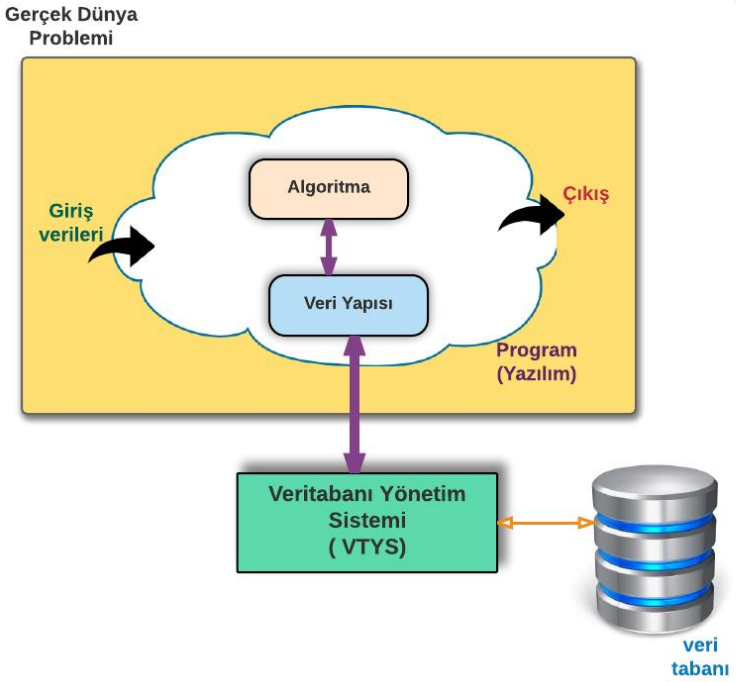
SAKARYA ÜNİVERSİTESİ

EKO458 - Python ve R ile Veri Analizi

HAFTA 2
Algoritmalar ve Akış Diyagramları

Arş. Gör. Dr. Muhammed KOTAN

Büyük Resim – Algoritmik Problem



Gerçek dünya problemlerini bilgisayar yardımıyla çözmek için **yazılım (program)** geliştirilir.

Algoritma Nedir?

- Gerçek hayattaki "plan" kelimesinin karşılığı olarak düşünülebilir.
- Bir problemin çözümüne yönelik işlem basamaklarının sıralı biçimde ve sonlu olarak ifade edilmesi
- Bilgisayar dilinde “ bir sorunun çözümü için öngörülen işlemlerin mantıksal ve sembolik anlatımı”
- 9. yüzyılda yaşamış Türk-İslam matematikçi ve astronomu El-Harezmi, toplama, çıkarma, ikiye bölme, bir sayının iki katını bulma, denklem çözümü, vb. gibi cebirsel işlemleri açıklayan bir çalışma yaptı. Batılılar, onun bu çalışmalarına Latince "algorismus" yani bugünkü adı ile **algoritma** dediler ve algoritma kavramı ilk defa burada kullanılmış oldu.
- Bir algoritma yazmak için mutlaka bir dile bağımlı kalma zorunluluğu yoktur. Önemli olan yazılan algoritmanın herhangi bir programlama diline uyarlanabilir olmasıdır.

Algoritma…

- Algoritmanın etkin bir şekilde oluşturulması Program yazma adımından çok daha önemlidir.
- Tasarlanan algoritma iyi değilse kullanılan programlama dilinin önemi yoktur. (C, Java, Python, C++)
- Bir problemin çözümü için birbirinden farklı birden fazla sayıda algoritma hazırlanabilir.
- Herhangi bir problemin çözümü için birbirinden farklı yüzlerce bilgisayar programı yazılabilir.

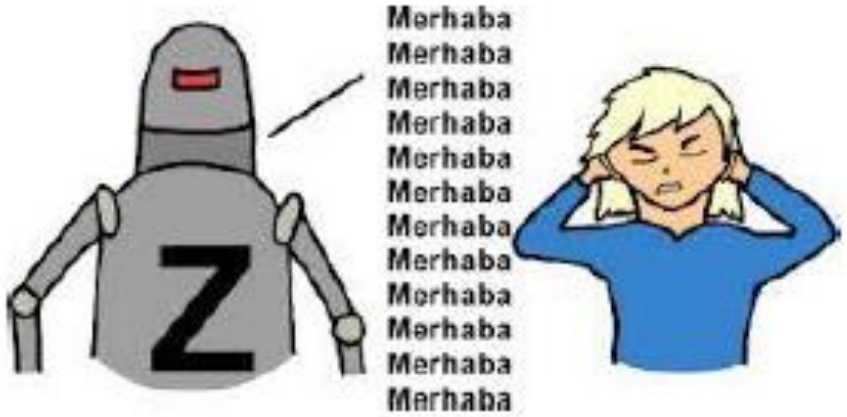
Algoritma…

- **İyi Bir Algoritmada Neler Olmalıdır?**
  - **Etkinlik**
  - **Sonluluk**
  - **Kesinlik**
  - **Giriş/Çıkış**
  - **Başarım/Performans**

Algoritma Özellikleri

- **Etkinlik**
  - Yazılan algoritmalar gereksiz tekrarlardan uzak oluşturulmalıdır.
  - Algoritmalar genel amaçlı yazılıp diğer algoritmalar içerisinde de kullanılabilir olmalıdır.
    - Örn: Verilen n adet sayının ortalamasını bulmakta kullandığımız algoritma varsa bu algoritma, bir sınıfta öğrencilerin yaş ortalamasını bulan bir algoritma için de kullanılabilmelidir.

Algoritma Özellikleri

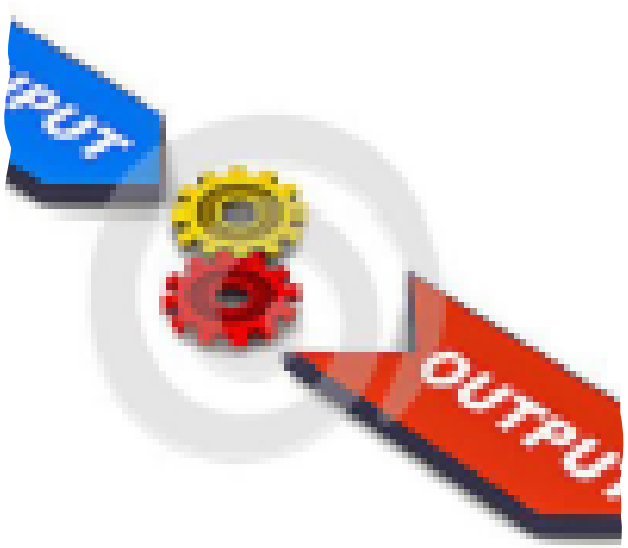


- **Sonluluk**
  - Her algoritma bir başlangıçtan oluşmalı, belirli işlem adımı içermeli ve bir bitiş noktasına sahip olmalıdır.
  - Olasılıklar değerlendirilerek algoritma sonlu adımda bitmelidir.
  - Kısır bir döngüye girmemelidir.

Algoritma Özellikleri

- **Kesinlik**
  - İşlem sonucu kesin olmalıdır.
  - Her adımı anlaşılır, basit ve kesin bir biçimde ifade edilmiş olmalıdır.
  - Yorum gerektirmemeli ve belirsiz ifadelere sahip olmamalıdır.

Algoritma Özellikleri



- **Giriş/Çıkış**
  - Algoritma giriş (üzerinde işlem yapılacak değerler) ve çıkış (yapılan işlemler neticesinde üretilen sonuç değerler) değerlerine sahip olmalıdır.
  - Algoritmaların temel amacı giriş bilgisini işleyerek çıkış bilgisi oluşturmaktır.
  - Bazı durumlarda bir algoritmanın çıkış bilgisi başka bir algoritmanın giriş bilgisi olarak kullanılabilir.

Algoritma Özellikleri

- **Başarım/Performans**
  - Amaç, donanım gereksinimi (bellek kullanımı, vs.), çalışma süresi, vs. gibi performans kriterlerini dikkate alarak yüksek başarımlı programlar yazmak olmalıdır.
  - Gereksiz tekrarlar ortadan kaldırılmalıdır. Bir algoritmanın performans değerlendirmesinde aşağıdaki temel kriterler göz önünde bulundurulur.
    - Birim İşlem Zamanı
    - Veri Arama ve Getirme Zamanı
    - Kıyaslama Zamanı
    - Aktarma Zamanı

Çay demleme algoritması?

Pilav yapma algoritması?

Çay Yapma Algoritması

Pilav Yapma Algoritması

Algoritma

- **Algoritmalar Farklı Şekillerde İfade Edilebilir mi?**
  - **Algoritmanın metin olarak yazılması**
    - Çözülecek problem, adım adım metin olarak yazılır.
  - **Sözde/Kaba Kod** (Pseudo Code)
    - Algoritmanın sözde kodlarla yazılmasıdır.
    - Problemin çözüm adımları komut benzeri anlaşılır metinlerle ifade edilir.
    - Yarı kod yarı metin olarak da adlandırılır.
    - Sözde kod, programlar gibi derlenmez ve işlenmez.
  - **Akış Diyagramı**
    - Problemin çözüm adımları geometrik şekiller ile ifade edilir.

