

Onur
SEZER
121044074

CSE 321 - HW04

```
1) def polyval (polyArray, x):  
    if len (polyArray) == 1 :  
        return polyArray [ len (polyArray) - 1 ]  
    else :  
        return x * (polyArray [ len (polyArray) - 1 ] +  
                    polyval (polyArray, x))
```

// polinomun derecesi kadar carpma islemi yapilir,
bir fazlası kadar toplama islemi yapilir

```
2) def search (list)  
    if len (list) == 0 :  
        return -1  
    mid = len (list) / 2  
    if mid == list [mid]:  
        return mid  
    elif mid > list [mid]:  
        return search (list [mid+1:])  
    else :  
        return search (list [:mid])
```

Complexity $\rightarrow O(\log n)$

3)

```
def hanoi (disk, first, second, third):
    if disk == 1:
        print " Move disk 1 from %s to rod %s " %
            (first, second)

    hanoi (disk-1, first, second, third)
    hanoi (disk-1, second, third, first)
    print " Move disk %d from %s to rod %s " %
        (first, second)
    hanoi (disk-1, third, second, first)
```

// 1 ve 3. grublar arasında hareket olmadan diskler büyükten küçüğe doğru sıralı olarak 2. gruba taşınır

4)

```
def MaxIndex(A[l...r]):
    if l == r:
        return l
    else:
        temp1 = MaxIndex(A[l ... ((l+r)/2)])
        temp2 = MaxIndex(A[((l+r)/2)+1 ... r])
        if A[temp1] >= A[temp2]:
            return temp1
        else:
            return temp2
```

5) Sorunu çözmek için algoritmanın operasyonlarını full binaryde tutarız. Kırılabilir parçalar parent node'larda, yapraklarda ise 1-1'lik çikolata barının parçalarını ifade ederiz. Çikolata barının boyutuna nm dersek her defasında $nm-1$ 'lik azalma olur. Bu işlem böyle tekrarlanır.

- 6) a) Her iki teknikte sorunu daha küçük parçalara bölmeye dayanır.
- b) İki problemin farkı; dinamik programlamada problemi parçalara böldükten sonra aynı problemin tekrarı olan parçaları bir kere de çözüp her tekrar için ayrı-bir çözüm yapmamasıdır. Divide and conquer de ise yine problem küçük parçalara ayrılır fakat parçalar ayrı ayrı çözüldükten sonra birleştirilir.

7) a) A'nın kazanma olasılığı p
B'nin kazanma olasılığı $q = 1-p$

A'nın i galibiyete, B'nin ise seriyi kazanmak için $j-1$ galibiyete ihtiyacı vardır.
Bu durum tekrarlamaya neden olur.

$$P(i, j) = pP(i-1, j) + qP(i, j-1) \quad i, j > 0 \text{ için}$$

b)

$i \backslash j$	0	1	2	3	4
0		1	1	1	1
1	0	0.40	0.64	0.78	0.87
2	0	0.16	0.35	0.52	0.66
3	0	0.06	0.18	0.32	0.46
4	0	0.03	0.09	0.18	0.29

Table 2 şeklinde
bulunan formülden
çıkarılmıştır.

$$P[4, 4] \approx 0.29$$

8)

