

Hangi programlama dilini öğrenirsek öğrenelim 4 tane kalıp her programlama dilinde geçerlidir. Bunlar ;

—**Değişkenler** (İstediğimiz zaman başka bir değerle değiştirebileceğimiz bir bilgi taşıdır.)
—**Sabitler** (Doğrudan işleme sokulurlar ve program çalıştığı sürece değişmezler.)
—**Mesajlar** (Program ile kullanıcı arası iletişimi sağlar. Mesajlar aslında birer sabittir. Çünkü program çalıştığı sırada değişmezler.)
—**Komutlar** (Bilgisayarın o an ne yapacağını belirten bir emirdir. Kısaltılmış yada tam İngilizce kelimelerden oluşurlar.)

Değişkenler : Değişken, bilgileri saklamak üzere kullandığımız ismini ve tipini belirlediğimiz bellek alanlarına verilen isimdir.

—**Basic**

Değişken tipi tamsayı ise A%
Değişken tipi tek duyarlıklı gerçel sayı A! , A (7 basamaklı)
Değişken tipi çift duyarlıklı gerçel sayı A# (17 basamaklı)
Değişken tipi string (alfa sayısal) A$

Örnek :

```
CLS
  PRINT "22 / 7 Sayısının gösterimi"
  A%= 22 / 7
  PRINT "Tamsayı değeri :" , A%            →  3
  A!= 22 / 7
  PRINT "Tek duyarlıklı gerçel sayı değeri :" , A!      →  3,142857
  A# = 22 / 7
  PRINT "Çift duyarlıklı gerçelsayı değeri :" , A#      →  3,142857074737545
  A$ = 22 / 7
  PRINT A$          →  22 / 7
END.
```

—Pascal

1- Sayısal değişkenler :

Tamsayılar :

- Byte → 0 – 355 →1 Byte
- Integer → -32768 ... +32767 →2 Byte
- Word → 0 – 65535 →2 Byte
- Long int → - 2 milyar... +2 milyar →4 Byte
- Short int → - 128 ... +128 →1 Byte

Gerçel sayılar :

- Real →6 Byte
- Single →4 Byte

:**2.String (alfasayısal –alfanümerik) değişkenler**

char → 1 karakter

string → string [n]

Sabitler :

Belleğin bir bölümündeki bilgiyi ifade ederler. Sabitlerin bir adı vardır. Doğrudan işleme sokulurlar ve programın çalışması esnasında değişmezler.

Örnek :

```
Cls
  İnput "yarıçapı gir",r
  Pi = 3.14
  Print "dairenin alanı"
  Alan = pi * (r^2)
  Print "dairenin alanı" ,alan
End
```

Not: Var A : İnteger
Const PI = 3.14

Mesajlar : Aslında birer sabittir. Çünkü program çalıştığı sürece değişmezler.
Kullanıcı ile program arasındaki iletişimi sağlarlar.

Algoritma :

Bir problem sınıfını sonlu bir zamanda çözebilmek için geliştirilmiş açık seçik, yürütülebilir, sıralı, basit ve gerektikçe tekrar eden adımlardan oluşan yöntemdir.

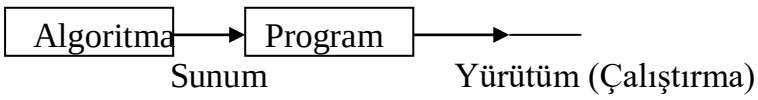
Algoritma İle İlgili Temel Kavramlar :

1. Sunum(presentation)
2. Program
3. Yürütüm(execution)

1.**Sunum:** Algoritmanın ifade şeklidir.

2.**Program:**Bir algoritmanın bilgisayar için geliştirilmiş biçimsel (format) sunum şeklidir.Bu amaç için çeşitli programlama dilleri kullanılır.

Basıc	Pascal	C
Print "Gir"	Writeln "Gir	Print "Gir"



3.**Yürütüm :**Bir programın yürütümü demek o program tarafından sunulan algoritmayı gerçeklemek demektir.

Akış Diyagramı : Algoritmanın şekillerle gösterilmesine denir.

Program Yazarken Takip Edilecek Hususlar :

1. Programın algoritmasını oluşturup (akış diyagramını) çiziniz.
2. İstenilen algoritma ve akış diyagramını bir programlama dilinde kodlayınız.
3. Programın bilgisayarda çalıştırılması.

Test
4. Derleyici hata veriyorsa düzeltiniz.
5. Programda istediğiniz sonuçları alamıyorsanız 1. adımı kontrol ediniz.

Programlama Hataları :

1. Yazım hatası
2. Mantık hatası

Algoritmada Kullanılan Operatörler :

Operatörler, nesneler veya sabitler üzerinde önceden tanımlanmış birtakım işlemleri yapan atomlardır.

Operatörler CPU (işlemci) tarafından bir faaliyete neden olurlar ve bu faaliyet sonunda bir değer üretilmesini sağlarlar.

Programlama dillerindeki her operatör bir yada birden fazla makine komutuna karşılık gelir.

Örn: c=(a*b) + 2 işleminde 4 operatör vardır.

1. Matematiksel işlem operatörleri :

^	Üst alma
*	Çarpma
/	Bölme
+	Toplama
-	Çıkarma

2. Mantıksal işlem operatörleri :

VE →	AND→ •
VEYA →	OR → +
DEĞİL →	NOT → |
DOĞRU →	TRUE
YANLIŞ →	FALSE

3. Karşılaştırma operatörleri :

=	Eşittir
<>	Eşit değildir
<	Küçüktür
>	Büyüktür
<=	Küçük eşit
>=	Büyük eşit

4. Genel işlem operatörleri :

=	Aktarma
( )	Parantez

Örnek : Bir işyerinde çalışan işçiler arasından yalnızca yaşı 23' ün üzerinde olup asgari ücret alanların isimlerini listeleyen komut satırını yazınız ?

— Eğer (yaş > 23)$_{1.koşul}$ ve (maaş = asgari ücret)$_{2.koşul}$ ise isim "listele"

— Eğer yaş > 23 veya maaş = asgari ücret ise isim "listele"

Örnek : Bir sınıfta Programlama Temelleri dersine 65' in üzerinde not alıp Türk Dili veya Yabancı Dil derslerinin herhangi birinden 65' in üzerinde not alanların isimlerini listeleyiniz ?

—Eğer Prog.Tem. > 65 ve (Türk dili > 65 veya Yabancı dil >65) ise isimleri listele

İki Sayının Toplamının Yazılması :

İNPUT Say 1 , Say 2
Toplam = Sayı 1 + Sayı 2
PRİNT TOPLAM

Algoritmalarda Kullanılan Özel Terimler

--Değişken : Programın çalışması esnasında farklı değerler alabilen bilgi alanlarıdır.
--Aktarma : Bir değişkeni başka bir değişkene atama (aktarma) olayıdır.
--Sayaç : Bir değişkenin değerini belirli aralıklarla fazlalaştırma veya azaltma olayıdır.

Değişkenlere isim verme kuralı :
1- Harfle başlamalı ve Türkçe karakter kullanılmamalı.
2- 0' dan 9' a kadar rakam kullanılabilir.
3- Özel karakterler ve boşluk karakteri kullanılmaz.
4- Değişken isimleri anlamlı olmalıdır.

Aktarma :

A = 5 ← 5'i A' ya aktar.

Örnek : Bilgisayara dışardan üç sayı girecek bu üç sayının toplamını veren algoritmayı oluşturun.

1- Başla
2- T = 0
3- Bir sayı gir (A)
4- T = T + A (T = 0+A → T=A)
5- 2. sayıyı gir (B)
6- T = T + B
7- 3. sayıyı gir. (C)
8- T = T + C → T = C
9- T' yi göster.