Algoritma…

- **ÖRNEK**
  - Problem**:** Klavyeden girilen sayının karesini hesaplayarak ekrana yazdıran programın algoritmasını yazınız.
  - Algoritma:

    - **Başla**
    - **Sayıyı (A) gir**
    - **Sayının karesini hesapla (Kare = A*A işlemini yap)**
    - **Sonucu (Kare) yaz**
    - **Dur**

Algoritma…

- **ÖRNEK…**
  - Problem: Klavyeden girilen sayının karesini hesaplayarak ekrana yazdıran programın sözde kodunu yazınız.

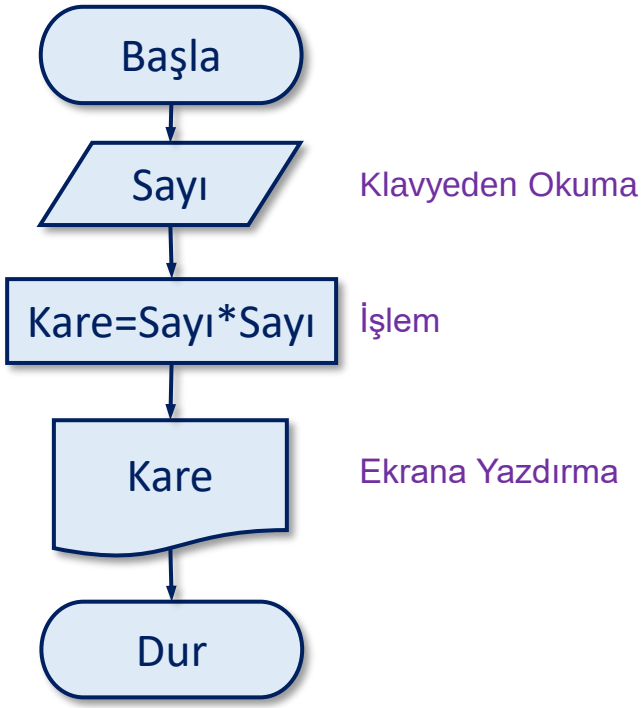
  - Sözde Kod:

    - **/* Kare alma programı */**
    - **read A**
    - **kare=A*A**
    - **print kare**
    - **end**

Algoritma…

- **ÖRNEK…**
  - Problem: Klavyeden girilen sayının karesini hesaplayarak ekrana yazdıran programın akış diyagramını çiziniz.

  - Akış Diyagramı:



Algoritma Tasarımında Kullanılan Terimler

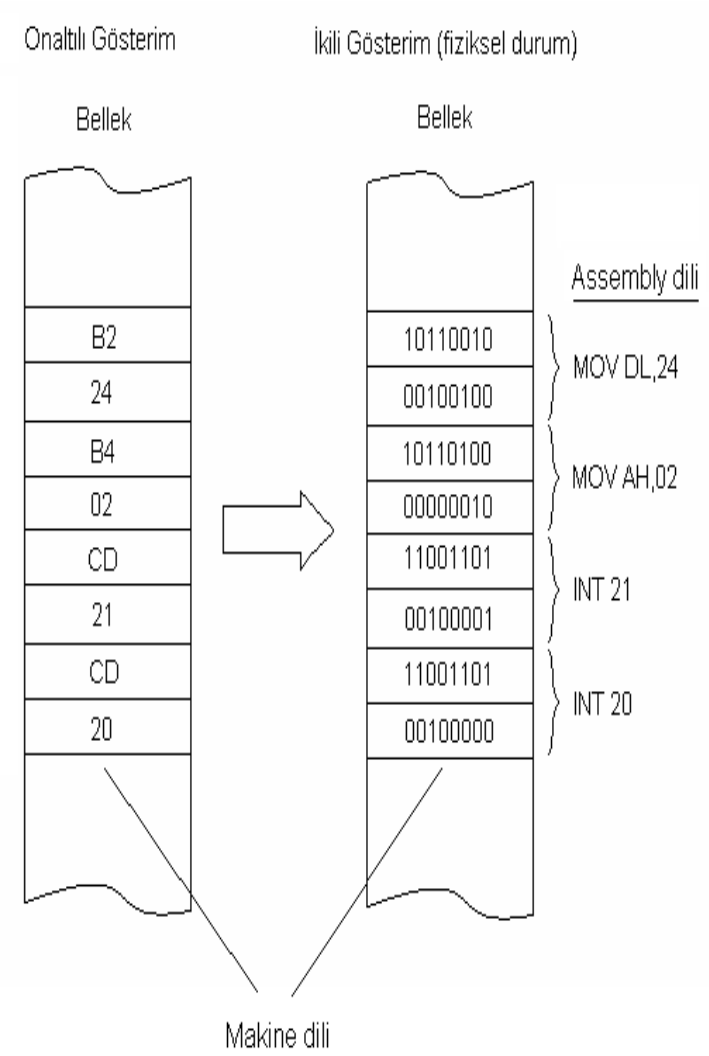
- **Operatör**
  - İşlemleri belirten yani veriler üzerinde işlem yapma özelliği olan simgelerdir.
  - Örnek:
    - + = >

Algoritma Tasarımında Kullanılan Terimler…

- **Tanımlayıcı**
  - Programcı tarafından oluşturulan/verilen ve programdaki değişkenleri, sabitleri (değişmez), sınıf, nesne, özel bilgi tiplerini, alt programları vb. adlandırmak için kullanılan kelimelerdir.
  - Tanımlayıcı, yerini tutacağı ifadeye çağrışım yapmalıdır.
  - Tanımlayıcı kelimeler oluşturulurken uyulması gereken kurallar:
    - İngiliz alfabesindeki A-Z veya a-z arası 26 harf kullanılabilir
    - 0-9 arası rakamlar kullanılabilir
    - Simgelerden sadece alt çizgi (_) kullanılabilir
    - Tanımlayıcı isimleri, harf veya alt çizgi ile başlayabilir
    - Rakam ile başlayamaz veya sadece rakamlardan oluşamaz
    - Kullanılan programlama dilinin komutu ya da saklı kelimelerinden olamaz

Algoritma Tasarımında Kullanılan Terimler…

- **Değişken**
  - Verilerin saklandığı bellek alanlarına verilen simgesel isimlerdir.
  - Programın her çalıştırılmasında, farklı değerler alabilen/aktarılabilen bilgi/bellek alanlarıdır.
  - Bir programda birbirinden farklı kaç tane veri tutulacak ise o kadar değişken tanımlanmalıdır.
  - Değişkenleri adlandırma, tamamen programcının isteğine bağlıdır.
  - Değişken adlandırmada tanımlayıcı isimlendirme kuralları geçerlidir.
  - Değişken adının, yerini aldığı ifadeyi çağrışım yapması programın anlaşılırlığı açısından önemlidir.
    - Örnek:
      - Bir kişinin adını tutmak için -> isim,
      - doğum tarihini tutmak için -> dogumTarihi

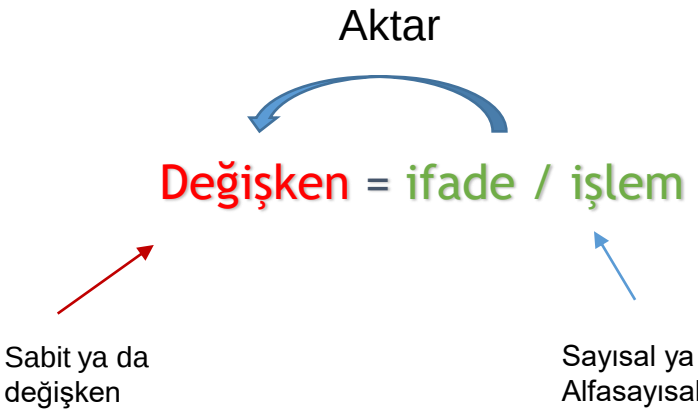


Algoritma Tasarımında Kullanılan Terimler

- **Sabit**
  - Programlardaki değeri değişmeyen ifadelere sabit denir.
  - Sabit adlandırmalarında da tanımlayıcı kuralları geçerlidir.
  - Sabitlere değer aktarma
    - Sayısal veriler doğrudan aktarılır.
      - Örnek: SabitKomutu pi = 3.14
    - Alfasayısal (karakter) veriler ise tek/çift tırnak içerisinde aktarılır.
      - Örnek: SabitKomutu ilkharf = 'A'
      - Örnek: SabitKomutu okulAdi = "Sakarya"

Algoritma Tasarımında Kullanılan Terimler

- **Aktarma**
  - Bir bilgi alanına, veri yazma
  - Bir ifadenin sonucunu başka bir değişkende gösterme vb. görevlerde aktarma " = " operatörü kullanılır.
  - Kullanım şekli:



Örnek:

