131044028

Hazal Güneş

CSE-321 HW 02

1.) Sort the array $A = \{3, 44, 38, 5, 47, 15\}$ in increasing order by selection sort, bubble sort, insertion sort, quick sort. Show steps of sorting algorithms as well.

Selection Sort:

3	44	38	5	47	15	min=3	3 < 44
3	44	38	5	47	15	min=3	3 < 38
3	44	38	5	47	15	min=3	3 < 5
3	44	38	5	47	15	min=3	3 < 47
3	44	38	5	47	15	min=3	3 < 15
3	44	38 (min)	5	47	15	min=44	44 > 38 min=38
3	44	38	5 (min)	47	15	min=38	38 > 5 min=5
3	44	38	5 (min)	47	15	min=5	5 < 47
3	44	38	5 (min)	47	15	min=5	5 - 44 change
3	5	38	44	47	15	min=38	38 < 44
3	5	38	44	47	15	min=38	38 < 47
3	5	38	44	47	15	min=38	38 > 15 min=15 change
3	5	15	44	47	38	min=44	44 < 47
3	5	15	44	47	38	min=44	44 > 38 min=38 change
3	5	15	38	47	44	min=47	47 > 44 min=44 change
3	5	15	38	44	47		

Scanned by CamScanner

Bubble Sort:

3 44 38 5 47 15	$3 < 44$	
3 44 38 5 47 15	$44 > 38$	change
3 38 44 5 47 15	$44 > 5$	change
3 38 5 44 47 15	$44 < 47$	
3 38 5 44 47 15	$47 > 15$	change
3 38 5 44 15 47	$3 < 38$	
3 38 5 44 15 47	$38 > 5$	change
3 5 38 44 15 47	$38 < 44$	
3 5 38 44 15 47	$44 > 15$	change
3 5 38 15 44 47	$44 < 47$	
3 5 38 15 44 47	$3 < 5$	
3 5 38 15 44 47	$5 < 38$	
3 5 38 15 44 47	$38 > 15$	change
3 5 15 38 44 47	$38 < 44$	
3 5 15 38 44 47	$44 < 47$	

Scanned by CamScanner

Insertion Sort:

3 44 38 5 47 15 (3 sıralıdır)

3 44 38 5 47 15 (3, 44 sıralıdır)

3 44 38 5 47 15 (38 sırayı bozar)

3 38 44 5 47 15 (5 sırayı bozar)

3 5 38 44 47 15 (sıralıdır)

3 5 38 44 47 15 (15 sırayı bozar)

3 5 15 38 44 47 (sıralıdır.)

Scanned by CamScanner

Quick Sort:

3 44 38 5 47 15 pivot=5

3 44 38 5 47 15

3 44 38 5 47 15

3 5 38 44 47 15 pivot =47

3 5 38 44 47 15

3 5 38 44 47 15

3 5 38 44 47 15

3 5 38 44 15 47 pivot=44

3 5 38 44 15 47

3 5 38 44 15 47

3 5 38 15 44 47 pivot 15

3 5 15 38 44 47

Scanned by CamScanner

2.) Briefly explain your answers for question below.

a.) Is selection sort stable? → Kararsız

b.) Is bubble sort stable? → Kararlı

c.) Is it possible to implement selection sort for linked lists with the same $\Theta(n^2)$ efficiency as the array versions?

d.) Is it possible to implement insertion sort for sorting linked lists? Will it same $O(n^2)$ efficiency as the array version? Recall that we can access elements of a singly linked list only sequentially.

a.) Selection sort kararsızdır. Bunun nedeni dizide değerleri birbirine eşit olan öğeler birbirlerine göre konumlarını korumazlar. Ör:

4^R 2 3 4^S 1

1 2 3 4_1 4_2

4^R 4^S'den önce gelmesine rağmen sıralamadan sonra yerler değişir. Kararsızdır.