Sayaç : Sayaç değişkeni = Sayaç değişkeni + Artım değeri
Sayaç oluştururken ;
1- İlk önce sayaç değişkenine başlangıç değeri verilir.
2- Sayaç artan veya azalan bir şekilde tanımlanır.

Örnek : 1'den 10'a kadar sayıların toplamının algoritmasını oluşturunuz ?

1- Başla
2- T = 0 , S = 1
3- T = T + S
4- S = S + 1
5- Eğer S <= 10 ise 3. adıma git.
6- T' yi göster.
7- Dur.

T	S
0	1
1	2
3	3
6	4
10	5
15	6
21	7
28	8
36	9
45	10
55	11

Örnek : 1' den n' ye kadar sayıların toplamını yapan programın algoritmasını oluşturunuz ?

1. Başla
2. T = 0 , S = 1
3. N sayısını giriniz (5)
4. T = T + S
5. S = S + 1
6. Eğer S <= N ise 4.adıma git.
7. T' yi göster.
8. Dur.

T	S
0	1
1	2
3	3
6	4
10	5
15	6

Örnek : Üç kişi arasından boyu en uzun olanı bulan işlemin algoritmasını yazınız ?

I. YOL :

1. Başla
2. 3 kişinin boyunu giriniz (B1 , B2 ,B3)
3. En uzun = B1
4. Eğer B2 > En uzun ise En uzun = B2 ' dir.
5. Eğer B2 > En uzun ise En uzun = B3 ' dür.
6. En uzun olan en uzundur.
7. Dur.

B1	B2	B3	En Uzun
60	70	80	80

II. YOL :

1. Başla
2. 3 kişinin boyunu giriniz (B1 , B2 , B3)
3. Eğer B1 > B2 ve B1 > B3 ise en uzun B1
4. Eğer B2 > B1 ve B2 > B3 ise en uzun B2
5. Eğer B3 > B1 ve B3 > B2 ise en uzun B3
6. En uzunu göster.
7. Dur.

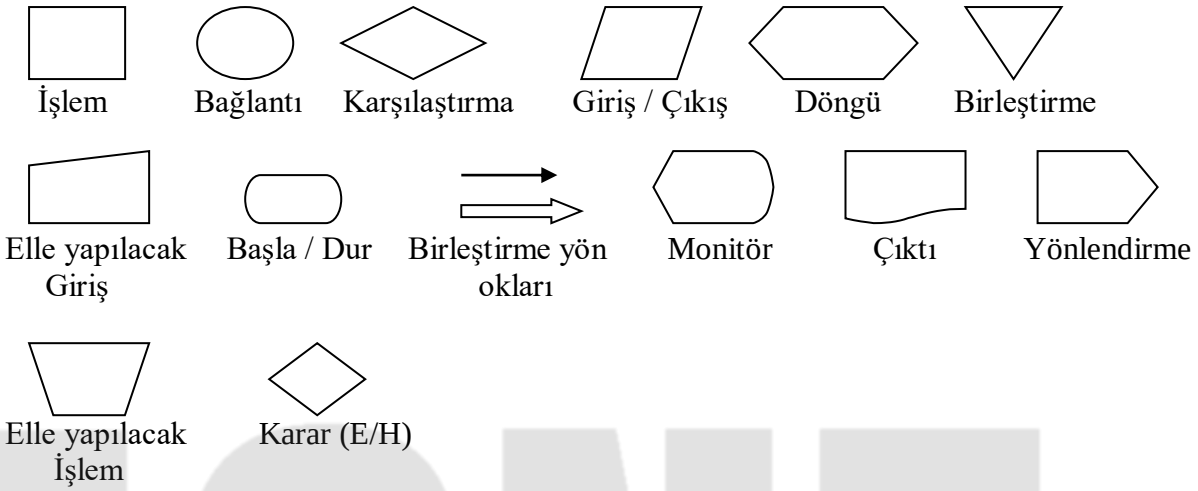
Algoritma Hazırlarken Dikkat Edilecek Hususlar :

1. Problemi iyice inceleyiniz.
2. Programlama mantığına en uygun çözümü seçiniz.
3. Algoritmada her işlem adımına bir satır numarası veriniz.
4. Adımlar arası geçişte kullanılan (dallanma komutu) "git" gideceği satır numarası ile birlikte kullanılır.
5. Mümkün olduğu kadar az dallanma komutu kullanınız.

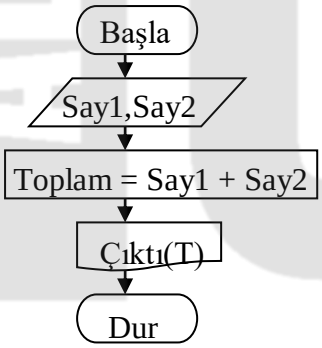
Bir problemi çözerken :

1. Önce değişken isimlerini belirleyiniz.
2. Veri girişlerini yapınız.
3. Yapılacak işlemle ilgili formülleri yazınız.
4. Sonuçları ekrana yazınız.

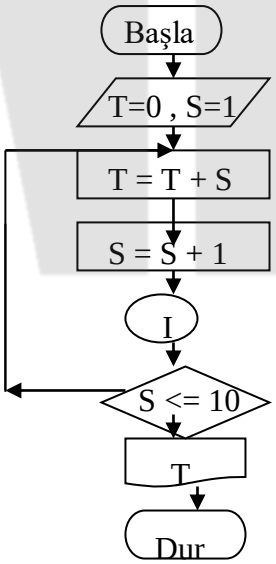
Akış Diyagramları :



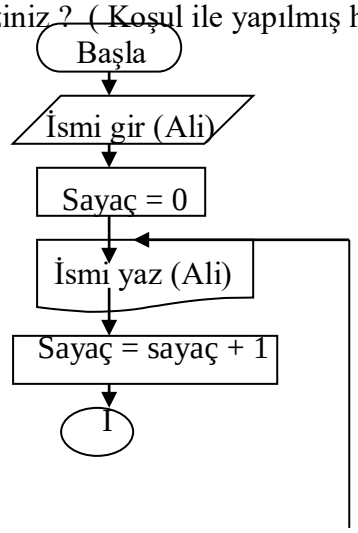
Örnek : İki sayının toplamı



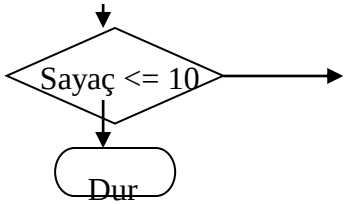
Örnek : 1'den 10' a kadar olan sayıların toplamı



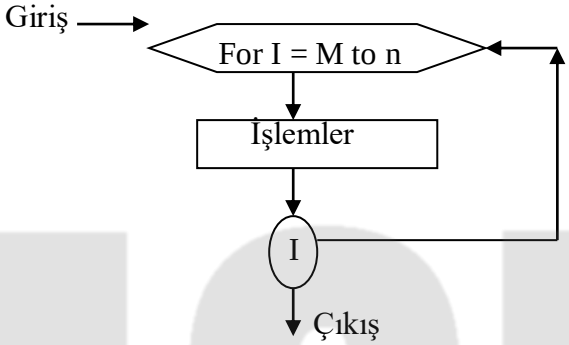
Örnek : Girilen bir ismi 10 kez tekrarlayan programın algoritmasını oluşturunuz, akış diyagramını çiziniz ? (Koşul ile yapılmış hali)



1 Başla.
2 İsmi Gir (Ali)
3 Sayaç = 0 (Sayacın ilk değeri)
4 İsmi yaz (Ali)
5 Sayacı 1 artır. (Say = Say + 1)
6 Eğer say. 10' a küçük eşitse (sayaç <= 10) ise 4. adıma git
7 Dur.