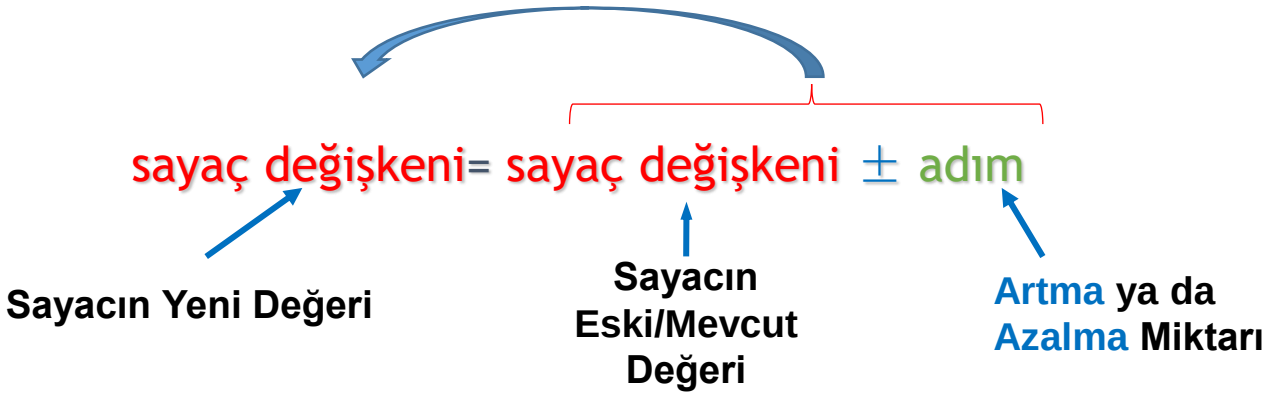
- Sayısal ifade olarak
  - A= 2, B= 3
  - C = A + B
- Alfasayısal ifade olarak
  - A = "Sak" , B = "arya"
  - C = A+B

C=5

C="Sakarya"

Algoritma Tasarımında Kullanılan Terimler

- **Sayaç / Sayıcı**
  - Belirli sayıda yapılması istenen işlemleri takip etmek için
  - İşlenen ya da üretilen değerlerin sayılması gerektiği durumlarda kullanılır.
  - Örnek: Rasgele oluşturulmuş bir dizideki tek sayıların tespitinde tek sayıların adedini belirlemek için sayaç kullanılır.
  - Kullanım şekli:



Örnek:

- Say = Say + 1

Algoritma Tasarımında Kullanılan İşlemler

Matematiksel (Aritmetik) İşlemler

İşlem	Matematik	Bilgisayar
Toplama	a + b	a + b
Çıkarma	a - b	a - b
Çarpma	a . b	a * b
Bölme	a ÷ b	a / b
Üs Alma	a^b	a ^ b

Algoritma Tasarımında Kullanılan İşlemler

Matematiksel (Aritmetik) İşlemler…

MATEMATİKSEL YAZILIM	BİLGİSAYARA KODLANMASI
a+b-c+2abc-7	*a+b-c+2*a*b*c-7*
a+b²-c³	*a+b^2-c^3*
$a - \frac{b}{c} + 2ac - \frac{2}{a+b}$	*a-b/c+2*a*c-2/(a+b)*
$\sqrt{a+b} - \frac{2}{b^2 - 4ac}$	*(a+b)^(1/2)-2/(b^2-4*a*c)*
$\frac{a^2 + b^2}{2ab}$	*(a^2+b^2)/(2*a*b)*

Algoritma Tasarımında Kullanılan İşlemler…

- **Karşılaştırma İşlemleri**
  - Büyüklüklerin karşılaştırılması amacıyla kullanılır.
  - Karşılaştırma İşlemlerinin Bilgisayar Dilindeki Karşılıkları

Sembol	Anlamı
==	Eşittir
<> Veya !=	Eşit Değildir
>	Büyüktür
<	Küçüktür
>= veya =>	Büyük eşittir
<= veya =<	Küçük eşittir

➢ **Örnek:**

Eğer X > Y ise Yaz " X sayısı Y'den büyüktür "

Algoritma Tasarımında Kullanılan İşlemler…

- **Karşılaştırma İşlemleri…**
  - Sayısal karşılaştırmalarda doğrudan değerler karşılaştırılır.
    - Örnek: 35 > 15 “35 sayısı 15 ten büyüktür”
  - Karakter olarak karşılaştırmalarda ise karşılaştırma işlemine ilk karakterlerden başlanılarak sıra ile karşılaştırılır.
    - Örnek: a > c “1. karakter alfabetik olarak daha önde”

  - Not: Karakter karşılaştırma işlemlerinde, karşılaştırma karakterler arasında değil, karakterlerin ASCII kodları arasında yapılır. Örneğin, A nın ASCII kod karşılığı 65 tir. a nın ise ASCII kod karşılığı 97 dir. Büyük ve küçük harfler arasındaki ASCII kod farkı ise 32’dir.

Algoritma Tasarımında Kullanılan İşlemler…

- **Mantıksal İşlemler**
  - Temel Mantıksal İşlem Karşılıkları

İşlem	Komut	Anlamı
VE	AND	Koşulların hepsi doğru ise sonuç doğrudur
VEYA	OR	Koşullardan en az biri doğru ise sonuç doğrudur
DEĞİL	NOT	Sonuç koşulun tersidir. 1 ise 0 dır

  - Mantıksal İşlemlerde Öncelik Sıraları

Sıra	İşlem	Komut
1	Parantez İçindeki İşlemler	(…..)
2	DEĞİL	NOT
3	VE	AND
4	VEYA	OR

Algoritma Tasarımında Kullanılan İşlemler…

- **Mantıksal İşlemler…**
  - Örnek: Bir işyerinde çalışan işçiler arasından yalnızca yaşı 23 üzerinde olup, maaş olarak asgari ücret alanların isimleri istenebilir.
  - Burada iki koşul vardır ve bu iki koşulun da doğru olması gerekir. Yani;

EĞER Yaş>23 VE maaş=asgari ücret ise YAZ

1.KOŞUL *2.KOŞUL*

- *YAZ komutu 1. ve 2.koşulun her ikisi de sağlanıyorsa çalışır*

Algoritma Tasarımında Kullanılan İşlemler…

- **Mantıksal İşlemler…**
  - Örnek: Bir sınıfta Programlama dersinden 65 in üzerinde not alıp, Türk Dili veya Yabancı Dil derslerinin herhangi birinden 65 in üzerinde not alanların isimleri istenmektedir.
  - Burada 3 koşul vardır.
    - Programlama dersinden 65 in üzerinde not almış olmak temel koşuldur.
    - Diğer iki dersin notlarının herhangi birinin 65 in üzerinde olması gerekir.

EĞER Prog_not>65 VE (TDili_not>65 VEYA YDil_not>65) ismi YAZ

Algoritma Tasarımında Kullanılan İşlemler…

- **Sorgu / Seçme**
  - Algoritmada, işlemler genellikle sıralı adımlardan oluşur.
  - Bazı koşul/şart durumlarında işlem sıralarının değiştirilmesi ve diğer bir işlem sırasının seçilmesi yada programın yeni işlem sırasından devam etmesi gerekebilir.
  - İşlem sıralarının değiştirilmesi ya da bazı koşulların sorgulanması için 'EĞER' sorgu deyimi kullanılır.
  - Örnek:

EĞER ORT > 50 GİT ADIM 7

KOŞUL *Gidilecek İşlem Sırası*

Algoritma Tasarımında Kullanılan İşlemler/Yapılar

- **Sorgu / Seçme**
  - Örnek: Dışarıdan girilen iki sayıyı kıyaslayan programın algoritmasını çıkarınız?

1. **Başla**
2. **OKU X, Y**
3. **Eğer X > Y ise YAZ "X BÜYÜK" Git 6**
4. **Eğer X < Y ise YAZ "X KÜÇÜK" Git 6**
5. **YAZ "EŞİT"**
6. **Dur**

Algoritma Tasarımında Kullanılan İşlemler/Yapılar

- **Sorgu / Seçme…**
  - Örnek: Dışarıdan 10'dan büyük 20'den küçük sayı girilene kadar programı devam ettiren algoritmayı yazınız.

1. **Başla**
2. **OKU X**
3. **Eğer (X > 10 VE X < 20) ise Git 6**
4. **YAZ " SAYI 10 İLE 20 ARASINDA DEĞİLDİR"**
5. **GİT 2**
6. **YAZ "SAYI 10 İLE 20 ARASINDADIR"**
7. **Dur**

Algoritma Tasarımında Kullanılan İşlemler/Yapılar

- **Döngü**
  - Bazı işlemleri belirli sayıda tekrar etmede,
  - Belirli bir aralıktaki ardışık değerler ile işlem yapmada döngü kullanılır.
  - Diğer bir deyişle, programdaki belirli işlem bloklarını, verilen sayıda gerçekleştiren işlem akış çevrimlerine DÖNGÜ denir.

  - Örnek:
    - **1'den 5'e kadar sayıların toplamı**
    - **Ard arda ekrana 4 defa SAKARYA yazdırma**
    - **1 ile 100 arasındaki çift sayıların ya da tek sayıların toplamı**

Algoritma Tasarımında Kullanılan İşlemler/Yapılar

- **Döngü…**
  - Döngü oluşturma kuralları
    1. Döngü değişkenine başlangıç değeri verilir
    2. Döngünün artma ya da azalma miktarı belirlenir
    3. Döngünün bitiş değeri belirlenir
    4. Eğer Döngü karar ifadeleriyle oluşturulduysa; döngü değişkeni, döngü içinde adım miktarı kadar arttırılmalı/azaltılmalıdır.