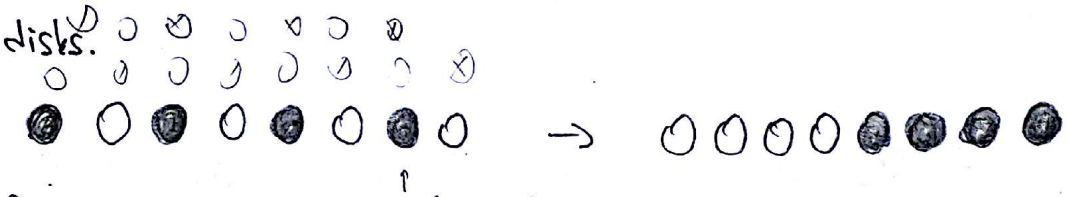
b.) Bubble Sort kararlıdır. R ve S eşit değerli elemanlar olsun R S'den önce geliyor ise sıralandıktan sonrada R S'den önce gelir.

c.) Selection sortu linked list ile implement etmek mümkündür. $\Theta(n^2)$ verimlilikte çalışır.

d.) Evet insertion sortu implement ederken linked list kullanmak mümkündür.

Scanned by CamScanner

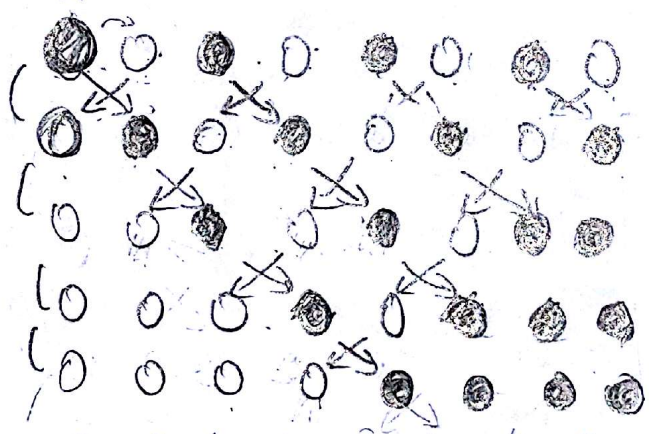
3. Alternating disks: You have a row of 2n disks of two colors, n dark and n light. They alternate: dark, light, dark, light and so on. You want to get all the dark disks to the right-hand end, and all the light disks to the left-hand end. The only moves you are allowed to make are those which interchange the positions of two neighboring disks.



Design an algorithm for solving this puzzle and determine the number of moves it makes. Hint: Thinking about the puzzle as a sorting like problem may and may not lead you to the most simple and efficient solution.

Quick Sort pivot ⇒ en ortası seçerim.

dark = 0
light = 1



10 islem yapıldı

$$\sum_{i=0}^{n} i = \frac{n \cdot n+1}{2}$$

$$4+3+2+1 \Rightarrow \frac{n \cdot (n+1)}{2}$$

$$n = \frac{size}{2}$$

```
def Alternating disks (* disks):
    for i in range(0, len(disks)/2)
        for j in range(i, len-i, 2)
            if (disks[j] < disk[j+1])
                swap (disks[j], disks[j+1])
return
```

$\frac{n}{2}$

len-*

*

$\frac{n}{2}$

0 -> 4
1 -> 3
2 -> 2
1 -> 1

$\frac{n}{2} - i$

$\frac{len-2*}{2}$

Scanned by CamScanner

4.) Give an example of a text of length n and a pattern of length m that constitutes the worst-case input for the brute-force string-matching algorithm. Exactly how many character comparisons are made for such input?

Hint: It will suffice to limit your search for an example to binary texts and pattern.

worst case: $\underbrace{0000000000\overset{(9)}{0}1}_{n}$ $\quad \underbrace{1}_{m}$ $\quad$ Text[0 ... n-1]

Pattern[0 ... m-1]

```
for i=0 to n-m
    J=0
    while ( J<m and P[J]=T[i+J] ) do
        J = J+1
    endwhile
    if( J==m)
        return i
    endif
end for.
```

$n-m+1$ $\quad$ m

$m.(n-m+1) \Rightarrow O(m.n)$ //

$O(m^2)$ olmamasının sebebi $-m$ olmasıdır.

