Temel Bilgi Teknolojileri I ders notları.

input Fonksiyonu

Kullanıcıdan input almak için `input` fonksiyonu kullanılır.

```
a = input('a = ')
```

'a = ' yerine tırnak içine herhangi bir şey yazılabilir ve kullanıcıya bilgi verilebilir.

ÖRNEK: Kullanıcıdan istenilen üç adet sayının ortalmasını bulan MATLAB programı

```
a = input('a = ');
b = input('b = ');
c = input('c = ');
ort = (a+b+c)/3;
disp(ort);
```

fprintf fonksiyonu

fprintf fonksiyonu bir değişkenin değerini yazdırmak için % kullanır

```
a = 5;
fprintf('a = %d \n', a);
```

`\n` = new line yani yeni satır
`\t` = tab yani 4 boşluk

```
a = 5;
b = 7;
fprintf('a = %d \t b = %d \n', a, b);
```

`%d` tam sayı için kullanılırken `%f` (float) ondalıklı sayı için (6 ondalığa kadar yazdırır) kullanılır.

```
a = 5;
b = 22/7;
fprintf('a = %d \t b = %f \n', a, b);
```

`%.2f` gibi 2 yerine yazılarak kaç basamak ondalık yazılacağı belirtilebilir.
String yazdırmak için `%s` kullanılır. Char için de `%c` kullanılır.

```
a = 5;
b = 22/7;
c = "OGU";
d = 'F';
fprintf('a = %d \t b = %f \nc = %s, d = %c\n', a, b, c, d);
```

`%g` ise sayıları yazdırmak için kullanılır ve f ile d arasında kararı kendi verir. 5 ondalık yazdırır.

ÖRNEK: Kullanıcıdan istenilen 4 adet sayı için aşağıdaki x değerini hesaplayan MATLAB programı

$$x = |a^2 - sin(2bc) + \sqrt[12]{d^5}|$$

```
a = input('a = ');
b = input('b = ');
c = input('c = ');
d = input('d = ');
x = abs(a^2 - sin(2*b*c) + (d^5)^(1/12));
disp(x);
```

ÖRNEK: Kullanıcıdan istenilen yarıçap değeri için bir dairenin çevresini ve alanını bulan program yazınız. ($Çevre = 2\pi r$, $Alan = \pi r^2$)

```
r = input('Yarıçap değeri = ');
cevre = 2*pi-r;
alan = pi*r^2;
fprintf('Cevre = %f \nAlan = %f\n', cevre, alan);
```

Diziler

Diziler tanımlanırken [] kullanılır.
`A = [1,2,3,4,5]` bu şekilde arasına `,` koyarsanız satır vektörü olur. Eğer `B = [1;2;3;4;5]` şeklinde `;` koyarak kullanırsanız sütun vektörü oluşur.

```
A =

     1     2     3     4     5


B =

     1
     2
     3
     4
     5
```

Dizinin belli bir elemanını almak için kaçıncı elemanı olduğunu parantez içinde yazın. Örn `A(2)` demek 2. elemanı demek. Aynı zamanda `A(2) = x` (x herhangi bir değer) diyerek o elemanı değiştirebilirsiniz. Matrislerde eleman alırken `M(satır,sütun)` şeklinde alınabilir. Ardışık sayı dizisi oluşturmak şu şekilde yapılabilir `B = 1:200`, B 1den 200e kadar olan sayılar dizisi. Aynı zamanda `B = 1:5:200` diyerek 5er 5er artır diyerek kullanılabilir. `10:-1:1` gibi de kullanılabilir, `1:0.1:10` gibi de.

- Vektörü ters çevirmek için `fliplr` fonksiyonu kullanılabilir, `fliplr(A)` gibi.
- `linspace` fonksiyonu ile x den y ye a elemanlı dizi oluşturabilirsiniz. Örneğin `linspace(1,10,2)` çıktısı `1 10` olur. Elemanlar doğrusal artan ayarlanır.
- `zeros` fonksiyonu ile 0 matrisi oluşturulabilir. `zeros(satır,sütun)` gibi, aynı şekilde `ones` ile birler matrisi oluşturulabilir.
- `rand` fonksiyonu kullanılarak elemanları rastgele 0 ile 1 arasında olan bir matris oluşturulabilir, `rand(satır,sütun)` şeklinde. \
- `randi` fonksiyonu belirtilen sayılar arasında, istenilen satır ve sütün sayısıyla, istenilen sayıda matris verir.

1 - 4 arası rastgele sayı.

```
>> randi(4)

ans =

     2
```

1-4 arası sayılardan oluşan 2×3 lük matris

```
>> randi(4,2,3)

ans =

     3     4     4
     4     1     4
```

1-2 arası sayılardan oluşan 4×2 lik 3 tane matris

```
>> randi(2,4,2,3)
```

```
ans(:,:,1) =

     1     2
     2     2
     1     2
     1     2


ans(:,:,2) =

     1     2
     2     2
     2     1
     2     2


ans(:,:,3) =

     1     1
     2     1
     1     2
     1     2
```

2-4 arası sayı

```
>> randi([2,4])

ans =

     2
```

- `round` fonksiyonu yuvarlamaya yarar. `round(2.1)` çıktısı 2 dir. 2. parametre olarak kaç ondalık olacağı belirtilebilir. `rand(22/7,2)` nin çıktısı 3.1400 dır.
- `length` fonksiyonu uzunluk almak için kullanılır.
- `size` fonksiyonu boyut için kullanılır. Çıktı satır sütün şeklindedir.
- `sum` fonksiyonu topla demek. Dizinin elemanları toplamını verir.
- `min` ve `max` fonksiyonları adı üstünde min ve max elemanları döner.
- `:` işareti hepsi demek.