  - Örnek:

1. **Başla**
2. **T = 0**
3. **J = 1**
4. **Eğer J > 10 ise Git 8**
5. **T = T + J**
6. **J = J + 2**
7. **Git 4**
8. **Yaz T**
9. **Dur**

Algoritma Tasarımında Kullanılan İşlemler/Yapılar

- **Döngü…**
  - Döngü oluşturma kuralları
    1. Döngü değişkenine başlangıç değeri verilir
    2. Döngünün artma ya da azalma miktarı belirlenir
    3. Döngünün bitiş değeri belirlenir
    4. Eğer Döngü karar ifadeleriyle oluşturulduysa; döngü değişkeni, döngü içinde adım miktarı kadar arttırılmalı/azaltılmalıdır.

  - Örnek: 1-10 arası tek sayıların toplamı hesaplayan programın algoritmasını çıkarınız?

1. **Başla**
2. **T = 0**
3. **J = 1**
4. **Eğer J > 10 ise Git 8**
5. **T = T + J**
6. **J = J + 2**
7. **Git 4**
8. **Yaz T**
9. **Dur**

Akış Diyagramı

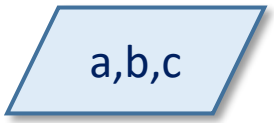
- Akış diyagramı, algoritmaların geometrik şekillerle ortaya konulmasıdır.
- Akış diyagramı, problemin çözümü için yapılması gerekenlerin, başından sonuna kadar geometrik şekillerden oluşan semboller ile ifade edilmesidir.
- Her simge genel olarak yapılacak bir işi veya komutu gösterir.
- Algoritma doğal dille yazıldığı için herkes tarafından anlaşılmayabilir ancak akış diyagramlarında her bir şekil standart bir anlam taşıdığı için farklı yorumlanmaz.

Akış Diyagramında Kullanılan Semboller

- **Başla**
  - Programın nereden başlayacağını belirtir. Standart olarak her bağımsız algoritmada bir tane bulunur

- **Dur**
  - Programın nerede sonlanacağını belirtir. Birden fazla olabilir. Mümkün ise sadece bir tane dur simgesi kullanılmalıdır.

- **Giriş**
  - Bilgisayara dışarıdan bilgi girişini temsil eder. Bu sembolün içine dışarıdan girilen bilgilerin aktarılacağı değişkenler yazılır.



Akış Diyagramında Kullanılan Semboller

- **Çıkış**
  - Ekrana veya yazıcıya bilgi göndermeyi temsil eder

- **İşlem**
  - Programın işlemesi sırasında yapılacak işlemleri ifade etmek için kullanılır

- **Karşılaştırma (sorgu)**
  - Verilen koşulun sonucuna göre farklı işlem yapılacağını ifade etmek için kullanılır.

Akış Diyagramında Kullanılan Semboller

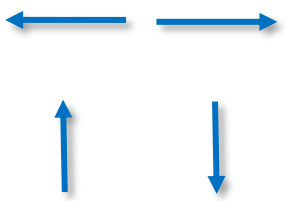
- **Döngü**
  - Belirli bir işin veya bir grup işin birden çok yinelenmesi gerektiğinde kullanılır. Döngüdeki çevrim sayısı, döngü sayacı ve sayaç artırımı açıkça yazılır.

- **Fonksiyon Çağırma**
  - Daha önce oluşturulmuş bir algoritmanın yazılan algoritma içerisine konulmadan çağrılarak kullanılmasını ifade eder.

- **Dosyaya Saklama**
  - Elde edilen bilgilerin bir dosyada saklanması veya dosyadan okunmasını simgeler.

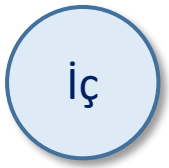
Akış Diyagramında Kullanılan Semboller

- **Akış Yönü**
  - Bir işlem bittikten sonra akışın nereye yönleneceğini belirler.



- **Bağlantı**
  - Akış diyagramı çizilirken sayfaya sığmama durumunda çizimin başka bir yerden devam etmesi için kullanılır.

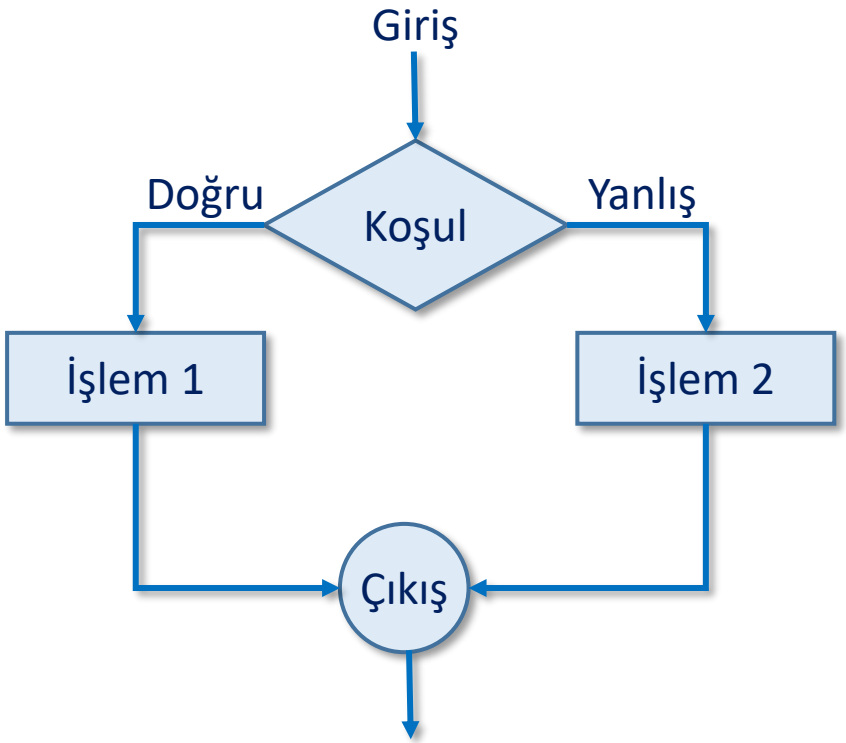
Aynı sayfaya



Ayrı sayfaya

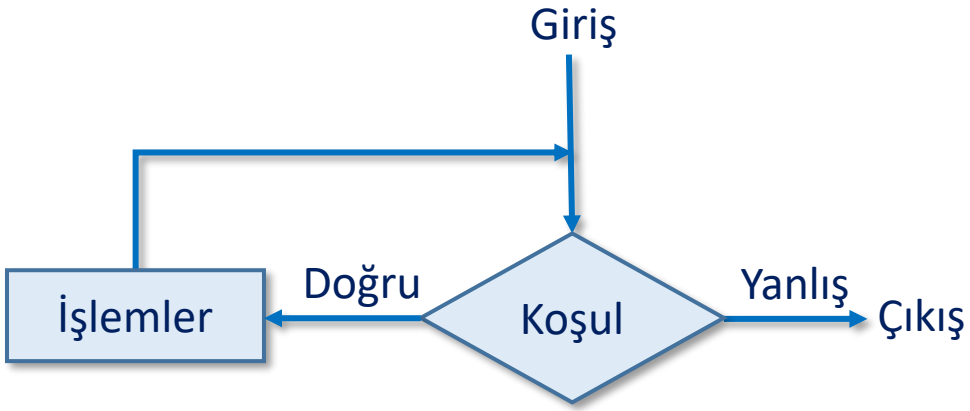
Akış Diyagramında Kullanılan Semboller

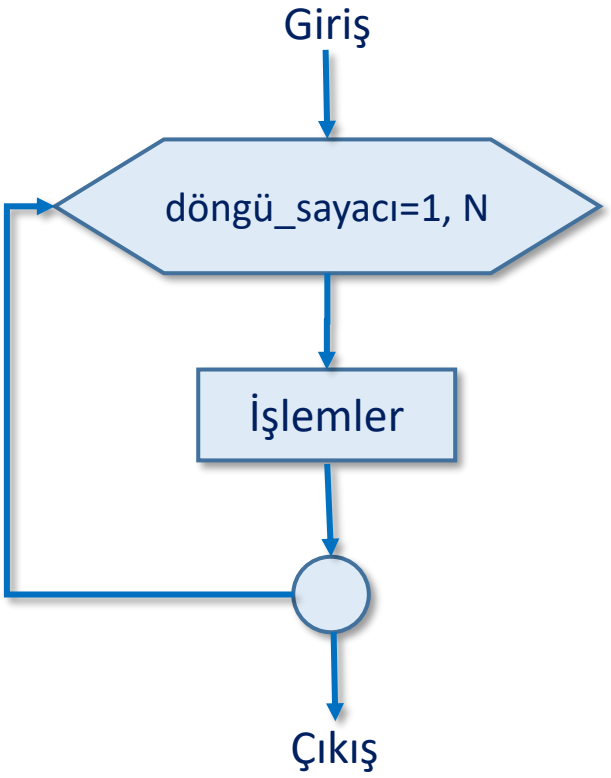
- Algoritma tasarımındaki EĞER yapısının gerçekleştirilmesi



Akış Diyagramında Kullanılan Semboller

- Algoritma tasarımındaki Döngü yapısının gerçekleştirilmesi





Akış Diyagramı Örnekleri - 1

- **Problem:** 1'den N'ye kadar olan sayıların toplamını hesaplayan programın akış diyagramını çizelim.