Scanned by CamScanner

5. Consider the problem of counting, in a given text, the number of substrings that start with A and end B. For example, there are four such substrings

C A B A A X B y A ABAABA 2 1 1

a.) Design a brute-force algorithm for this problem and determine its efficiency class. → recursive mi?

b.) Design a more efficient algorithm for this problem.

a) ⇒
```
def brute_force(*T):
    count = 0;
    for i in range(0, n-2):
        if (T[i] == 'A'):
            for (j in range(i+1, n-1)):
                if(T[j] == B):
                    ++count
return
```

i=0	n
i=1	n-1
i=2	n-2
+ i=n-2	2

$(n.(n+1)/2) - 1 = \in \Theta(n^2)$

Eger hiç B yok ise ve hepsi A ise worst case olur.

Scanned by CamScanner

b.) Sondan başlayıp B'leri sayarak A'ya geldiğinde toplanırsa daha efficient bir program olur.

```
def brute_force_b (*T):
    temp = 0;
    count = 0;
    for i in range(n-1, 0, -1):

        if (T[i] == B):

            ++ temp;
        if (T[i] == A):
            count = count + temp;

return:
```

Bu algoritma $\in \Theta(n)$'dir.

Scanned by CamScanner

6. Ferrying soldiers: A detachment of n soldiers must cross a wide and deep river with no bridge in sight. They notice two 12-years old boys playing in rowboat by the shore. The boat is so tiny, however, that it can only hold two boys or one soldier. How can the soldiers get across the river and leave the boys in joint possession of the boat? How many times does the boat needed to pass from shore to shore?

Hint: solve the problem for $n=1$

2 çocuk ya da 1 asker taşıyabilir ve çocuklar başlangıç konumunda olmalı.

sol taraf	kayık	nehir	kayık	sağ taraf

	sol taraf	kayık	nehir	kayık	sağ taraf	
	bbs					kayıkta 2 çocuk →
1. adım	s	bb			bb	
	s					kayıkta 1 çocuk ←
2. adım	s	b			b	
	sb				b	
	b	s			b	asker →
3. adım	b				sb	
	b	b			s	kayıkta 1 çocuk ←
4. adım	bb				s	

1 asker için 4 adımda karşıya geçilir.

n asker için 4.n adım gerekir.

Scanned by CamScanner

8.) Flipping Pancakes: There are n pancakes all of different sizes that are stacked on top of each other. You are allowed to slip a flipper under one of the pancakes and flip over the whole stack above the flipper. The purpose is to arrange pancakes according to their size with the biggest at the bottom.

a.) Design an algorithm to solve this problem.

b.) Test your algorithm with 1, 2, 10, 7, 8, 3 number labeled pancakes. 1 is at the top, 3 is at the bottom and use | to show flipper position. Show your result step by step applying algorithm.

c.) Analyze your algorithm for best case and worst case scenaria.

a.) - En büyük pancake'i bul
- En yukarı döndür.
- En alta döndür.

```
def flip(*arr, i):
    temp = 0:
    start = 0:
    while(start < i):
        temp = arr[start]:
        arr[start] = arr[i]:
        arr[i] = temp:
        start++;
        i--;
return arr:
```

```
def findMax(*arr, n)
    mi = 0
    i = 0
    for(i in range(n):
        if (arr[i] > arr[mi]) :
            mi = i:
    return mi:
```

```
def pancakeSort(*arr, n):
    for(curr_size in range(n, 1, -1);
        mi = findmax(arr, curr_size):
        if (mi != curr_size - 1) :
            flip(arr, mi):
            flip(arr, curr_size - 1)
return arr:
```

Scanned by CamScanner

b.) 1 2 10 7 8 3 → max = 10 i = 2 (index) currsize = 6

10 2 1 7 8 3

3 8 7 1 2 10 → max 8 i = 1 currsize = 5

8 3 7 1 2 10

2 1 7 3 8 10 → max = 7 i = 2 currsize = 4

7 1 2 3 8 10

3 2 1 7 8 10 → max = 3 i = 0 currsize = 3

3 2 1 7 8 10

1 2 3 7 8 10 → max = 2 i = 1 currsize = 2

2 1 3 7 8 10

1 2 3 7 8 10 → max = 1 i = 0 cursize = 1 X

1 2 3 7 8 10

c.) $B(n) \in O(n^2)$

$W(n) \in O(n^2)$

Scanned by CamScanner