Kümeler (Hücre dizileri)

{} kullanılarak oluşturulur. Elemanları her şey olabilir.

```
s = {1,2,3,'a'}
s(1)
% 1. elemanı olan alt küme
s{1}
% 1. eleman
```

Aritmetik ve Mantıksal Operatörler

`=` işareti atama için kullanılırken `==` "eşit mi" gibi bir anlama gelir.
6, 6ya eşit mi?

```
>>> 6 == 6
ans =
        logical
        1
```

6, 5e eşit mi?

```
>>> 6 == 5
ans =
        logical
        0
```

2, 3e eşit değil mi?

```
>>> 2 ~= 3
ans =
        logical
        1
```

Operatör	Anlam
==	Eşittir
~=	Eşit değildir
<	Küçüktür
>	Büyüktür
<=	Küçük eşittir
>=	Büyük eşittir
&& (`and()`)	Ve

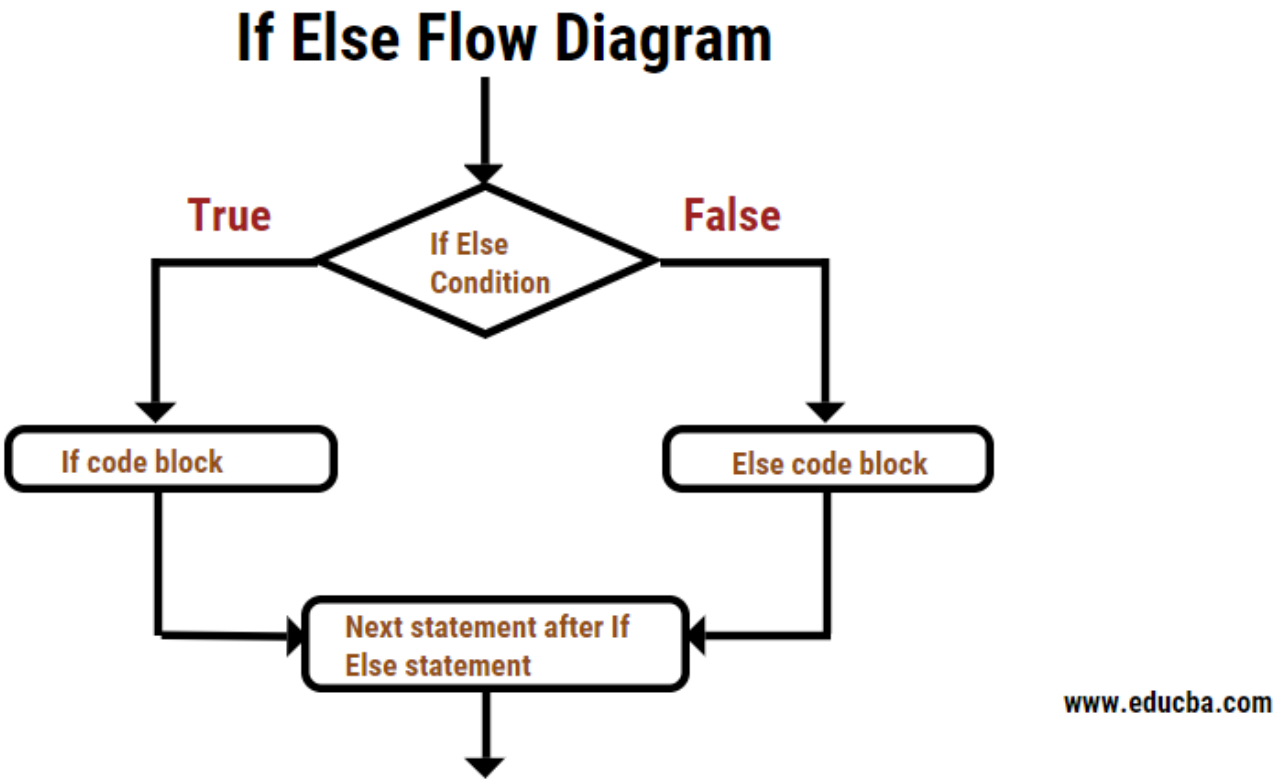
Operatör	Anlam
|| (`or()`)	Veya
xor()	Koşullu veya (ya da)

Not : && ve || işaretleri sadece skalerler için kullanılır. Eğer vektörler (diziler) için kullanmak isterseniz & ve | kullanmanız gerekir. Aynı zamanda `and()` ve `or()` fonksiyonlarını da kullanabilirsiniz.
String karşılaştırmak için unutmamak gereken şey "" ın skaler gibi '' vektör gibi işler.

```
'abc' == 'def'
0 0 0

"abc" == "def"
0
```

Koşullu Durumlar




```
if (kosul)
        islem1
else
        islem2
end
```

`kosul` doğru ise islem1 çalışır. `kosul` yanlış ise islem2 çalışır.