Akış Diyagramı Örnekleri - 1

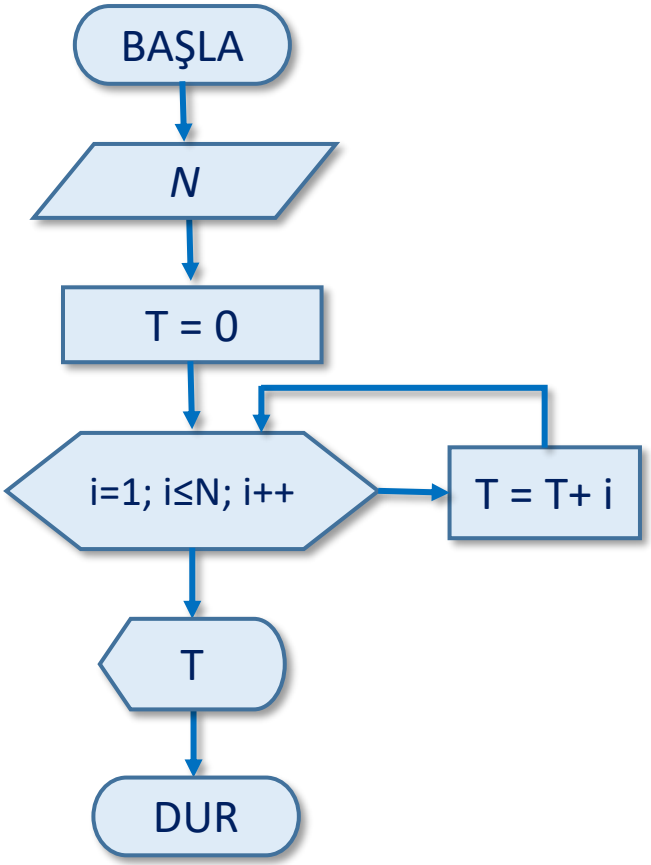
- **Problem:** 1’den N’ye kadar olan sayıların toplamını hesaplayan programın akış diyagramını çizelim.

- 1’den N’ye kadar, N adet sayı vardır. Birer artan döngü içinde sayıları toplayabiliriz. Döngü artışını kontrol edeceğimiz değişken i olsun. Toplam değerini de T değişkeni ile ifade edelim. Döngü değişkeni i, 1’den başlayacak ve birer artarak N ye ulaşacak. T başlangıçta 0 ile başlayacak ve döngü içerisinde 1’den N’ye değişen i değeri ilave edilecek.

Akış Diyagramı Örnekleri – 1…

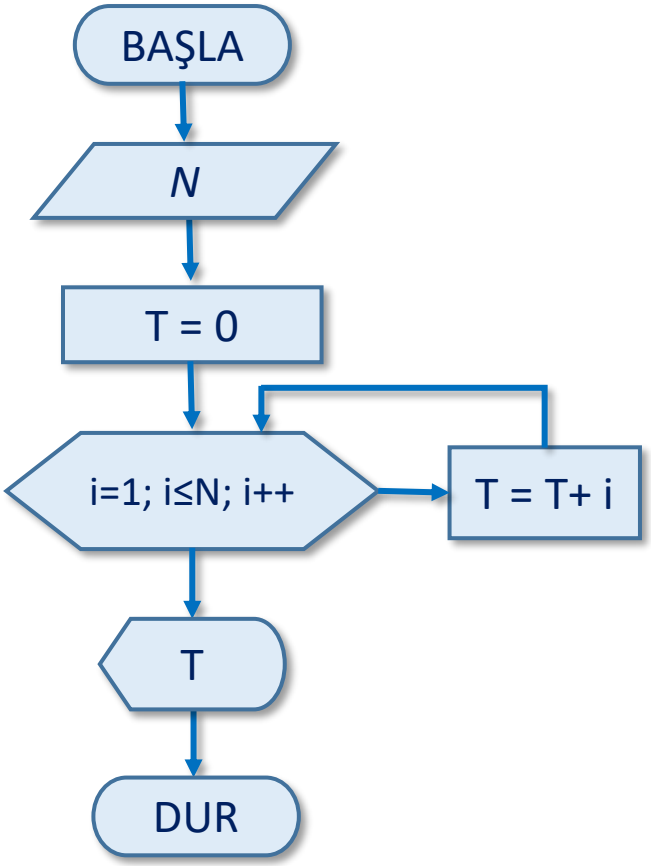
- **Problemin Algoritması:**
  - Algoritma sözde kodlar ile ifade edildiğinde aşağıdaki şekilde yazılabilir:

- Adım 1: Başla
- Adım 2: Oku (N)
- Adım 3: T = 0, i = 0
- Adım 4: T = T + i
- Adım 5: i = i + 1
- Adım 6: Eğer $i \leq N$ ise Adım 4'ye git
- Adım 7: Yaz (T)
- Adım 8: Dur



i	N	işlem	T

Akış Diyagramı Örnekleri – 1…



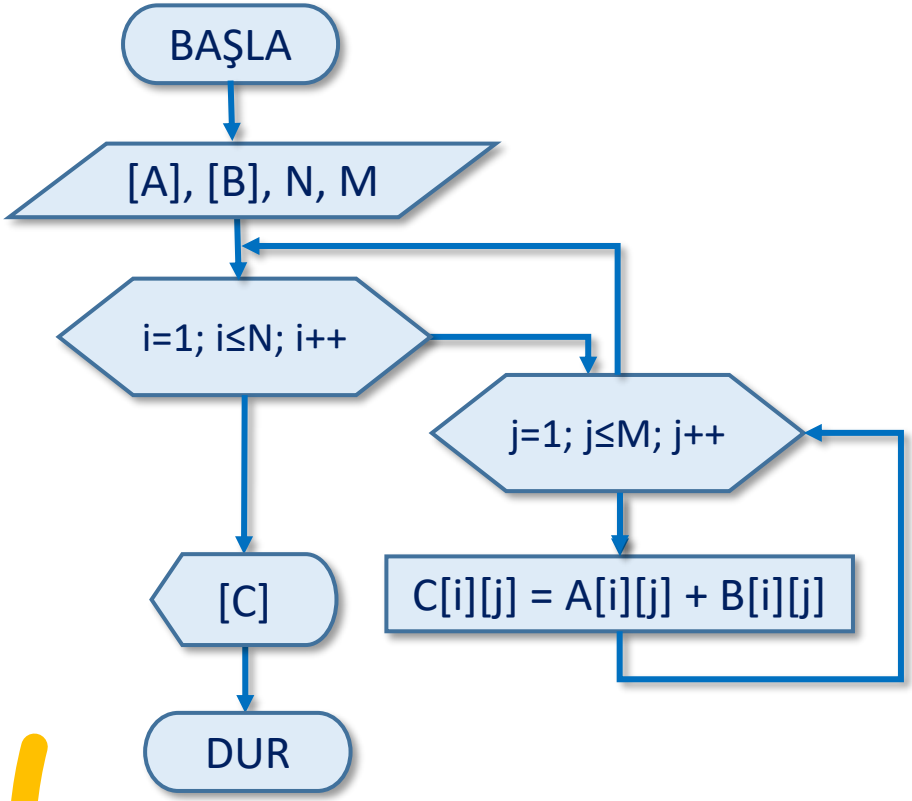
i	N	işlem	T
-	10	-	-
-	10	T=0	0
1	10	T=0+1	1
2	10	T=1+2	3
3	10	T=3+3	6
4	10	T=6+4	10
5	10	T=10+5	15
6	10	T=15+6	21
7	10	T=21+7	28
8	10	T=28+8	36
9	10	T=36+9	45
10	10	T=45+10	55

Akış Diyagramı Örnekleri – 1…

Akış Diyagramı Örnekleri – 2

- **Problem:** NxM boyutlu iki matrisin toplamını hesaplayan bir algoritmanın akış diyagramını çizelim.

- Matrislerin elemanlarını ifade eden indisleri i ve j ile gösterelim (i=1,...,N, j=1,...,M). Bu durumda A matrisinin her bir elemanı matematiksel olarak $A_{i,j}$ veya programlama açısından A[i][j] şeklinde ifade edilebilir.
- Elemanları toplanacak matrisler A ve B matrisleri ve toplam sonucunda oluşacak matris ise C matrisi olsun.
- Bilindiği gibi matris toplamında birinci matrisin [i][j] indeksli elemanı ile ikinci matrisin [i][j] indeksli elemanı karşılıklı olarak toplanarak toplam matrisin [i][j] indeksli elemanını elde edilir.



