LAB 2 Çalışma Kağıdı

Aşağıdaki matrisleri Octave'da gösteriniz.

$$A = \begin{bmatrix} -3 & 2 & 0 & 1 \\ 0 & 4 & 1 & -1 \\ 2 & -2 & 1 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 3 & -1 & 2 & 11 \\ 0 & 3 & 7 & -1 \\ -12 & 4 & 6 & 5 \end{bmatrix}, \quad C = \begin{bmatrix} -1 & 4 \\ 0 & 9 \\ 1 & 1 \\ -1 & 2 \end{bmatrix},$$

1. Matris – Skaler Çarpımı

A matisini 5 skaleriyle iki for loop kullanarak çarpın.

```
[m,n]= size(A);
for i=1 : m
      for j = 1 : n
           A(i,j)=A(i,j)*5;
      end
end
```

2. Matris – Matris Toplamı

A ve B matislerini iki for loop kullanarak toplayın. Sonucu yeni bir matriste tutun.

```
clc % command window'u temizlemek icin
D = zeros(m,n); % toplamın sonucunu bu matriste tutatacagiz
for i=1 : m
      for j = 1 : n
           D(i,j)=A(i,j)+B(i,j);
      end
end
```

3. Üst Üçgen Matris

A matrisini ust ucgen matris haline getirin.

(İp ucu bir A matrisi ust ucgen ise her $i > j$ için $A_{ij} = 0$ olur.)

```
clc
for i=1 : m
      for j = 1 : n
           if i > j
              A(i,j)=0;
```

```
            end
        end
end
```

Daha efektif kod:

Dikkat edilirse $i > j$ olan yerler (yani satır indisinin sutun indisinden buyuk oldugu yerler:

$(2,1)$

$(3,1)$ $(3,2)$

$(4,1)$ $(4,2)$ $(4,3)$

$(5,1)$ $(5,2)$ $(5,3)$ $(5,4)$

Yani satır indisi 2'den başlayıp son satira kadar gidecek. Sutun indisi ise (yani saga dogru ne kadar gidecegimiz) hangi satirda oldugumuza bagli. Bu indis her satir icin 1'den basliyor satir indisinden bir once bitiyor.

```
clc
for i=2 : m
      for j = 1 : i-1
          A(i,j)=0;
      end
end
```

4. Diagonal Matris

Girilen bir matrisin diagonal olup olmadiina karar veren program yazin.

(ipucu bir matris diagonal ise hem alt ucgen matris hem de ust ucgen matristir, yani diagonal elemanlari (A_{ii}) dışındakı tüm elemanları 0 dır)

```
  [m,n]= size(A);
   myBool=true; %once diagonal oldugunu kabul ediyoruz
   for i=1:m
     for j=1:n
       if i~=j && A(i,j)~=0 %i, j'den farkli iken A(i,j)'de 0 dan farkli ise
         myBool=false;
         break
       end
     end
   end
```

5. Matrisin Transpozunu Almak

Hatirlarsak bir matrisin transpozunu alirken satırlar sütün oluyordu (yada sutunlar satir). Buradan hareketle A matrisinin transpozunu A'nin satırlarını transpozunda sutun olacak sekilde alınız.

```
[m,n]= size(A);
transA=zeros(n,m); %A'nin transpozunu bu matriste tutacagiz
for i=1:m
    transA(:,i)= A(i,:); %A'in i. satiri transA'nin i. sutunu
end
```

Alternatif olarak A matrisinin transpozunu A'nin sütünları transpozunda satır olacak sekilde alalım.

```
transA=zeros(n,m); %A'nin transpozunu bu matriste tutacagiz
for i=1:n
    transA(i,:)= A(:,i); %A'in i. sütunu transA'nin i. satırı
end
```

6. Matris – Vektör Çarpımı

Bir matrisi ve bir sütun vektörün input olarak alan; bu matris ve bu vektörun carpimi sonucu olusan vektörü output olarak veren bir fonskiyon yaziniz.

Not: Fonksiyonunuz çarpima baslamadan once aldigi matris ve vektörun çarpilabilir oldugunu test etmeli (yani matrisin sutun sayisi ile vektörün satir sayisinin ayni olup olmadigini test etmeli). Eğer matris ve vektör carpilabilir degilse hata mesaji vermeli.

```
function carpimVektor = matrisVektorCarp(A,v)
%A burada matris; v ise sutun vektöru
[m,n]=size(A);
if length(v)~=n
   error('Matris ve vektor carpilabilir degil!');
else
   carpimVektor=zeros(m,1); %carpim sonucu olusacak vektor
   for i=1:m
       carpimVektor(i)=A(i,:)*v;
   end
end
```

7. Matris – Matris Çarpımı

Simdi input olarak aldigi iki matrisi output olarak carpimini veren bir fonksiyon yazalim. Hatirlarsak bunu iki yolla yapiyorduk.

a. Birinci yolda carpim matrisinin her bir sutunu, birinci matris ile ikinci matrisin bir sutunu ile carpilmasindan olusuyordu. Bu yolu kullanarak aldigi iki matrisi carpan bir fonskiyon yazin. Ayrica burada karsilastiginiz matris – vektör carpimi icin 6. görevde olusturdugunuz fonksiyonu kullabilirisiniz.

b. İkinci yolda carpim matrisi, birinci matrisin her bir sutunu ile ikinci matriste bu sutunlara karsilik gelen satirlarin dis carpimlarinin olustrudugu matrisler toplanarak bulunuyordu. Ikinci olarak iki matrisi bu yolla carpan bir fonksiyon yazin. Bunun icin aldigi iki vektörun dis carpimina donen harici bir fonksiyon yazip; daha sonra bu fonksiyonu matris carpiminda kullanabilirsiniz.

Not: İki matrisin carpilabilir olmasi icin oncelikle birinci matrisin sutun sayisi ile ikinci matrisin satir sayisinin ayni olmalidir. Bu yuzden yazacaginiz iki fonksiyonda da oncelikle aldigi iki matrisin carpilabilir olup olmadigini test etsin. Carpilabilir degilse error mesaji versin.

a.

```
function carpimMatris = matrisCarp1(A,B)
  %aldigi A ve B matisini carpimina donen fonksiyon
  [a,b]=size(A);
  [c,d]=size(B);
  %A ve B'nin carpilabilmesi icin b=c olmak zorundadir.
  if b~=c
     error('Iki matris carpilabilir degildir!');
   else
     carpimMatris=zeros(a,d); %carpimin saklanacagi matris
     for i=1:d
       carpimMatris(:,i)= matrisVektorCarp(A,B(:,i));
     end
   end
```

b.

```
  function carpimMatris = disCarpim(vek1, vek2)
  %this function returns outter product of two given vectors
  %vek1 sutun vektoru, vek 2 satir vektöru
  m=length(vek1);
  n=length(vek2);
```

```
 carpimMatris= zeros(m,n);
 for i=1:m
    for j=1:n
        carpimMatris(i,j)=vek1(i)*vek2(j);
    end
 end
```

```
                    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function carpimMatris = matrisCarp2(A,B)
%aldigi A ve B matisini dis carpim kullanarak carpimina donen fonksiyon
[a,b]=size(A);
[c,d]=size(B);
%A ve B'nin carpilabilmesi icin b=c olmak zorundadir.
if b~=c
   error('Iki matris carpilabilir degildir!');
else
  carpimMatris = zeros(a,d);
   for i=1:b
     carpimMatris=carpimMatris+disCarpim(A(:,i),B(i,:));
   end
end
```