ÖRNEK: Kullanıcıdan istenilen sayının karesini ve sayı sıfırdan büyükse karekökünü bulan program.

```
sayi = input('Sayı = ');
kare = sayi^2;
fprintf('Sayı = %d\nKaresi = %d', sayi, kare);
if (sayi > 0)
        kok = sqrt(sayi);
        fprintf('Karekökü = %d\n', kok);
end
fprintf('\n');
```

ÖRNEK: Kullanıcıdan istenilen sayını eğer 0dan küçükse karesini, değilse karekökünü bulan program.

```
sayi = input('Sayı = ');
if (sayi < 0)
        kare = sayi^2;
        fprintf('%d sayısının karesi %d', sayi, kare);
else
        kok = sqrt(sayi);
        fprintf('%d sayısının karekökü %d', sayi, kok);
end
fprintf('\n');
```

ÖRNEK: Kullanıcı tarafından girilen bir sayı $0 \leq x < 9$ biçminde ise $\sqrt{x} + ln(x)$ değerini hesaplayan, sayı bu aralıkta değil ise ekrana "Sayı yanlış aralıkta" mesajını veren bir program yazınız.

```
sayi = input('Sayı = ');
if (sayi >= 0 && sayi < 9)
        islem = sqrt(sayi) + log(sayi)
else
        fprintf('Sayı yanlış aralıkta\n');
end
```

NOT: Hata vermek için `fprintf()` yerine `error()` kullanarak hata verilebilir.

```
sayi = input('Sayı = ');
if (sayi >= 0 && sayi < 9)
        islem = sqrt(sayi) + log(sayi)
else
        error('Sayı yanlış aralıkta\n');
end
```

ÖRNEK: Bir araç satış firmasında çalışan personel için maaş hesabı yapılmak istenmektedir. Buna göre aylık satış adedi 5'ten az ise 5500₺ sabit maaş ve satılan her

araç için 450₺ prim verilmektedir. Satılan araç adedi 5 veya daha fazla ise 6500₺ sabit maaş ve satılan ilk 5 araç için araç başına 475₺ prim, 5'ten sonraki her araç için 575₺ prim verilmektedir. Buna göre satış adedi verilen personelin maaşını bulan program yapınız.

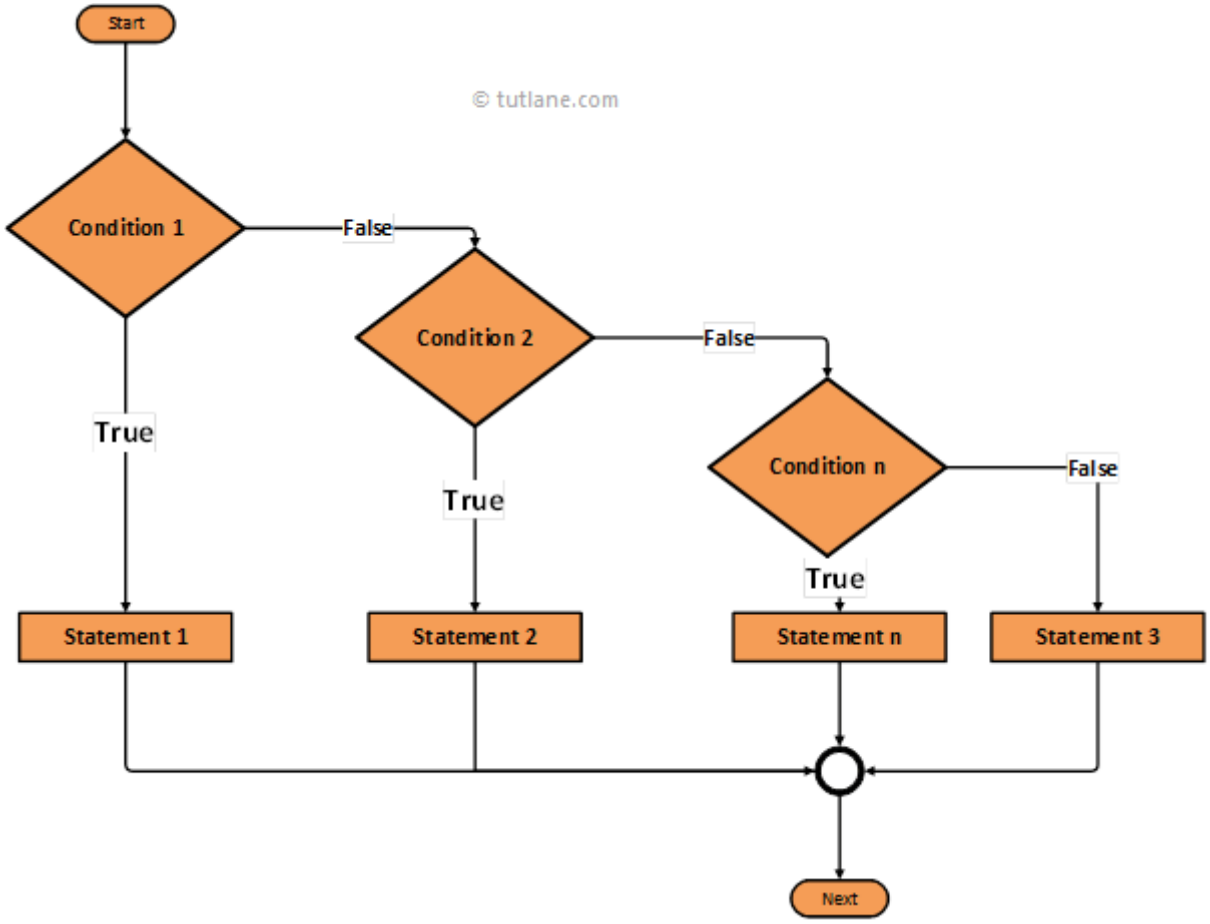
```
satis = input('Satış adedi = ');

if (satis < 0 )
        error('Satış adedi 0dan küçük olamaz!');
end

if (satis < 5)
        maas = 5500 + satis*450;
else
        maas = 6500 + 5*450 + (satis-5)*575;
end
fprintf('Maaş = %d', maas);
```