Akış Diyagramı Örnekleri – 2…

$$[A]=\begin{bmatrix}1 & 3\\ 1 & 2\end{bmatrix} \quad [B]=\begin{bmatrix}2 & 1\\ 3 & 0\end{bmatrix}$$

matrisleri ve N=M=2 değerleri için

i	j	işlem	C[i][j]
1	1	C[1][1]=1+2	3
1	2	C[1][2]=3+1	4
2	1	C[2][1]=1+3	4
2	2	C[2][2]=2+0	2

$$[C]=\begin{bmatrix}3 & 4\\ 4 & 2\end{bmatrix}$$

ÖRNEK-1

- Ekrana "Merhaba Dünya" yazan programın algoritmasını yazınız.

ÖRNEK-1

- Ekrana "Merhaba Dünya" yazan programın algoritmasını yazınız.

1. BAŞLA
2. YAZ "Merhaba Dünya"
3. DUR

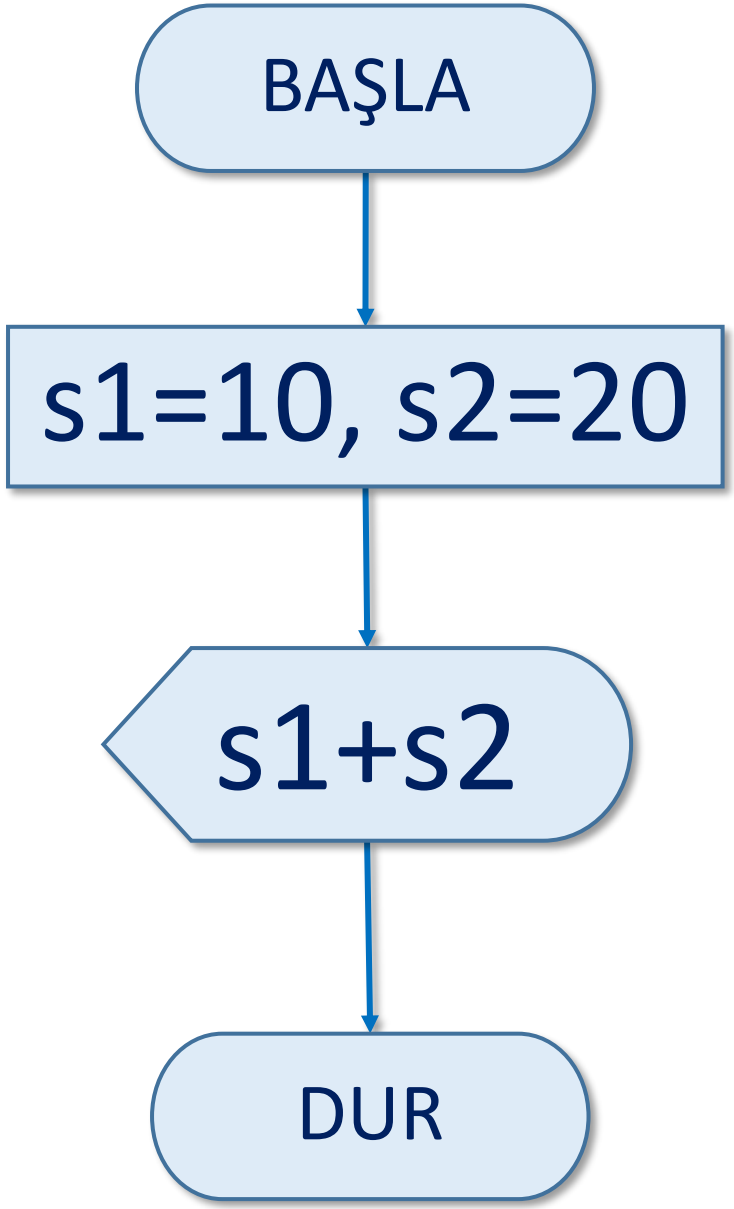
ÖRNEK-2

- İki sayıyı toplayıp sonucunu ekrana yazdıran yazan programın algoritmasını yazınız ve akış diyagramını çiziniz.

ÖRNEK-2

- İki sayıyı toplayıp sonucunu ekrana yazdıran yazan programın algoritmasını yazınız ve akış diyagramını çiziniz.

1. BAŞLA
2. s1=10, s2=20
3. YAZ s1+s2
4. DUR



ÖRNEK-2

- İki sayıyı toplayıp sonucunu ekrana yazdıran yazan programın algoritmasını yazınız ve akış diyagramını çiziniz.

1. BAŞLA
2. s1=10, s2=20
3. YAZ s1+s2
4. DUR

ÖRNEK-3

- Klavyeden girilen iki sayıyı toplayıp sonucunu ekrana yazdıran yazan programın algoritmasını yazınız ve akış diyagramını çiziniz.

ÖRNEK-3

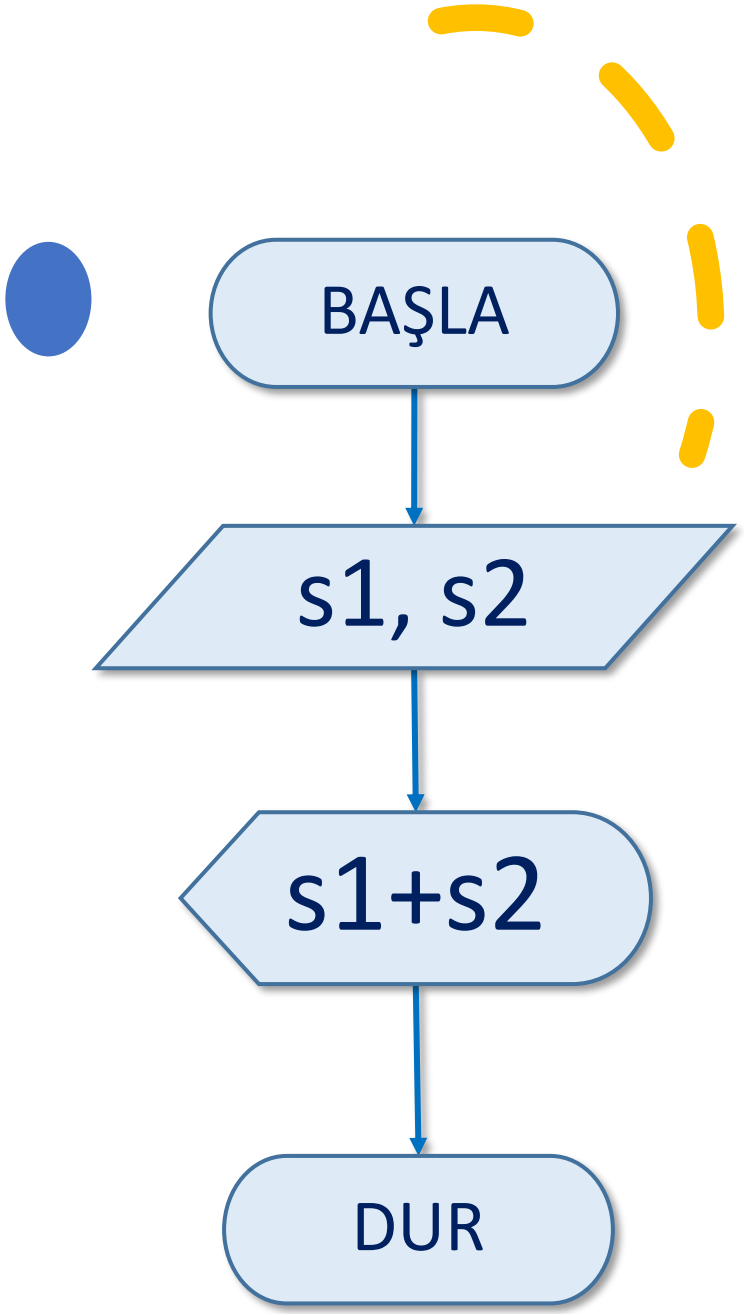
- Klavyeden girilen iki sayıyı toplayıp sonucunu ekrana yazdıran yazan programın algoritmasını yazınız ve akış diyagramını çiziniz.

1. BAŞLA
2. OKU s1, s2
3. YAZ s1+s2
4. DUR

ÖRNEK-3

- Klavyeden girilen iki sayıyı toplayıp sonucunu ekrana yazdıran yazan programın algoritmasını yazınız ve akış diyagramını çiziniz.

1. BAŞLA
2. OKU s1, s2
3. YAZ s1+s2
4. DUR



ÖRNEK-4

- Klavyeden girilen x ve y değerlerine göre $x^2 + y$ denkleminin sonucunu hesaplayıp ekrana yazdıran yazan programın algoritmasını yazınız ve akış diyagramını çiziniz.

ÖRNEK-4

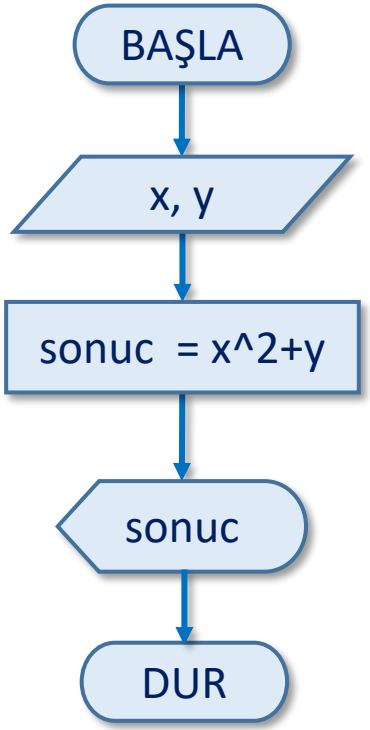
- Klavyeden girilen x ve y değerlerine göre $x^2 + y$ denkleminin sonucunu hesaplayıp ekrana yazdıran yazan programın algoritmasını yazınız ve akış diyagramını çiziniz.