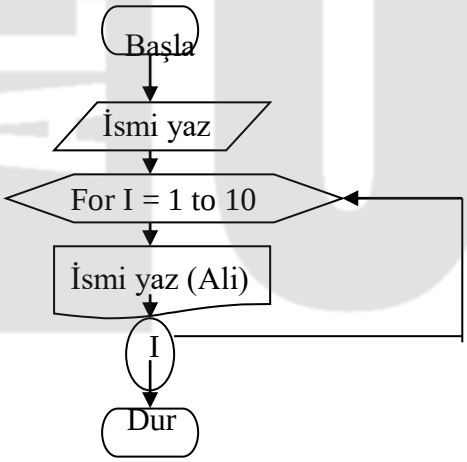
— Döngülerin Akış Diyagramı :



I Döngü Değişkeni
M Döngü Değişkeninin İlk Değeri
N Döngü Değişkeninin Son Değeri

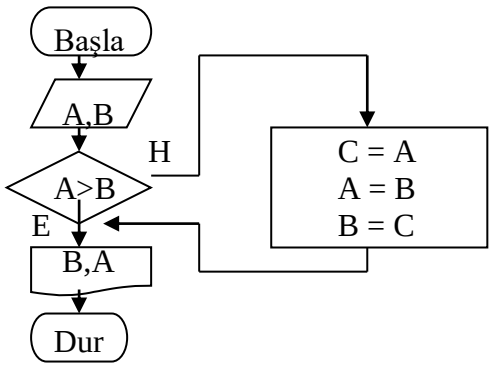
—Döngü değişkeni döngü dışına en son aldığı değeri bir artırarak çıkar.

Örnek : Girilen bir ismi 10 kez tekrarlayan programın algoritmasını oluşturunuz, akış diyagramını çiziniz ? (Döngü ile yapılmış hali)



1 Başla
2 İsmi gir (Ali)
3 Döngü oluştur (1'den 10' a kadar)
4 İsmi yaz (Ali)
5 Döngüyü oluştur.
6 Dur

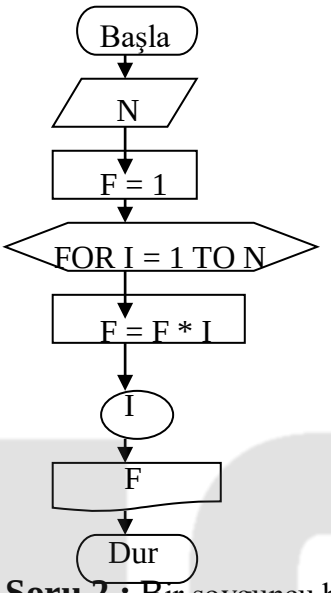
Örnek : Rasgele girilen iki sayıdan küçük olanı önce yazan programın algoritmasını oluşturup akış diyagramını çiziniz ?



1 Başla.
2 2. sayıyı giriniz (A , B).
3 Eğer A>B ise 5. adıma git.
4 A ile B' yi giriniz.
5 B' yi önce, A' yi sonra yaz.
6 Dur.

Soru 1 : Bilgisayara girilen 10 sayıyı küçükten büyüğe doğru sıralayan işlemin algoritmasını yapınız ?

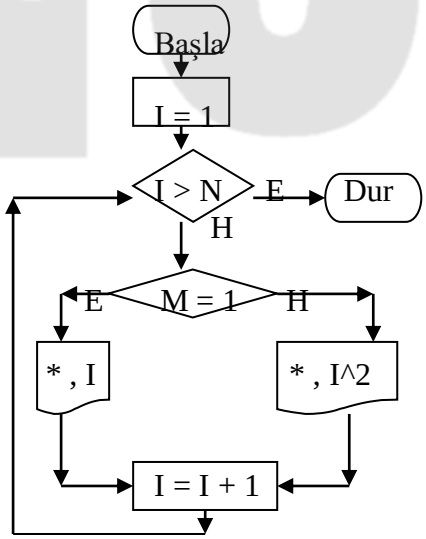
Örnek : Dışardan girilen n sayısının faktöriyelini alan programın algoritmasını oluşturunuz, akış şemasını çiziniz ?



1 Başla
2 Değişkeni gir (N)
3 F = 1
4 Döngü oluştur.(1' den N' ye)
5 F = F * I
6 Döngüyü sonlandır.
7 Sonucu göster. (F)
8 Dur.

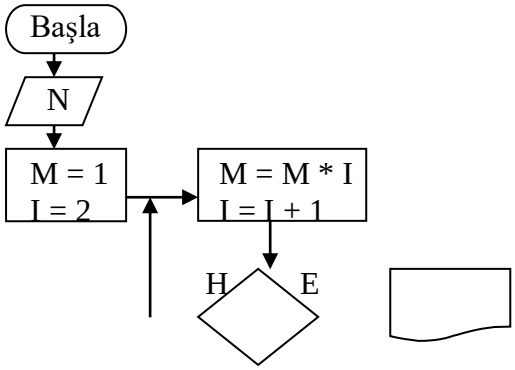
Soru 2 : Bir soyguncu bir sepet elma çalıyor. Kaçarken karşısına 3 bekçi çıkıyor. Bekçilere her defasında sepette bulunan elmanın yarısından 1 fazlasını veriyor. 3 bekçiye de bu yöntemle elma verdikten sonra hırsıza 1 elma kalıyor. Sepette ilk başta kaç elma olduğunu bulan programın algoritmasını yapınız ?

Örnek : M değeri 2 ve N değeri 5 ise ekrana ne yazılır ?

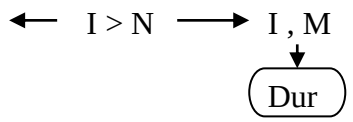


M	N	I	
2	5	1	* , 1
		2	* , 4
		3	* , 9
		4	* , 16
		5	* , 25
		6	

Örnek : N değeri 6 girilirse M ve I ' in son değeri ne olur ?



N	M	I
6	1	2
	2	3
	6	4
	24	5
	120	6
	720	7



Döngü Yapısı :

1. Tekrar sayısı belli olan döngüler:

FOR Döngü_değişkeni= Başlangıç_değeri TO Bitiş_değeri , Artım_miktarı

NEXT

Artan döngü : Başlangıç değeri bitiş değerinden küçük ise artan döngüdür.
Azalan döngü : Başlangıç değeri bitiş değerinden büyük ise azalan döngüdür.

Soru 3: Çarpım tablosu yazan programın algoritmasını oluşturup akış diyagramını çiziniz ?

2. Şarta bağlı döngüler (Tekrar sayısı belli olmayan döngüler) :

—Şart gerçekleşene kadar tekrar eden döngüler

sürece (koşul) yap (işlem)
while (koşul) do (işlem)

Not : Koşul sağlandığı sürece işlem tamamlanır.

Tekrarla (işlem) ta ki (koşul)
Repeat (işlem) until (koşul)

—İşlemi yap sonra koşula bak.

Sözde – Kodlar (Pseudo – code 'ler) :

1. Şarta (koşula) bağlı işlemler veya aktiviteler

— Eğer (koşul) doğru ise yap
değilse yap

if (koşul) then <işlemler>
if (koşul) then <işlemler>
else <işlemler>

Örnek : Girilen yaş değerine göre bebeği, çocuğu,genci, orta yaşlıyı,ihtiyarı bulan programı yazınız ?