ÖRNEK: Kullanıcıdan alınan 3 sayının en küçüğünü bulan program.

```
sayi1 = input('1. sayı = ');
sayi2 = input('2. sayı = ');
sayi3 = input('3. sayı = ');
if (sayi1 <= sayi2)
        if (sayi1 <= sayi3)
                ek = sayi1;
        else
                ek = sayi2;
        end
else
        if (sayi2 <= sayi3)
                ek = sayi2;
        else
                ek = sayi3;
        end
end
fprintf('En küçük %d', ek);
```



Aynı örneği `elseif` kullanarak çözelim.

```
sayi1 = input('1. sayı = ');
sayi2 = input('2. sayı = ');
sayi3 = input('3. sayı = ');
if (sayi1 <= sayi2 && sayi1 <= sayi3)
        ek = sayi1;
elseif (sayi2 <= sayi1 && sayi2 <= sayi3)
        ek = sayi2;
else
        ek = sayi3;
end
fpritf('En küçük %d', ek);
```

Başka bir yol.

```
sayi1 = input('1. sayı = ');
sayi2 = input('2. sayı = ');
sayi3 = input('3. sayı = ');
fprintf('En küçük %d', min([sayi1,sayi2,sayi3]))
```

ÖRNEK: Gelir vergisinin aşağıdaki kurallara göre alındığını varsayalım.
$gelir \leq 150.000 \rightarrow$ vergi oranı %25
$gelir \leq 300.000 \rightarrow$ vergi oranı %30
$gelir \leq 600.000 \rightarrow$ vergi oranı %35

$gelir \leq 1.200.000 \rightarrow$ vergi oranı %40
$gelir > 1.200.000 \rightarrow$ vergi oranı %50
Buna göre yıllık geliri verilen firmanın ödemesi gereken vergiyi hesaplayan matlab programı yazınız. (gelir=2.000.000?)

```
g = input('Yıllık gelir = ');

if (g <= 0)
        v = 0;
        fprintf('İşlem: Vergi kontrolü');
elseif (g <= 150)
        v = g*0.25;
elseif (g <= 300)
        v = 150*0.25 + (g-150)*0.30;
elseif (g <= 600)
        v = 150*0.25 + 150*0.30 + (g-300)*0.35;
elseif (g <= 1200)
        v = 150*0.25 * 150*0.30 * 300*0.35 + (g-600)*0.40;
else (g > 1200)
        v = 150*0.25 * 150*0.30 * 300*0.35 + 600*0.40 + (g-600)*0.50;
end

fprintf('Vergi = %g\n', v);
```

Döngüler

for Döngüsü

```
for i=baslangic:artis_m:bitis
        deyimler
end
```

baslangic değerinden başlayarak i yi artis_m kadar artırarak bitis e kadar gider ve her bir artırmada içindeki deyimler işlenir.

while Döngüsü

```
while (kosul)
        deyimler
end
```

`kosul` doğru olduğu sürece deyimler çalışır.

Örnek: 1den kullanıcının girdiği sayıya kadar olan sayıların toplamını hesaplayan program.
for döngüsü kullanarak:

```
n = input('n = ');
toplam = 0;
for i=1:1:n
        toplam = toplam + i;
end
fprintf('Toplam = %d\n', toplam);
```

while döngüsü kullanarak:

```
n = input('n = ');
toplam = 0;
i = 1;
while (i <= n )
    toplam = toplam + i;
    i = i + 1;
end
fprintf('Toplam = %d\n', toplam);
```

ÖRNEK: $\sum_{n=1}^{100} \frac{sin^2(n)}{n^2}$ değerini hesaplayan program.

```
toplam = 0;
for i=1:1:100
    toplam = toplam + ((sin(i)^2)/i^2);
end
fprintf('Toplam = %d\n', toplam);
```

ÖRNEK: Kullanıcının girdiği bir n değeri için $n!$ değerini hesaplayan Matlab programını yazınız.

```
n = input('n = ');
if (n<0)
        error('n sayısı sıfırdan küçük olamaz.');
end

fac = 1;
for i=1:n
    fac = fac * i;
end
fprintf('%d! = %d\n', n, fac);
```

ÖRNEK: 1den kullanıcının girdiği bir değere kadar tek sayıların toplamını bulan program.

```
n = input('n = ');
t = 0;
for i=1:2:n
    t = t + i;
```

```
end
fprintf('Toplam = %d\n', t);
```