1. BAŞLA
2. OKU x, y
3. sonuc = x^2 + y
4. YAZ sonuc
5. DUR

ÖRNEK-4

- Klavyeden girilen x ve y değerlerine göre $x^2 + y$ denkleminin sonucunu hesaplayıp ekrana yazdıran yazan programın algoritmasını yazınız ve akış diyagramını çiziniz.

1. BAŞLA
2. OKU x, y
3. sonuc = x^2 + y
4. YAZ sonuc
5. DUR



ÖRNEK-5

- Klavyeden girilen x değerlerinin toplamı 100'den büyük oluncaya kadar kullanıcıdan değer alan programın algoritmasını yazınız ve akış diyagramını çiziniz.

ÖRNEK-5

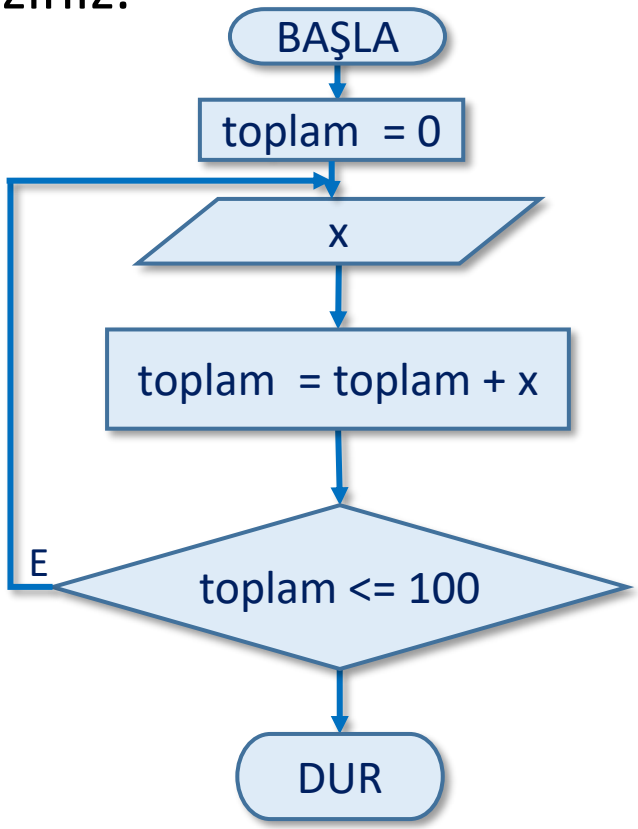
- Klavyeden girilen x değerlerinin toplamı 100'den büyük oluncaya kadar kullanıcıdan değer alan programın algoritmasını yazınız ve akış diyagramını çiziniz.

1. BAŞLA
2. toplam = 0
3. OKU x
4. toplam = toplam + x
5. EĞER toplam <= 100 GİT 3
6. DUR

ÖRNEK-5

- Klavyeden girilen x değerlerinin toplamı 100'den büyük oluncaya kadar kullanıcıdan değer alan programın algoritmasını yazınız ve akış diyagramını çiziniz.

1. BAŞLA
2. toplam = 0
3. OKU x
4. toplam = toplam + x
5. EĞER toplam <= 100 GİT 3
6. DUR



ÖRNEK-6

- Klavyeden girilen sayının faktöriyelini hesaplayan programın algoritmasını yazınız ve akış diyagramını çiziniz.

ÖRNEK-6

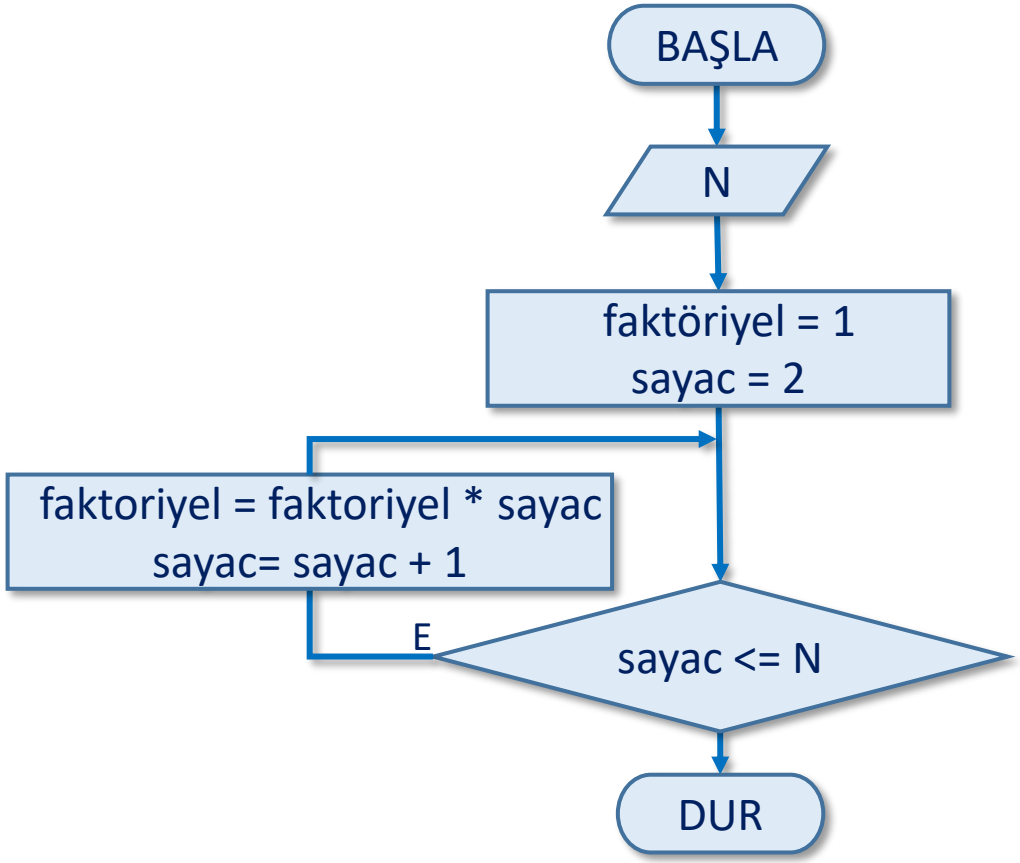
- Klavyeden girilen sayının faktöriyelini hesaplayan programın algoritmasını yazınız ve akış diyagramını çiziniz.
  - **1. ÇÖZÜM:**

1. BAŞLA
2. OKU N
3. faktoriyel = 1
4. sayac = 2
5. EĞER (sayac <= N)
   1. faktoriyel = faktoriyel * sayac
   2. sayac = sayac +1
6. YAZ faktoriyel
7. DUR

ÖRNEK-6

- Klavyeden girilen sayının faktöriyelini hesaplayan programın algoritmasını yazınız ve akış diyagramını çiziniz.
  - **1. ÇÖZÜM:**

1. BAŞLA
2. OKU N
3. faktoriyel = 1
4. sayac = 2
5. EĞER (sayac <= N)
   1. faktoriyel = faktoriyel * sayac
   2. sayac = sayac +1
6. YAZ faktoriyel
7. DUR



ÖRNEK-6

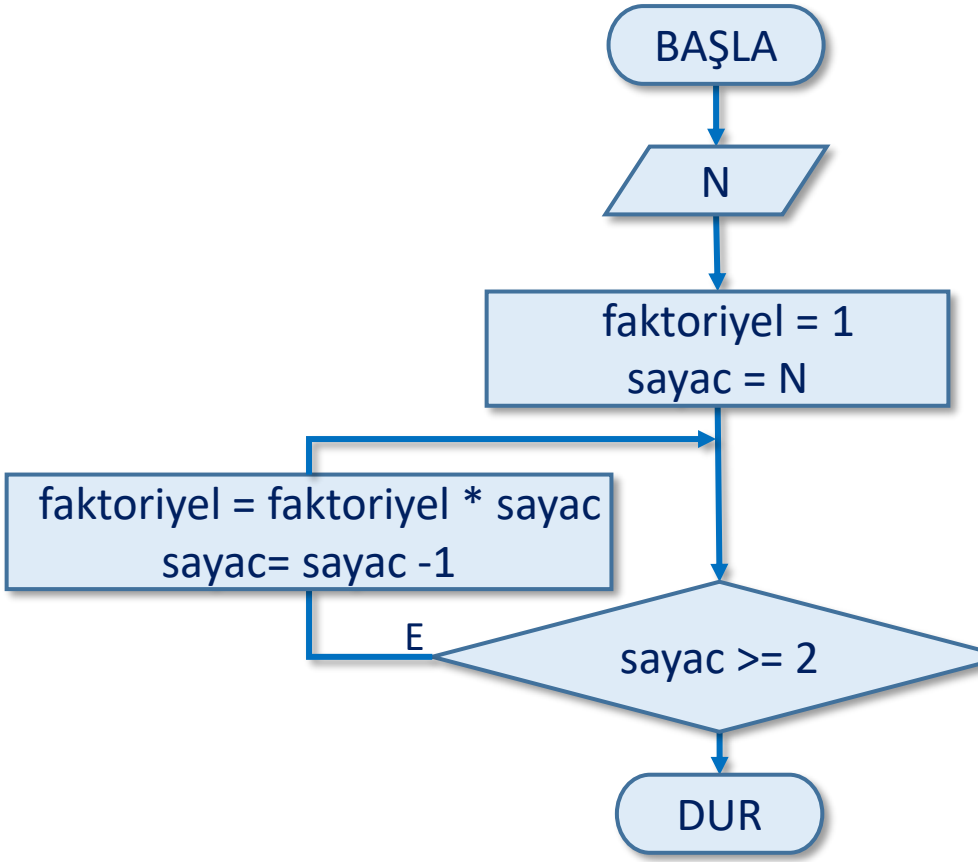
- Klavyeden girilen sayının faktöriyelini hesaplayan programın algoritmasını yazınız ve akış diyagramını çiziniz.
  - **2. ÇÖZÜM:**