0 – 5 Bebek	13 – 28 Genç	45 - .. İhtiyar
6 – 12 Çocuk	29 – 45 Orta yaşlı	

1 Başla
2 Yaşları giriniz (yas)
3 Eğer 0 ≤ yas ≤ 5 ise "bebek"
4 Eğer 6 ≤ yas ≤ 12 ise "çocuk"
5 Eğer 13 ≤ yas ≤ 28 ise "genç"
6 Eğer 29 ≤ yas ≤ 45 ise "ortayaşlı"
7 Eğer 46 ve üzeri ise "ihtiyar"
8 Dur

İf – Else yapısı :

```
İnput "Yaş değerini giriniz :" , Yas
İf (yas <= 5) Then  Print "bebek"
İf (yas <= 12) Then Print "çocuk"
İf (yas <= 28) Then Print "genç"
İf (yas <= 45) Then Print "orta yaşlı"
İf Else Print "ihtiyar"
End
```

Case Yapısı : (Pascal)

```
Uses crt;
Var yas: byte;
Begin
   Write('yaşı giriniz');read (yas);
   Case yas Of
      0 .. 5 :write ('bebek')
      6 ..12:write ('çoçuk')
      13 .. 28 : write (' Genç')
      29 .. 45 : write ('ortayaşlı')
   Else Write ('ihtiyar');
   End;
   End.
```

Quick Basic :

```
İnput "yaşı giriniz:" , yas
Select Case yas
Case 0 to 5 : Print "Bebek"
Case 6 to 13 : Print "Çocuk"
Case 14 to 28 : Print "Genç"
Case 29 to 45 : Print "ortayaşlı"
      Case Else  Print "İhtiyar"
End select
End
```

2. Koşul sürdükçe devam eden aktiviteler :

—Sürece (koşul) yap (işlemler)
While (koşul) **do** (işlemler)
—Tekrarla (işlemler) taki (koşul)
Repeat (işlemler) **until** (koşul)

} Pascal ' da kullanılır.

3. Anlamlı bir isme değer vermek :

Bir malzemenin fiyatı maliyet, kâr ve vergiyi toplayarak bulunur. Bunu komut halinde yaz.

Assign Toplam **Value** (maliyet + Kâr + vergi) (Toplam = maliyet + kâr + vergi)

Procedures : Algoritmaları sözde kodlarla sunmak veya bir programlama dilini çevirmek için kullanılan program dilimlerine alt program (subprogram) , alt yordam (subroutine), modül (module), fonksiyon (function) , yordam (procedure) gibi isimler verilir. Bu isimlerin hepsi değişik anlam ve içerik belirtmekle birlikte hepside geliştirilen bir sözde koda başlık vermek için kullanılır.

— Dürümsel yapılar : İçinde iterasyon bulunan popüler algoritmalarda en önemlileri arama (search) ve sıralama (sort) algoritmalardır.

1. Ardışık arama algoritması : Eğer alfabetik listede arama yapacaksak A<B<C<.....<Z aramaya ismi bulunca yada listedeki isimleri bulunca duracaktır.

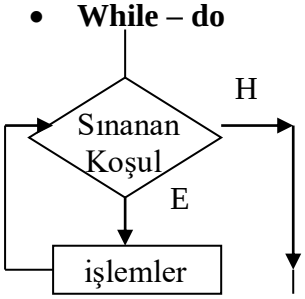
```
If (liste boş) then (arama başarısız) else
   ilk satırı test satırı olarak seç

[while (aranan değer <> test satırı And listede başka isimler varsa)
         do alt satır, test satırı olarak seç]

if (aranan değer = test satırı) then "Arama başarılı"
                                else "Arama başarısız"
```

Döngü yapıları :

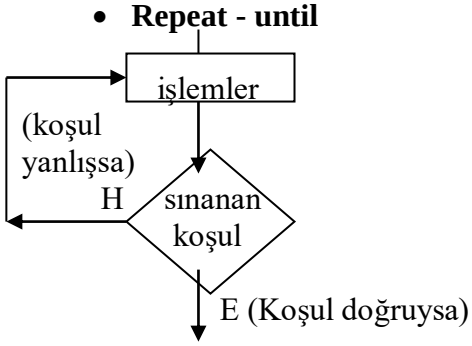
i. while (koşul) do (işlem)
ii. repeat (işlem) until (koşul)
iii. do – loop döngüleri

iv. for – next döngüsü

- **While – do**



- **Repeat - until**



- **Do – loop**

1- Do While (koşul) (işlemler) Loop	2- Do (işlemler) Loop until (koşul) (Doğruysa döngüden çıkar)
3- Do Until (koşul) (işlemler) Loop	4- Do (işlemler) Loop While (koşul) (Yanlışsa döngüden çıkar)

- **For – Next** (Döngü miktarı belli ise)

For döngü değişkeni = başlangıç **To** bitiş [step artım miktarı]
(işlemler)
Next Döngü değişkeni

Örnek :

```
A=5
Do while  A < 3
A = A + 1
Loop
Print A
```
→ Çıktısı 5

```
A = 5
Do
A = A + 1
Loop while A < 3
Print A
```
→ Çıktısı 6

```
A=5
Do until A < 3
A = A + 1
Loop
Print A
```
→ Çıktısı yok

```
A = 5
Do
A = A – 1
Loop until A < 3
Print A
```
→ Çıktısı 2

2. Yerleştirme Sıralı algoritması :

İnteraktif yapılar kullanılarak geliştirilen algoritmalardan bazılarda sıralama algoritmalardır. Bu algoritmalar elimizde bulunan gelişi güzel bir listeyi belli kurallar dahilinde düzenler.

Örnek : Listenin ikinci elemanından sonra gri yapınız .

Repeat (gri listedeki ilk elemanı beyaz yap ve bu elemana pivot adını ver)
Pivot elemanını geçici bir yere taşıyarak listede boşluk oluştur.
While (boşluğun üstündeki eleman, pivottan büyükse) **Do** (büyük elemanı aşağı kaydır.)
Oluşan boşluğa pivot yerleştir.
Until (listedeki elemanlaer beyaz ise)

BUBBLESORT:

İndisli değişkenlerde sayı veya alfabetik sıralama yapmakta kullanılan programlardan biri de BUBBLESORT dur. BUBBLE hava kabarcığı demektir. Suda hava kabarcıklarının peşpeşe yükselmesi gibi bu programda da büyük değerler peşpeşe yükselmesi gibi bu programda da büyük değerler peşpeşe liste sonuna varırlar.

Mesela; 5,10,2,4,12,1 rakamlarından oluşan dizinin rakamlarını bu yöntemle sıralayalım.

- Sırayla sayılar ikişer ikişer kıyaslanır. Önceki sonrakinden büyükse yer değiştirirler.

5,10,2,4,12,1
5,10,2,4,12,1 → 5,2,10,4,12,1
5,2,10,4,12,1 → 5,2,4,10,12,1
5,2,4,10,12,1
5,2,4,10,12,1 → 5,2,4,10,1,12

```
REM BUBBLESORT
DATA 5,10,2,4,12,1,1000
DIM A(10)
FOR I=1 TO10
READ A(I)
IF A(I)=1000 THEN GOTO SIRALA
NEXT I
SIRALA:
FOR N=I-1 TO 2 STEP -1
FOR M=1 TO N-1
IF A(M) <= A(M+1) THEN GOTO SIRASON
T=A(M) : A(M)=A(M+1):A(M+1)=T
NEXT M,N
FOR N=1 TO I-2
PRINT A(N) ; “,”;
NEXT N
PRINT A(I-1)
END
```

Quick Basic

Örnek : 1'den 100'e kadar olan sayıların toplamını bulan programı yapınız ?