ÖRNEK: 1den kullanıcının girdiği bir değere kadar 3e veya 4e bölünmeyen sayıların toplamını bulan program yazınız.

```
n = input('n = ');
toplam = 0;
for i=1:n
    if ( (mod(i,3) ~= 0) && (mod(i,4) ~= 0) )
        toplam = toplam + i;
    end
end
fprintf('Toplam = %d\n', toplam);
```

ÖRNEK: 10₺nin yıllık %30 faiz ile 50₺yi geçtiği veya eşit olduğu ilk yılı ve paranın kaç ₺ olduğunu bulan bir Matlab programı yazınız.

```
para = 10;
yil = 0;

while (para < 50)
    yil = yil + 1;
    para = para + (para*0.3);
end

fprintf('%d yıl %g TL\n', yil, round(para,2));
```

ÖRNEK: Çarpım tablosu.

```
for i=1:10
    for j=1:10
        fprintf('%5d', i*j);
    end
    fprintf('\n');
end
```

ÖRNEK: $\frac{1}{2!} + \frac{1}{3!} + \frac{1}{4!} + \ldots + \frac{1}{15!}$ işlemini hesaplayan programı yazınız.

```
toplam = 0;

for i=2:15
    fac = 1;
    for j=1:i
        fac = fac * j;
    end
    toplam = toplam + 1/fac;
```

```
end

fprintf('%g', toplam);
```

break ve continue

Döngünün içerisinde kullanılırlar.

break

Break kırmak demektir ve kullanılan yerde döngüyü kırıp döngüden çıkar.

```
for i=1:10
    break;
    fprintf('%d\n', i);
end
fprintf('Döngü bitti\n');
```

Yukarıdaki kodda döngüye girdiği gibi `break` okuyup döngüyü bitirir ve çıktı sadece "Döngü bitti" olur.

Örnek: Kullanıcı negatif bir sayı girinceye kadar kullanıcının girdiği sayıları toplayan, negatif sayı girilince uygulamayı sonlandırıp sayıların toplamını söyleyen bir MATLAB programı yazınız.

```
x = 0;

while 1
    x = input('Pozitif bir sayı giriniz = ');
    if x < 0
        break
    end
    t = t + x;
end

fprintf('Girdiğiniz sayıların toplamı = %d\n', t);
```

continue

Devam et anlamına gelir. Döngü içerisinde kullanıldığı yerde döngünün o aşamasını atlar.

```
for i=1:10
    continue;
    fprintf('%d', i);
end
fprintf('Döngü bitti\n');
```

Her seferinde continue diyip sonraki aşamaya geçildiği için i yi hiç yazdırmadan döngüyü bitirir.

Örnek: Kullanıcıdan 10 adet sayı isteyen ancak sadece pozitif olanların toplamını bulan MATLAB programı yazınız.

```
t = 0;

for i=1:10
    x = input('Pozitif bir sayı giriniz = ');
    if x < 0
        continue
    end
    t = t + x;
end

fprintf('Girdiğiniz pozitif sayıların toplamı = %d\n', t);
```

Örnek: $x = 10$ ve $y = 7$ olarak verilsin. Bu sayıların toplamı 52347den küçük kaldığı müddetçe x sayısı ikiye katlanarak bu iki sayı toplanmaya devam edilmektedir. Döngü bittiğinde x sayısı kaç kez ikiye katlanmış olur?

```
x = 10;
y = 7;
i = 0;
t = x + y;

while t < 52347
    x = x * 2;
    t = x + y;
    i = i + 1;
end

fprintf('x %d kere 2ye katlandı\n', i);
```

(`i` değişkeni kaç kere ikiye katlandığını, `t` değişkeni toplamı ifade ediyor)

`num2str()` ve `str2double()` fonksiyonları

`num2str()` fonksiyonu verilen sayıyı string e çevirir. `str2double()` fonksiyonu da stringi double a çevirir.

ÖRNEK: Kullanıcının girdiği sayının rakamları toplamı.

```
n = input('n = ');
s = num2str(n);
t = 0;
```

```
for i=1:length(s)
    t = t + str2double(s(i))
end

fprintf('Basamaklar toplamı = %d\n', t);
```

NOT: Listenin elemanları toplama

Yol I

```
liste = [12,13,14,15,16];
```

Yol II

```
toplam = sum(liste);
fprintf('Toplam = %d \n', toplam);

%% Yol II
toplam = 0;
for i=1:numel(liste)
    toplam = toplam + liste(i);
end
fprintf('Toplam = %d \n', toplam);
```

Yol III

```
toplam = 0;
for i=liste
    toplam = toplam + i;
end
fprintf('Toplam = %d \n', toplam);
```

ÖRNEK: 3 basamaklı sayılar içerisinde basamak değerlerinin 3. kuvvetlerinin toplamı kendisine eşit olan sayılara Armstrong sayısı denir. Buna göre tüm Armstrong sayılarının toplamı kaçtır.

```
l = [];

for i=100:999
    s = num2str(i);
    uctop = 0;
    for j=1:length(s)
        uctop = uctop + (str2double(s(j))^3);
    end
    if uctop == i
        l = [l, i];
    end
end
```

```
fprintf('Armstrong sayılarının toplamı = %d\n', sum(l));
```

Ara sınava yönelik çalışmalar

Örnek: Kullanıcıdan istenilen satır ve sütun sayısına karşılık adresleri belirtilerek bir matris oluşturun.

```
x = input('Satır sayısı = ');
y = input('Sütun sayısı = ');

M = [];

for i=1:x
    for j=1:y
        fprintf('[%d,%d] = ', i, j);
        M(i,j) = input('');
    end
end

disp(M);
```

Not: `transpose(M)` veya `M'` ile bir matrisin transpozunu bulabilirsiniz.