// Python code:

```
import numpy as np
```

```
def printMaxSubSquare(arr):
```

```
    row = len(arr)
```

```
    column = len(arr[0])
```

```
    arr2 = np.zeros((row, column))
```

```
    for i in range(0,row):
```

```
        arr2[i][0] = arr[i][0]
```

```
    for j in range(0,column):
```

```
        arr2[0][j] = arr[0][j]
```

```
    for i in range(1,row):
```

```
        for j in range(1,column):
```

```
            if arr[i][j] == 1:
```

```
                arr2[i][j] = min(arr2[i][j - 1], arr2[i - 1][j], arr2[i - 1][j - 1]) + 1
```

```
            else:
```

```
                arr2[i][j] = 0
```

```
    max_of_s = arr2[0][0]
```

```
    max_i = 0
```

```
    max_j = 0
```

```
    for i in range(0, row):
```

```
        for j in range(0, column):
```

```
            if max_of_s < arr2[i][j]:
```

```
                max_of_s = arr2[i][j]
```

```
                max_i = i
```

```
                max_j = j
```

```
    print("Maximum size sub-matrix: ")
```

```
    for i in range(max_i, int(max_i - max_of_s), -1):
```

```
        for j in range(max_j, int(max_j - max_of_s), -1):
```

```
            print(arr[i][j], end="")
```

```
        print(" ")
```

```
def min(a, b, c):
```

```
    minValue = a
```

```
    if minValue > b:
```

```
        minValue = b
```

```
    if minValue > c:
```

```
        minValue = c
```

```
    return minValue
```

```

if __name__ == '__main__':
    Ar = []
    Ar.append([0, 1, 1, 0, 1])
    Ar.append([1, 1, 0, 1, 0])
    Ar.append([0, 1, 1, 1, 0])
    Ar.append([1, 1, 1, 1, 0])
    Ar.append([1, 1, 1, 1, 1])
    Ar.append([0, 0, 0, 0, 0])

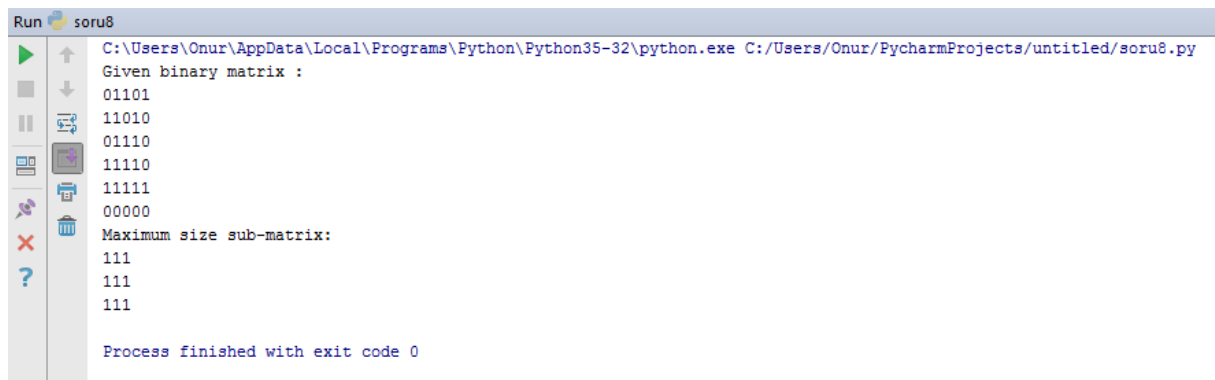
    print("Given binary matrix :")
    row = len(Ar)
    column = len(Ar[0])
    for i in range(0,row):
        for j in range(0,column):
            print(Ar[i][j], end="")
            print(" ")

    printMaxSubSquare(Ar)

```

//////////

Output:



```

Run soru8
C:\Users\Onur\AppData\Local\Programs\Python\Python35-32\python.exe C:/Users/Onur/PycharmProjects/untitled/soru8.py
Given binary matrix :
01101
11010
01110
11110
11111
00000
Maximum size sub-matrix:
111
111
111
Process finished with exit code 0

```

1) printMaxSubSquare methodunda mainden parametre olarak gelen 2d arrayin boyutlarında arr2 oluşturulur.

- a) ilk olarak ilk sütun ve ilk satır kopyalanır
- b) diğer elemanlar için aşağıdaki koşul kullanılır
 - if $M[i][j]$ is 1 then

$$S[i][j] = \min(S[i][j-1], S[i-1][j], S[i-1][j-1]) + 1$$
 - else

$$S[i][j] = 0$$

2) Kopyalanan arr2 nin max elemanı bulunur.

3) arr2 deki max elemanın değerini ve kordinatlarını kullanarak, parametre olarak gelen arr'in sub-matrix'i print edilir.

9)

// Python Code

```
import sys
```

```
def MatrixChainOrder(arr, size):
```

```
    m = [[0 for x in range(size)] for x in range(size)]
```

```
    for i in range(1, size):
```

```
        m[i][i] = 0
```

```
    # len zincir uzunluğudur
```

```
    for len in range(2, size):
```

```
        for i in range(1, size-len+1):
```

```
            j = i+len-1
```

```
            m[i][j] = sys.maxsize
```

```
            for k in range(i, j):
```

```
                # q = cost/scalar multiplications
```

```
                q = m[i][k] + m[k+1][j] + arr[i-1]*arr[k]*arr[j]
```

```
                if q < m[i][j]:
```

```
                    m[i][j] = q
```

```
    return m[1][size-1]
```

```
if __name__ == '__main__':
```