İF

```
Cls
  T = 0 , S = 0
Git :
  S = S + 1
  T = T + S
  İf S <=  100  then goto Git
  Print  T
End.
```

Çıktısı : 5050

FOR

```
Cls
  T = 0
  For S = 1 to 100
  T = T + S
  Next S
  Print T
End
```

Çıktısı : 5050

WHİLE

```
Cls
  T = 0 ,S = 0
  While S <= 101
  S = S + 1
  T = T + S
  Wend
  Print T
End.
```

DO WHİLE

```
Cls
  T = 0, S = 0
  Do while S < 101
  S = S + 1
  T = T + S
  Exit Do
  Loop
  Print T
End.
```

Örnek : Dışarıdan girilen sayı 0' dan farklı olduğu sürece sayının karesini ve küpünü alan programı yazınız ?

İF

```
Cls
    İnput "sayıyı giriniz :" , A
    İf  A <> 0  then  print A^2, A^3
      else print "yanlış giriş"
End.
```

WHİLE

```
Cls
    İnput "sayıyı giriniz :" , A
    While A <> 0
    Print  A^2
    Print  A^3
    Cls
    İnput A
    Wend
End.
```

Örnek : Klavyeden sayı girildiği sürece bu sayının faktöriyelini bulup gösteren, sayı girmeden enter tuşuna basıldığında işlemi sonlandıran programı yapın ?

```
Tekrar:
    İnput "Sayı girişi :" , S
    İf  S=0 then END
    F=1
    FOR I = S  to  1 STEP- 1
    F = F * I
    NEXT I
    Print " sonuç :" , F
```

```
    Goto Tekrar
End
```

Örnek : Aşağıdaki program parçacıklarını P1, P2, P3 alt yordam (Procedure) B mantıksal değişken ise while – do döngüsüyle yapınız ?

PI;		P1;
REPEAT		P2;
P2 ;	While - Do	While (NOT B) Do
UNTIL B;		P2;
P3		P3;

Örnek : Çarpım tablosunu yazan programı yapınız ?

```
CLS
    FOR I = 1 TO 10
        FOR J = 1 TO 10
        Sonuç = I * J
        PRINT I , " * " , J , " = " , Sonuç
        NEXT  J
        SLEEP 2
    NEXT I
END
```

Örnek : İsteğe bağlı olarak üçgenin alanını veya dikdörtgenin alanını hesaplayan program yapınız?

```
CLS
    INPUT " Tabanı gir : ", taban
    PI = h
    PRINT " Üçgenin alanı : "
    Alan = ( h * taban ) / 2
    PRINT " Üçgenin alanı : " , Alan
END
```

SCREEN : Ekran parametresi

(0 - 13) → Ekranı text yada grafik moduna geçirir.

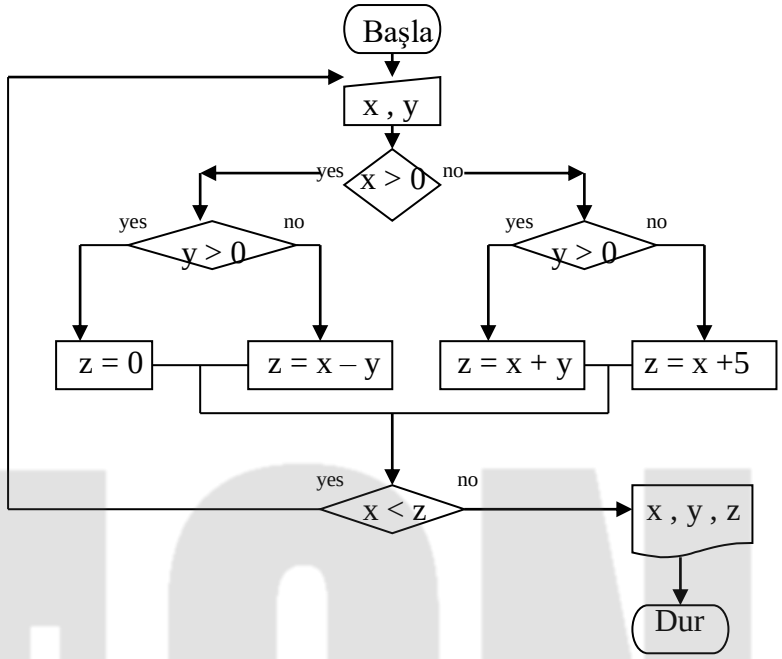
COLOR : Yazı rengi (0 – 15)

LOCATE : Belirtilen satır ve sütun no'ya gitmek için kullanılır.

Renkler:

1. Mavi
2. Yeşil
3. Açık mavi
4. Kırmızı
5. Mor
6. Kahverengi
7. Gri

Örnek :



b) cevap:

```
CLS
   Tekrarla :
   INPUT x,y
   IF  x > 0 THEN
       IF y > 0
           z = 0
       ELSE
           z = x – y
       END IF
   ELSE IF y > 0 THEN
           z = x + y
       ELSE
           z = x + 5
       END IF
   END IF
   IF x < z THEN GOTO Tekrarla
   PRINT x , y , z
END
```

a) X ve Y aşağıdaki değerleri aldığında çıkış değerleri ne olur ? (X,Y, Z)

X	**Y**
-5	2
3	-7
-4	-3
8	1

Son değerler

X = 8
Y = 1
Z = 0

b) Akış şeması çizilen programı Qbasic dilinde kodlayınız ?

PASCAL DİLİNİN YAPISI

1968 Niklams Wirth iki pascal çeşidi vardır.
1. Borland Pascal
2. Misrosoft Turbo Pascal

— Çok iyi bir yazılım geliştirme ortamına sahip olması (IDE)
— Program yazma kolaylığı
— Çok sayıda komuta sahip olması
— Programın kendi komut ve değişkenlerini tanımlamasına imkan vermesi
— Programcının hazırladığı Unitleri başka programlarda kullanabilme özelliği

Yazım Kuralları :

— Pascal' da büyük – küçük harf ayrımı yoktur.
— Her blok "Begin" ile başlayıp "End" ile sonlandırılmalıdır.
— Her Pascal ifadesi " ; " (noktalı virgül) ile bitirilmelidir. Yalnız begin , label, const, var, do gibi blok başlatan deyimlerden sonra kullanılamaz.
— İşlemler birbirini takip eden belli bir sıraya göre bloklar halinde yazılır.

Pascal Programlarının Genel Yapısı :

1. Program başlığı
2. Bilgi tanıtım bölümü
3. İşlem blokları bölümü

Program .....program başlığı ; ⟹ Program başlığı

Var.....değişkenin listesi ;
Const.....sabitlerin listesi ;
Type.....tiplerin listesi ;
Uses.....üniteler ;
Label.....etiket listesi ;
⟹ Bilgi tanıtım bölümü

Function .....;

Procedure.....;

Begin

End.
⟹ İşlem blokları bölümü

1. Program Başlığı : Programı tanıtım amacıyla kullanılır. Yani : programa isim vermek amacıyla kullanılır. 30 karakteri geçmeyecek şekilde isim verilmelidir. Zorunlu bir ifade değildir.