Not: `isprime()` fonksiyonu ile bir sayının asal olup olmadığı anlaşılabilir.

Örnek: 13×21 lik elemanları asal sayılar olan bir matrisin 3. sütünundaki elemanlar toplamı ile 2. satır elemanlar toplamı farkı

```
x = 13;
y = 21;
k = 2;
M = [];

for i=1:x
    for j=1:y
        while 1
            if isprime(k)
                M(i,j) = k;
                k = k + 1;
                break
            else
                k = k + 1;
            end
        end
    end
end
```

```
sum(M(:,3))-sum(M(2,:))
```

Örnek: Satır ve sütun sayısı kullanıcıdan istenilen ve 0 ile 15 arasında rastgele tamsayı değerler alan bir matris oluşturun. Daha sonra oluşan matristeki 0ların yerlerini ve sayısını bulan bir program yazınız.

```
boyut = input('Boyut ([x,y]) = ');
x = boyut(1);
y = boyut(2);

M = randi([0,15], x, y);

s = 0;

for i=1:x
    for j=1:y
        if M(i,j) == 0
            s = s + 1;
            fprintf('M(%d,%d) değeri 0\n', i, j);
        end
    end
end

fprintf('%d tane 0 var.\n', s);
```

Örnek: `transpose()` fonksiyonu kullanmadan kullanıcının girdiği matrisin transpozunu alınız.

```
M = input('M = \n');

T = [];

[x,y] = size(M);

for i=1:y
    for j=1:x
        T(i,j) = M(j,i);
    end
end

disp(T);
```

Örnek: Kullanıcının girdiği sayının mükemmel olup olmadığını bulan program.

```
x = input('x = ');

bolenler = [];
```

```
for i=1:x-1
    if mod(x,i) == 0
        bolenler = [bolenler, i];
    end
end

if sum(bolenler) == x
    fprintf('Mükemmel sayı\n');
else
    fprintf('Mükkemmel sayı değil\n');
end
```

Örnek: 1den 999a kadar bütün mükemmel sayıların toplamı.

```
m_liste = [];
for x=1:999
    bolenler = [];
    for i=1:x-1
        if mod(x,i) == 0
            bolenler = [bolenler, i];
        end
    end
    if sum(bolenler) == x
        m_liste = [m_liste, x];
    end
end
sum(m_liste)
```

switch-case Yapısı

```
switch durum
        case durum1
                deyim1
        case durum2
                deyim2
        ...
        otherwise
                deyimn
end
```

Örnek: Kullanıcıdan istenilen bir ayın kaç günden oluştuğunu bulan bir program yapınız.

```
n = input('Bir ay giriniz: ');
switch n
    case {1,3,5,7,8,10,12}
        fprintf('31 Gün\n');
```

```
    case {2}
        yil = input('Hangi yıl: ');
        if mod(yil,4) == 0
            if (mod(yil,100) == 0)
                if (mod(yil,400) == 0)
                    fprintf('29 Gün\n');
                else
                    fprintf('28 Gün\n');
                end
            else
                fprintf('29 Gün\n');
            end
        else
            fprintf('28 Gün\n');
        end
    case {4,6,9,11}
        fprintf('30 Gün\n');
    otherwise
        fprintf('Lütfen geçerli bir ay giriniz.\n');
end
```

try-catch Yapısı

```
try
        deyim1
catch
        deyim2
end
```

try-catch yapısında deyim1'i çalıştırmayı dener ve eğer deyim1 hata verirse deyim2 çalıştırılır.

Eğer hatayı bir değişkene atamak isterseniz catch den sonra değişken ismi yazabilirsiniz.

```
try
        deyim1
catch me
        deyim2
end
```

Örnek: Kullanıcıdan istenilen bir dizinin eleman sayısını (length, size, numel fonksiyonlarını kullanmadan) bulan bir program yazınız.

```
A = input('A = ');
i = 1;
while 1
```

```
    try
        A(i);
        i = i + 1;
    catch
        fprintf('Diznin uzunluğu %d\n', i-1);
        break;
    end
end
```

lambda Fonksiyonları

lambda fonksiyonları kısaltılmış fonksiyon tanımlardır. Bir lambda fonksiyonu `fonk_adi = @(gp) fonk_tanimi` şeklinde tanımalanabilir. Örneğin `kare = @(x) x.^2`. Başka bir örnek olarak parametre almayan fonksiyonlar, `sil = @() clc` olarak tanımlanabilir. Bu fonksiyonları kullanırken parantez boş bırakılır (`sil()` şeklinde). Yine bir fonksiyon birden çok parametre alabilir.

Örnek: Aşağıdaki fonksiyonu matlab'a aktarınız.

$$f(x,y,z) = \frac{e^{2x+1} - sin(y)}{2\sqrt{z} + ln(z)}$$

```
f = @(x,y,z) [(exp(2*x+1)-sin(y))/(2*sqrt(z)+log(z))]
```

MATLAB Fonksiyonları

MATLAB de yazılan bilgisayar programları fonksiyonlardır. Fonksiyonlar görsel olarak sonuç verebileceği gibi text olarak da sonuç verebilir.