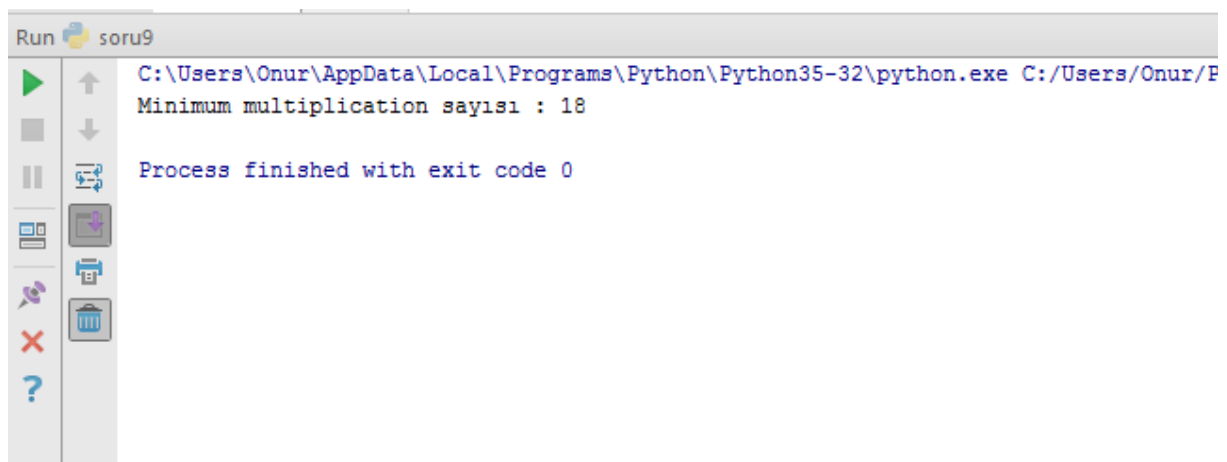
```
    arr = [1, 2, 3, 4]
```

```
    size = len(arr)
```

```
    print("Minimum multiplication sayısı : " + str(MatrixChainOrder(arr, size)))
```

////////

Output :



```
Run soru9
C:\Users\Onur\AppData\Local\Programs\Python\Python35-32\python.exe C:/Users/Onur/F
Minimum multiplication sayısı : 18
Process finished with exit code 0
```

Time Complexity: $O(n^3)$

10)

Devrim Arabaları

1961 Otomotiv Endüstri Kongresi sonrası verilen davette iş adamları, gazeteciler, bürokratlar, Cumhurbaşkanı Cemal Gürsel ülkenin kalkınmasının durumunu tartışmaktadırlar. Cemal Gürsel sinirlenip bu ülkenin otomobil bile imal edebileceğini söyler. Bir anda bu iddia ciddi bir meydan okumaya dönüşür. Devlet Başkanı Cemal Gürsel tümüyle yerli üretim bir otomobil yapılmasını emreder ve görevin TCDD işletmesine verildiği bildirilir. Yaklaşmakta olan Cumhuriyet Bayramı'na ilk yerli otomobil yetişecektir. Neredeyse imkansız bu görevi hem gerçekleştirebilecek hem de kabul edecek kişi aranır. Gündüz Serter'de karar kılınır. Gündüz Bey, güvendiği mühendislerden bir ekip kurar. Yaklaşık 130 günde sıfırdan bir otomobil imal edeceklerdir. Otomobilin gösterileceği 29 Ekim tarihine kadar neredeyse hiç görüşmemek üzere ailelerinden ayrılan ekip, Eskişehir'de kendilerine tahsis edilen eski bir atölyede buluşur. Araba yapmak için gerekli özel bir makine, tesisat olmadığı gibi basit bir vinç ve küçük el aletleri dışında hiçbir şeyleri yoktur. Filmde bir mühendislik başarısının, siyasi olaylarla baştan sona nasıl yokedilmeye çalışıldığı gözler önüne seriliyor. Türkiye'nin dışarıya bağımlılığını azaltacak bir proje olarak görünen bu olaya karşı ABD'den gelen yardım komiserlerinden, projeyi halkın gözünde küçük düşürmeye çalışmasında anlatılıyor. Uzun araştırmalar ve teknik toplantılardan sonra nasıl bir araba yapılacağına karar veren ekip, imalata geçtiğinde makine parkı eksikliğini fazlasıyla hisseder. Aslında arabadan önce yapılması gereken, arabayı yapacak makinelerin yapılmasıdır. Ancak buna zaman yoktur. Ekip her şeyi pratik çözümlerle, şartları zorlayarak halleder. Zor şartlarda, aksiliklerle son günlere yaklaşılırken ilk arabanın marşına basılır. Uykusuz geçen son hafta ile birlikte bir araba daha imal eden ekip, ertesi gün Ankara'da Paşa'nın huzuruna çıkacak arabaları 28 Ekim gecesi trene yüklerler. Devrim Arabaları filmi bize Türk mühendisinin, 20 sene öncesine kadar toplu iğne dahi üretemeyen bir ülkede inanç ve azimle her işin üstünden kalkabileceği mesajını vermektedir.