2. Bilgi Tanıtım Bölümü :

— **Uses :** Kullanılacak arşiv programlar (sistem veya unit olarak programcıların kendi oluşturdukları) tanımlar. Program başlığından sonraki satırda yer alır.
Uses arşiv prog.1, arşiv prog 2, arşiv prog 3.....;

Pascal da standart arşiv programları şunlardır:

1 - crt : Ekranla ilgili komut ve deyimler kullanılacaksa
2 – dos : Dos işletim sistemiyle ilgili komut ve deyimler kullanılacaksa
3 – groph : Grafik komut ve deyimleri kullanılacaksa
4 – graph : Turbo pascal 3.0 komut ve deyimleri kullanılacaksa
5 – overlay : Overlay dosyaların yönetimiyle ilgili komutlar kullanılacaksa
6 – printer : Yazıcı ile ilgili komut ve deyimler kullanılacaksa
7 – system : Bilgisayar sistemi ile ilgili komut ve deyimler kullanılacaksa
8 – turbo 3 : Yalnızca Turbo pascal 3.0 versiyonu ile uyumlu çalışır

— **Label :** Programın amacını yönlendirmek için etiket kullanılır

```
Label etiket 1, etiket 2, etiket 3.........;
```

— **Const :** Sabitleri tanımlamak için kullanılır.

```
Const PI, A ;
```

— **Var :** Program içerisinde kullanılarak değişkenlerin tipleriyle tanımlandığı kısımlardır.

```
Var     değişken 1 : Tipi ;
        değişken 2 : Tipi ;
        değişken 3 : Tipi ;
```

— **Type :** Kullanıcıların kendilerine ait özel bilgi tipleri tanımlamalarını veya dosya kayıt yapısındaki alan ve alan bilgi tanımlamaları bu blokta yapılır. Tanımlama var bloğundan önce yapılmalıdır. Tanımlanacak özel bilgi tipleri alfabetik karakterle başlamalı ve boşluk içermemelidir.

```
Type   özel bilgi adı: Tip ;
1- Type    kelime 5 : string [5]  ;
           ad 10    : string [10];
           matris   : integer    ;
```

2 – Kayıt (Record) tip tanımlanır.

```
Ad soyad : string [10]     ;
    No       : integer     ;
    Vize     : integer     ;
    Final    : integer     ;
End.
```

<u>3 – İşlem Blokları Bölümü :</u>

—**Procedure (Yardım) :** Ana programın yardımcı işlemlerin yapıldığı alt programlardır. Her biri başlı başına bir işlem tanımlar. Her hangi bir alt program çağrıldığı zaman (ana programa dönüş komutu exit olmadığı sürece) içindeki bütün komutlar sırayla işlem görürler. Alt programdaki bütün komutlar bittikten sonra program akışı , program çağıran komuttan sonraki satıra geçer. Yani ana program bütün alt programları çağırabilir. Bir alt program ancak kendinden daha alt seviyedeki başka alt programı çağırabilir. Alt programlar genellikle çağrılan satırdan önce tanımlanır. Daha sonra çağrılacaksa FORWAR komutu kullanılır. **2 şekilde tanımlanır.**

1 – Parametresiz Procedure (Alt Program) : Ana programdaki değişkenler ile çağrılan alt programdaki değişkenler aynıdır.

```
Var  değişkenle  ;
   Procedure Alt programismi;
Begin
        İsimler ;
End .
```

2 – Parametreli Procedure : Ana programdaki değişkenler ile çağrılan alt programlardaki değişkenler farklıdır. Büyük programlarda parametreli procedure kullanımı yazımı kolaylaştırır.

```
Procedure Alt programismi (var değişken listesi: ;)
```

```
            Begin
                İşlemler ;
            End .
```

Örnek : Girilen iki sayının karesini alan programı yapınız ?

Program parametresiz ;

```
Uses crt ,
Var x, y : integer ;
        Procedure kare ;
        Begin
           X = x * x ; y = y * y ;
        End;
Begin
        Clrscr ;
        Write (‘Birinci sayıyı giriniz , x’ ) ; readln (x) ;
        Write (‘İkinci sayıyı giriniz , y’) ; readln (y) ;
        Kare ;
        Write (‘Birinci sayının karesi ‘, x ) ;
        Wirte (‘İkinci sayının karesi ‘, y );
        Readln;
End .
```

Program parametreli ;

```
Uses crt ;
Var x , y : integer ;
        Procedure kare ( x, y : integer);
Begin
        z:= z * z ; k := k * k ;
End .
```

Fonksiyon (Function)

Bu alt programdan yalnızca bir sonuç alınır ve her functionun bir tipi vardır.

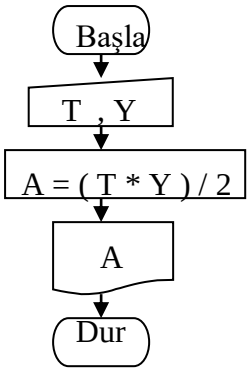
```
Function   altprogram_ismi   (değişken tipi) :  tipi ;

Begin
       İşlemler
       Değişken = altprogram_ismi
End ;
```

Örnek : Bir üçgenin Pisagor teoremini yazınız ?

Çözümü : $c^2 = a^2 + b^2$

```
Program fonksiyon;
Uses crt;
Var  x,y : integer ;
    Function pisagor ( x, y : integer ) ;
      Pisagor := sprt ( x*x + y*y );
Begin
    Clrscr;
    Write (‘Birinci dik kenarın değerini giriniz :’) ; readln (x ) ;
    Write (‘İkinci dik kenarın değerini giriniz :’) ; readln (y ) ;
    Writeln (‘ Pisagor :’ , pisagor (x,y) ) ; readln ;
End.
```

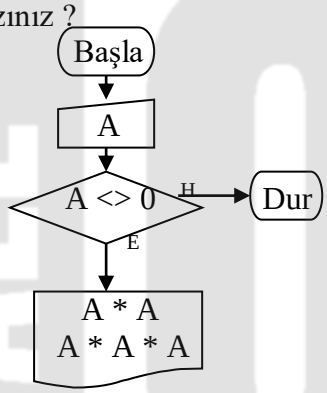
Örnek : Üçgenin alanını hesaplayan programı yazınız ?
Çözümü : Alan = (Taban * Yükseklik / 2)




```
Cls
    İnput "Taban ve Yüksekliği
            giriniz", T ,Y
    A = ( T * Y ) / 2
    Print "Üçgenin Alanı :" , A
End
```

```
Uses crt;
Var  T , Y : integer ;
     A : real;
Begin
     Write ( 'Taban ve yüksekliği
             giriniz:' )
     Readln ( T ) ; Readln ( Y ) ;
     A := ( T * Y ) / 2
     Write ( ' Üçgenin Alanı : ', A );
     Readln ;
End.
```

Örnek : Sayı 0' dan (sıfırdan) farklı olduğu sürece sayının karesini ve küpünü alan programı yazınız ?



1. Başla
2. Bir sayı giriniz (A)
3. Sayı 0' dan farklı olduğu sürece sayının karesini ve küpünü al.
4. Dur

```
Uses crt;
Var  x:integer ;
Begin
    Write ('Sayı giriniz : ', X );
    Readln ( X );
    İf  X <> 0 then
       Write (A^2 , A^3)
    Else End;
    Readln ;
End.
```

Örnek : e^x fonksiyonunun seriye göre açılımı aşağıdadır. Buna göre dışarıdan girilen X ve N değerine göre e^x ' i hesaplayan programı yapınız ?

$e^x = 1 + x^1/ 1! + x^2 / 2 ! + x^3 / 3 ! + x^4 / 4 ! + ... + x^N / N ! = \Sigma^N_{k=0} x^k / k !$

```
Cls
    İnput N, X
    T = 1 : F = 1
    For I = 1 to N
        F = F * I
        T = T + ( X ^ I / F)
        Print T
    Next I
    Print " e^x ' in değeri:", T
End.
```

```
Uses crt;
Var I, X, N : integer ;
T : real ;
Begin
        T:= 1 , F:= 1;
        Write ('N=');
        Readln (N);
        Write ('X=');
        Readln (X);
        For I=1 to N
        F=F * I;
        T= T +(X^I / F);
        Next
        Readln T
End.
```

DİZİLER :

Bellekte sürekli yer kaplayan aynı türden nesnelerin oluşturduğu kümeye dizi denir. Tanımdan da anlaşılacağı üzere bir dizinin dizi olabilmesi için;

1 – Bellekte yer kaplaması

2 – Aynı tür elemanlardan oluşmuş olması gerekir.