- Her fonksiyon kendine özel bir isme sahip olmalı. Mümkün olduğunca kısa ve amaca uygun olmalı. İsimlendirme kuralları değişkenler ile aynıdır.
- Bir fonksiyon genellikle üzerinde işlem yapacağı bir parametre alır. Giriş parametresi almayan fonksiyonlara örnek `clc()`. Genellikle kullanıcıya bir tane de çıkış parametresi verir.
  Bir fonksiyon şöyle tanımlanabilir:

```
function [outputArg] = func1(inputArg)
% FUNC1 Girilen değeri verir
    % Girdiğiniz bir değeri aynı şekilde geri verir.
    outputArg = inputArg;
end
```

`function` anahtar kelimesi ile bir fonksiyon tanımlayacağımızı belirtiriz. `[]` içine çıkış parametreleri tanımlanır. `=` den sonra fonksiyon ismi verilir. Unutulmaması gereken şudur ki dosya adı ile fonksiyon adı aynı olmalıdır. Yani `func1` in dosyasının adı

`func1.m` olmalıdır. Bir dosyada yalnızca 1 fonksiyon olabilir. Daha sonra `()` içerisine giriş parametreleri verilir. İlk 2 yorum satırı kullanıcının `help func1` yazdığında verilecek çıkıştır. Daha sonra işlemler yapılır ve çıkış olarak verilecek değer belirlenir. Ve `end` ile bitirilir.

```
function [cp] = fonk_adi(gp)
function [cp] = fonk_adi(gp1,gp2,...,gpn)
function [cp1,cp2,...,cpn] = fonk_adi(gp)
function [cp1,cp2,...,cpn] = fonk_adi(gp1,gp2,...,gpn)
function [cp] = fonk_adi()
function fonk_adi(gp)
```

Sırasıyla tek giriş parametreli ve çıkış parametreli fonksiyon; tek çıkış, çok giriş parametreli fonk; çok çıkış, tek giriş parametreli fonksiyon; çok giriş ve çıkış parametreli fonk; tek çıkış parametreli ve giriş parametresiz fonksiyon; çıkış parametresiz fonksiyon.

Örnek: İki nokta arası uzaklığı hesaplayan bir MATLAB foknsiyonu yazınız.

```
function sonuc = distance(nokta1, nokta2)
% DİSTANCE İki nokta arası uzaklığı bulur
    % Dik koordinat sisteminde iki nokta arası uzaklığı hesaplar. Giriş
olarak iki liste alır.
    % Kullanım : distance(nokta1, nokta2)    (nokta1 = [x1,y1], nokta2 =
[x2,y2])

    x1 = nokta1(1);
    x2 = nokta2(1);
    y1 = nokta1(2);
    y2 = nokta2(2);
    sonuc = sqrt((x2-x1)^2 + (y2-y1)^2);
end
```

Örnek: $f(x,y) = x^2y + \sqrt{xy} + \frac{ln(x+1)}{logy}$

```
function sonuc = f(x,y)
    sonuc = (x^2)*y + sqrt(x*y) + log(x+1)/log10(y);
end
```

Built-in bir fonksiyonla aynı isimde bir fonksiyon yazarsak çalıştırdığımızda çalışacak olan bizim tanımladığımızdır. Önem sırası:

1. Oturum değişkeni
2. Kullanıcı fonksiyonu
3. Built-in fonksiyon

Alt fonksiyonlar

Ana fonksiyonumuzu parçalara ayırmak için aynı dosya içine yardımcı alt fonksiyonlar oluşturulabilir.

```
function sonuc = anaFonk (x)
        sonuc = altFonk(x) + 1;
end

function sonuc = altFonk(a)
        sonuc = a^3 - 10;
end
```

Bu şuna eşittir:

```
function sonuc = fonk (x)
        sonuc = x^3 - 10 + 1;
end
```

Burada unutulmaması gereken şey `altFonk` un CLI (Command Line Interface)'da çağırılamamasıdır.

Örnek: `isPrime()` kullanmadan bir sayının asal sayı olup olmadığnı bulan bir fonksiyon yazınız.

```
function sonuc = asalMi (n)
    pBolen = bolenler(n);
    if length(pBolen) == 2
        sonuc = true;
    else
        sonuc = false;
    end
end

function liste = bolenler (n)
    liste = [];
    for i=1:n
        if mod(n,i) == 0
            liste = [liste, i];
        end
    end
end
```

Yazılan bir fonksiyon başka fonksiyonların içinde kullanılabilir.

Örnek: n'den küçük asal sayıların listesini bulan fonsiyon.

```
function liste = asallar(n)
    liste = [];
    for i=2:n-1
        if asalMi(i)
            liste = [liste, i];
        end
    end
end
```

Değişkenleri Yer Değiştirme

`x = 3` ve `y = 2` olsun. Bu iki değişkeni yer değiştirmek için geçici bir değişkene ihtiyaç var.

```
>> x = 3
x = 3
>> y = 2
y = 2
>> g = x
g = 3
>> x = y
x = 2
>> y = g
y = 3
```

Bu şekilde `g` geçici değişkenini kullanarak `x` ve `y` değerlerini değiştirdik. Şimdi bu işlem için bir fonksiyon yazalım.

```
function [x,y] = degistir(x,y)
    gecici = y;
    y = x;
    x = gecici;
end
```

Ve kullanalım.

```
>> x = 3
x = 3
>> y = 2
y = 2
>> [x,y] = degistir(x,y)
x = 2
y = 3
```

Sıralama Algoritmaları

Amacımız bir dizinin (vektörün) elemanlarını küçükten büyüğe doğru sıralamak.