1. BAŞLA
2. OKU N
3. faktoriyel = 1
4. sayac = N
5. EĞER (sayac >=2)
   1. faktoriyel = faktoriyel * sayac
   2. sayac = sayac -1
6. YAZ faktoriyel
7. DUR

ÖRNEK-6

- Klavyeden girilen sayının faktöriyelini hesaplayan programın algoritmasını yazınız ve akış diyagramını çiziniz.
  - **2. ÇÖZÜM:**

1. BAŞLA
2. OKU N
3. faktoriyel = 1
4. sayac = N
5. EĞER (sayac >=2)
   1. faktoriyel = faktoriyel * sayac
   2. sayac = sayac -1
6. YAZ faktoriyel
7. DUR



ÖRNEK-6

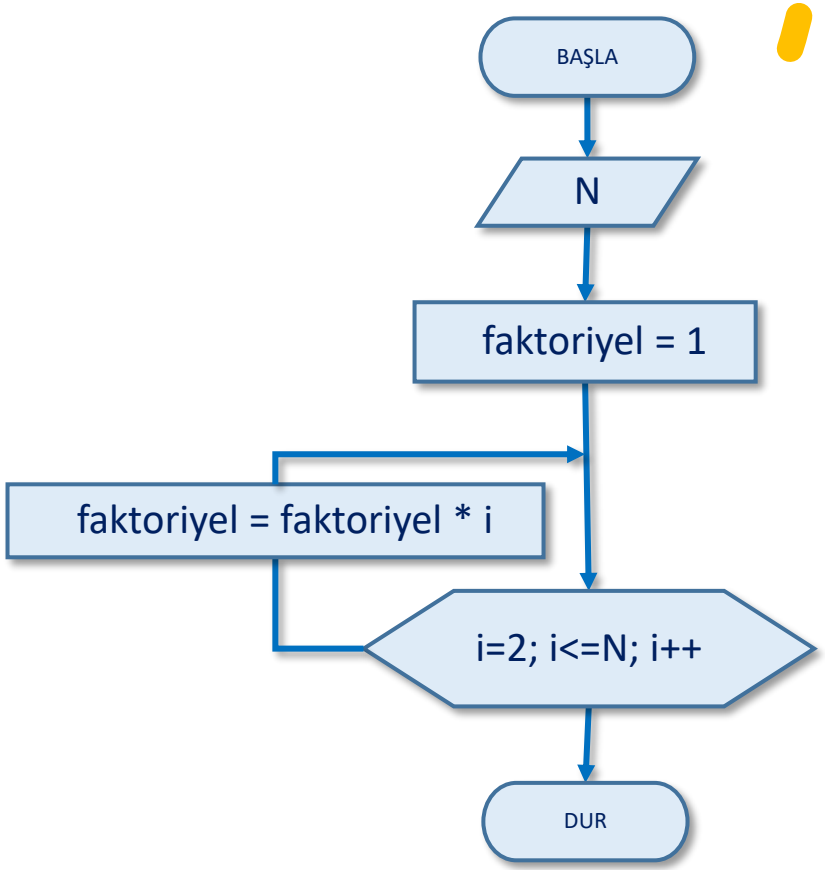
- Klavyeden girilen sayının faktöriyelini hesaplayan programın algoritmasını yazınız ve akış diyagramını çiziniz.
  - **3. ÇÖZÜM:** ⇒ for döngüsü

1. BAŞLA
2. OKU N
3. faktoriyel = 1
4. FOR (i=2; i<=N; i++)
    1. faktoriyel = faktoriyel * i
5. YAZ faktoriyel
6. DUR

ÖRNEK-6

- Klavyeden girilen sayının faktöriyelini hesaplayan programın algoritmasını yazınız ve akış diyagramını çiziniz.
  - **3. ÇÖZÜM:** ⇒ for döngüsü

1. BAŞLA
2. OKU N
3. faktoriyel = 1
4. FOR (i=2; i<=N; i++)
   1. faktoriyel = faktoriyel * i
5. YAZ faktoriyel
6. DUR



ÖRNEK-6

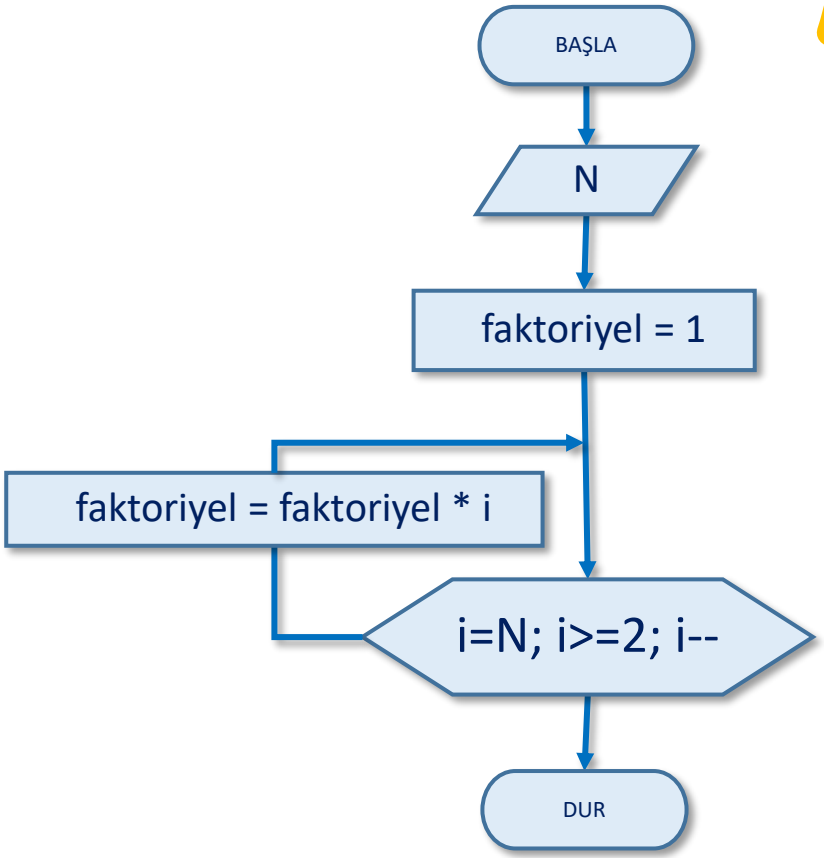
- Klavyeden girilen sayının faktöriyelini hesaplayan programın algoritmasını yazınız ve akış diyagramını çiziniz.
  - **4. ÇÖZÜM:** ⇒ for döngüsü

1. BAŞLA
2. OKU N
3. faktoriyel = 1
4. FOR (i=N; i>=2; i--)
    1. faktoriyel = faktoriyel * i
5. YAZ faktoriyel
6. DUR

ÖRNEK-6

- Klavyeden girilen sayının faktöriyelini hesaplayan programın algoritmasını yazınız ve akış diyagramını çiziniz.
  - **4. ÇÖZÜM:** ⇒ for döngüsü

1. BAŞLA
2. OKU N
3. faktoriyel = 1
4. FOR (i=N; i>=2; i--)
   1. faktoriyel = faktoriyel * i
5. YAZ faktoriyel
6. DUR



Kaynaklar

- C. Öz, C.Bayılmış, G.Çit «Ders Notları»
- Deitel, C++ How To Program, Prentice Hall
- Prof. Dr. Cemil ÖZ, Programlamaya Giriş Ders Notları
- Deitel, P., Deitel, H., Java How To Program, Prentice Hall, Ninth Edition, 2012.
- D.Kılınç «Ders Notları»
- https://sites.google.com/site/modernyazilimgelistirme/uml/uml-diyagramlari/use-case-diagramlari
- Microsoft Visio Professional 2019

- Resimler:
- https://www.animatedimages.org/img-animated-tea-and-teapot-image-0038-164972.htm