Dizilerin her elemanı ayrı bir nesne gibi ele alınmalıdır. Dizi elemanlarına erişmek amacıyla index operatörü kullanılmaktadır. Q basic' de diziler DIM komutu ile tanımlanır. Genelde dizilerin ilk elemanı 0. indisli elemanıdır.

DIM A(n) → n elemanlı A dizini

DIM A(5) → A dizisinin 5. indisli elemanı

İndislere isim verirken dikkat edilmesi gereken hususlar;

1 – İndisler negatif olamaz

2 – Tam sayı olmalıdırlar.

3 – İndisler en az bir tane olmalıdırlar. Maksimum indis sayısı kullanıcıya göre değişir.

```
DIM A(5)
A(0) = 100
A(1) = 50
A(2) = 70
...
A(5) = 80
```

Read : Değişkenlerin listesi

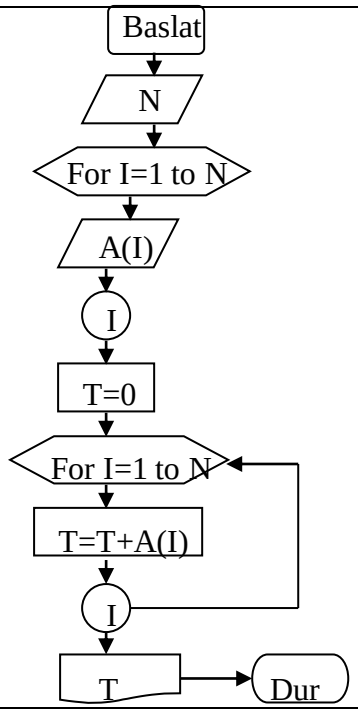
Data : Verilerin listesi

```
CLS
Read A, B, C $
Print A, B, C $
Data 15, 30,"Ali"
END
```

Örnek : Dışarıdan girilen N boyutlu dizinin tüm elemanlarını toplayan programı yapınız ?

1 – For - Next Dizi elemanlarının girilmesi

2 – For - Next Girilen dizinin elemanı ile ilgili işlemleri yapmak için




```
CLS
Input "Dizinin boyutunu giriniz ", N
DIM  A(N)
For I=1 to N
Print "A("; I ;") ="; Input "", A(I)
Next I
T=0
For I = 1 to N
T = T + A(I)
Next I
Print T
END
```

Örnek : A ve B gibi 10'ar elemanlı iki dizi veriliyor. A dizisinin aynı indisli elemanını B dizisinin aynı indisli elemanına bölen ve B=0 olması durumunda mesaj veren programı yapınız ?(Read-Data)

```
DIM A(10), B(10), C(10)
Data 10,9,8,7,6,5,4,3,2,1,1,2,3,4,5,6,7,8,9,10
For X=1 to 10
Read A(X)
Next X
For Y=1 to 10
Read (B)
Next Y
For I=1 to 10
If B(I)=0 then Print "B dizisinin elemanlarını 0'dan farklı girin"
C(I)=A(I) / B(I)
Next I
For I=1 to 10
Print C(I)
Next I
END
```

Örnek :

```
Read K
DIM X (K)
For A=1 to K
Read X(A)
Next A
Data 5,20,50,80,120,30
For A=1 to K-1
For B=A+1 to K
If X(B) > X(A) then goto Git
Z=X(A)
X(A)=X(B)
X(B)=Z
Git :
Next B,A
T=INT (X(K)-X(1))
If T>=100 then  T=T /10
Print:Print TAB(15) "ARALIK.....:"; T
Color 9.0 : Print  TAB(10) "SIRALI DİZİ"; Color=
Print
For I= 1 to K
Print TAB (8) X(I)
Next I
END
```

ARALIK	K	X(5)	A	B	X(2)>X(1)	T	X(5)	X(4)
SIRALI DİZİSİ	5	X(1)= 20	1	2	X(3)>X(2)	70	90	120
20		X(2)= 50	.	.	X(5)>X(4)			
50		X(3)= 80	.	.				
60		X(4)= 120	.	.				
90		X(5)= 90	4	5				
120								

Örnek :

A$ = “Niksar”
Print FIX (-3,75) →
Print INT (-3,75) →
Print CINT (37,50) → 38
Print CINT (8,49) → 8
Print LEFT$ (A$, 2) → Ni
Print RIGHT$ (A$, 2) → ar
Print MID$ (A$, 2 , 2) → ik
Print LEN$ (A$) → 6
Print TAB (5) A$ →.....Niksar
Print SPACE (10) A$ → .....Niksar (10 ttane boşluk bırakır)
Print ABS (-50) → 50 (mutlak değerini alır)
Print SQR (25) → 5 (sayının karekökünü alır)
Print LOG (25) → 3,25 (sayının logaritmasını alır)
Print SIN (90) → 1 (sayının sinüsünü alır)
Print COS (0) → 0 (sayının cosinüsünü alır)
Print SQRT (4) → 16 (sayının karesini alır)

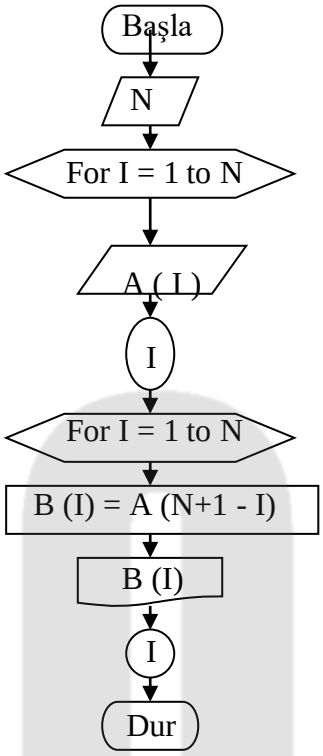
Örnek : N elemanlı bir dizinin elemanlarını küçükten büyüğe doğru sıralayan programı yapınız ?

```
Dim A (N)
For I = 1 to N
Print “A( “ ; I ; ” ) =” ; Input “  ” A(I)
Next I
For I = 1 to N-1
 For J = 1 to N
  İf A(J) < A(I) then SWAP  A(J), A(I)
 Next J
Next I
For I = 1 to N
 Print A(I)
Next I
END
```

Örnek : Dışarıdan girilen N elemanlı bir diziyi ters sırada B dizisine aktaran akış şemasını çizip, programını yapınız ?

```
Dim A(N), B(N)
İnput “ N değerini giriniz : ” , N
For I = 1 to N
 Print “A( ” ; I ; “) =” ; Input “ “ A(I)
Next I
For I = 1 to N
 B (I) = A (N+1 - I)
 Print “B ( “ ; I ; “ ) = “ ;Input “ ” A(I)
Next I
END
```

Akış diyagramı :



: İki Boyutlu Diziler (MATRİSLER) :

$$A_{i,j} = \begin{pmatrix} a_{11} & a_{12} & a_{13} & \ldots\ldots & a_{1j} \\ a_{21} & a_{22} & a_{23} & \ldots\ldots & a_{2j} \\ \ldots & \ldots & \ldots\ldots & \ldots\ldots & \ldots\ldots \\ a_{i1} & a_{i2} & a_{i3} & \ldots\ldots & a_{ij} \end{pmatrix}$$

eleman sayısı = i * j ile bulunur.