Seçmeli Sıralama (Selection Sort)

```
function [liste] = secmeliSirala(liste)
    lenListe = length(liste);
    for i=1:lenListe
        min = i;
        for j=i+1:lenListe
            if ( liste(j) < liste(min) )
                min = j;
            end
        end
        gecici = liste(i);
        liste(i) = liste(min);
        liste(min)= gecici;
    end
end
```

Kabarcık Sıralama (Bubble Sort)

```
function [liste] = kabarcikSirala(liste)
    len = length(liste);
    for i=1:len
        for j=len:-1:i+1
            if ( liste(j) < liste(j-1) )
                gecici = liste(j);
                liste(j) = liste(j-1);
                liste(j-1) = gecici;
            end
        end
    end
end
```

Fonksiyonlarda

`nargin`

`nargin` size fonksiyona verilen parametre sayısını verir.

```
function sonuc = fonk3(x,y)
    if ( nargin == 2)
        sonuc = 5;
    elseif ( nargin == 1)
```

```
        sonuc = 6;
    else
        sonuc = 7;
    end
end
```

```
>> fonk3()
ans = 7
>> fonk3(1)
ans = 6
>> fonk3(1,2)
ans = 5
```

nargout

`nargout` size yapılan çıkış parametresi sayısı verir.

```
function sonuc = fonk3(x,y)
    if ( nargin == 2)
        sonuc = 5;
    elseif ( nargin == 1)
        sonuc = 6;
    else
        sonuc = 7;
    end
    sonuc = nargout;
end
```

```
>> fonk3()
ans = 0
>> x = fonk3()
x = 1
```

varargin

`varargin` size giriş yapılan değişkenleri verir.

```
function sonuc = fonk3(varargin)
    if ( nargin == 2)
        sonuc = 5;
    elseif ( nargin == 1)
        sonuc = 6;
    else
        sonuc = 7;
    end
```

```
    sonuc = varargin;
end
```

```
>> A = fonk3(1,2)
A = ...
>> A{1}
ans = 1
>> A{2}
ans = 2
```

varargout

`varargout` size çıkış yapılan tüm değişkenleri verir.

```
function [varargout] = fonk3(varargin)
    if ( nargin == 2)
        sonuc = 5;
    elseif ( nargin == 1)
        sonuc = 6;
    else
        sonuc = 7;
    end
    varargout{1} = sonuc
end
```

```
>> x = fonk3(1)
varargout =
{
  [1,1] = 6
}

x = 6
```

Örnek: Kullanıcının girdiği tüm sayıların toplamını bulan bir fonksiyon yazınız.

```
function sonuc = topla (varargin)
    sonuc = 0;
    for i=1:nargin
        sonuc = sonuc + varargin{i};
    end
end
```

- `isnumeric`: Girilen değerin sayı olup olmadığını kontrol eder. Üstteki fonksiyonda kullanıcının karakter girmesine önlem olarak kullanılabilir.

```
function sonuc = topla (varargin)
    sonuc = 0;
    for i=1:nargin
            if ( isnumeric(varargin{i}) )
                sonuc = sonuc + varargin{i};
        end
    end
end
```

Örnek: Kullanıcının girdiği sayıların toplamını bulan fonksiyon, birden fazla çıktı istiyorsa sırasına göre kuvvetini verecek.

```
function [varargout] = topla2 (varargin)
    sonuc = 0;
    for i=1:nargin
        if ( isnumeric(varargin{i}) )
            sonuc = sonuc + varargin{i};
        end
    end

    for j=1:nargout
        varargout{j} = sonuc^j;
    end
end
```

Örnek: x ve y birbirinden farklı olmak üzere $|x| + |y| \leq 3$ sağlayan x ve y ikililerini ekrana yazan bir program yazınız.

```
function ornek1()
    for x=-3:3
        for y=-3:3
            if ( x ~= y)
                if ( abs(x) + abs(y) <= 3 )
                    fprintf('(%2d, %2d)\n', x, y);
                end
            end
        end
    end
end
```

Örnek: Faktöriyel bulan fonksiyon

```
function sonuc = faktoriyel(n)
    if n < 0
        error('n negatif olamaz');
    end

    sonuc = n;
```

```
    if n == 0
        sonuc = 1;
    else
        sonuc = sonuc * faktoriyel(n-1);
    end

end
```

Örnek: Girilen kelimenin polindrom olup olmadığını bulan fonksiyon.

```
function sonuc = pal(kelime)
    boyut = length(kelime);
    yari = fix(boyut/2);

    sonuc = true;

    for i=0:yari
        if ( kelime(i+1) ~= kelime(boyut-i) )
            sonuc = false;
            break;
        end
    end
end
```

Başka bir yöntem

```
function sonuc = pal(kelime)
    sonuc = all(kelime == fliplr(kelime));
end
```