$A_{ij} = A (i , j)$

Satır sayısı → Sütun sayısı

A (3,5) = 15 elemanlı bir A dizisi (3 satır 5 sütün)

A (2,4) = 8 elemanlı bir A dizisi (2 satır 4 sütün)

$$A_{(3,4)} = \begin{pmatrix} 3 & 4 & 5 & 3 \\ 6 & 7 & 8 & 2 \\ 9 & 5 & 4 & 1 \end{pmatrix}$$

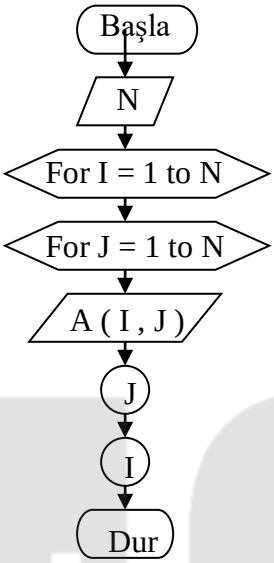
Satır esaslı :

3 4 5 3 6 7 8 2 9 5 4 1

Sütun esaslı :

3 6 9 4 7 5 5 8 4 3 2 1

N * N Dışardan iki boyutlu bir dizinin elemanlarının girilmesi



A (1,1) = ?	A (2,1) = ?	A (N,1) = ?
A (1,2) = ?	A (2,2) = ?	A (N,2) = ?
A (1,3) = ?	A (2,3) = ?	A (N,3) = ?
.....	.....	.....
A (1,N) = ?	A (2,N) = ?	A (N,N) = ?

```
CLS
    Input " N değerini giriniz :" , N
    Dim A ( N,N )
    For I = 1 to N
      For J = 1 to N
        Print "(";I;",";J;")=" ; Input " " A ( I , J )
      Next J
    Next I
END
```

<u>Örnek :</u> Dışarıdan girilen N * N tipinde A ve B matrislerini toplayan programı yapınız ?
(C = A + B)

```
CLS
İnput " N değerini giriniz :" , N
Dim A (N , N), B (N , N), C (N , N)
For I = 1 to N
    For J = 1 to N
      İnput " " A (I , J)
    Next J
Next I
For I = 1 to N
    For J = 1 to N
      İnput " " B (I , J)
    Next J
Next I
For I = 1 to N
    For J = 1 to N
      C (I , J) = A (I , J) + B (I , J)
      Input " " C (I , J)
    Next J
Next I
END
```

<u>Soru :</u> 10 kişilik bir öğrenci grubunun her birinin 1 dersten aldığı 3 notun (1.vize , 2. vize , final) ortalamasını alan programı yapınız ?

```
CLS
Dim A (10), B (10), C (10), Ort (10)
For I = 1 to 10
      İnput " " ,A ( I )
Next I
For I = 1 to 10
```

```
        İnput " " ,B ( I )
Next I
For I = 1 to 10
        İnput " " ,C ( I )
Next I
For I = 1 to 10
        Ort( I ) = A ( I ) + B ( I ) + C ( I ) / 3
Next I
For I = 1 to 10
        Print i + " .nci öğrencinin ortalaması "  , Ort ( I )
Next I
END
```

- **ÖRNEK PROGRAMLAR :**

```
REM TERSTEN YAZMA
PRINT " CÜMLEYİ GİRİN"
INPUT C$
FOR I=LEN (C$) TO 1 STEP -1
PRINT MID$(C$,I,1);
NEXT I
END
```

```
REM ASAL SAYILAR
CLS
INPUT "ASAL SAYI TAVAN RAKAMI=";S
PRINT : IF S<1 THEN PRINT " EN KÜÇÜK ASAL SAYI 1 DİR."
:END
PRINT 1, : IF S=1 THEN END
PRINT 2, : IF S=2 THEN END
FOR I=3 TO S
FOR N=2 TO I-1
I%=I/N : IF I-N*I%=0 THEN GOTO SON
NEXT N:PRINT I:NEXT I
END
```

KONTROL YAPILARI ÖZETİ:

- Sıralı (ardışık) - Sequence
- Karar (Seçim) - Selection
- Döngü (Tekrar) - Repetition

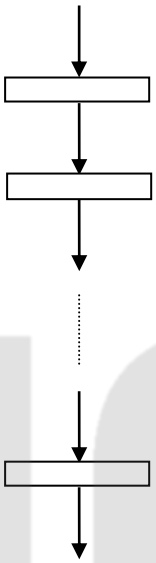
Sıralı yapı önemsizdir. Karar yapıları üç şekilde yapılabilir;

- **İf / Then** Yapısı (Tek seçim)
- **İf / Then / Else** Yapısı (Çift seçim)
- **Select Case** Yapısı (Çoklu seçim)

Döngü yapıları altı şekilde yapılabilir;

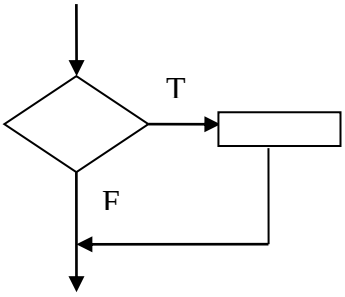
- **While / Wend** döngüsü
- **Do While / Loop** döngüsü
- **Do / Loop While** döngüsü
- **Do Until / Loop** döngüsü
- **Do / Loop Until** döngüsü
- **For / Next** döngüsü

Sıralı Yapı

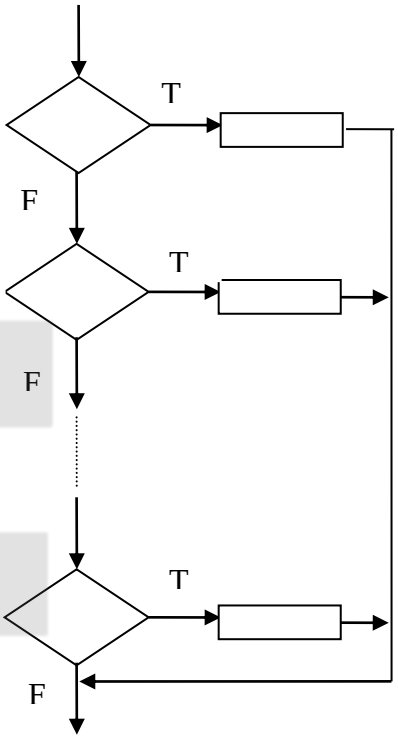


Karar Yapıları

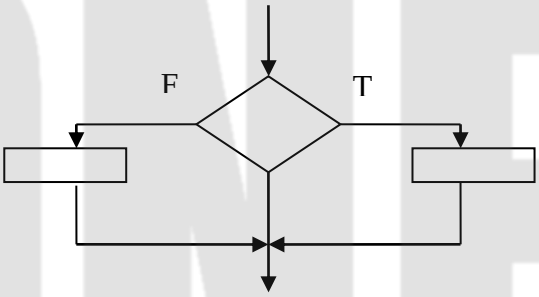
If / Then Yapısı
(Tek Seçim)



Select Case Yapısı
(Çoklu Seçim)



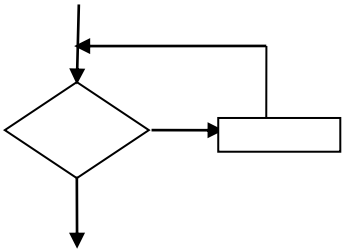
If / Then / Else Yapısı
(Çift Seçim



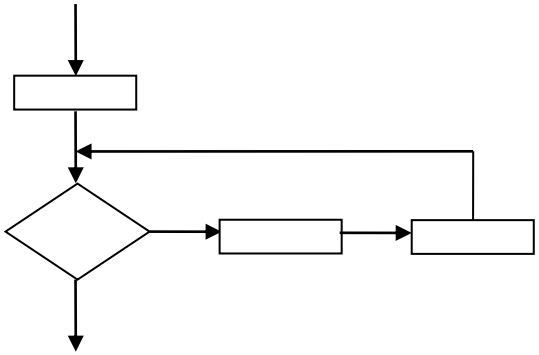
ALL ONE

Döngü Yapısı

While/Wend Yapısı



For/Next Yapısı



Do/Loop Until Yapısı

Do Until / Loop Yapısı

ALL ONE
ASİ_GENC