HARUN ALBAYRAK - 171044014 - CSE 222/505 - Homework 2

PART 1)

1) somefunction(rows, cols){
//θ(n) for(i=1; i<=rows; i++){
//θ(n) for(j=1; j<=cols; j++){
//θ(1) print(*)
}
//θ(1) print(newline)
}

Steps	freq.	total
1	rows+1	rows+1
1	rows(cols+1)	rowscols+rows
1	rowscols	rowscols
1	rows	rows
		2rowscols + 3rows + 1

T_w(Worst) = $\theta(n^2)$ = θ(rows.cols)

T_b(Best) = $\theta(n^2)$ = θ(rows.cols)

T_{av}(Average) = $\theta(n^2)$ = θ(rows.cols)

$T(n) = \theta(n^2)$

* The worst case and the best case must be the same. Because it has to go to the end of the loops.

* θ(cols) is come from inner loop. θ(rows) is also come from outer loop.

2) somefunction(a,b){
//θ(1) if(b==0)
//θ(1) return 1
//θ(1) answer = a
//θ(1) increment = a
//θ(n) for(i=1; i<b; i++){
//θ(n) for(j=1; j<a; j++){
//θ(1) answer += increment
}
//θ(n) increment = answer
}
//θ(1) return answer
}

steps	freq	total
1	1	1
1	1	1
1	1	1
1	1	1
1	b+1	b+1
1	b(a+1)	ba+b
2	ba	2ba
1	b	b
1	1	1
		3ba+3b+6

$T_w = \theta(n^2) = \theta(b*a)$

$T_b = \theta(1)$

$T_{av} = \theta(n^2) = \theta(b*a)$

$T(n) = O(n^2) = \Omega(1)$

* If it goes to the end of the loops, this is the worst case. $\theta(a)$ is come from inner loop and $\theta(b)$ is also come from outer loop.

* If it enter the first 'if' statement, this is the best case. Berouse it has got 'return' statement.

3)
```
somefunction( arr[], arr_len){
    val = 0
    for(i=0 ; i<arr_len/2 ; i++)
        val = val + arr[i]
    for( i =arr_len/2 ; i<arr_len ; i++)
        val = val - arr[i]
    if( val >=0)
        return 1
    else
        return -1
}
```

steps	freq	total
1	1	1
1	$\frac{arr\_len}{2}+1$	$\frac{arr\_len}{2}+1$
2	arr_len/2	arr_len
1	$\frac{arr\_len}{2}+1$	$\frac{arr\_len}{2}+1$
2	arr_len/2	arr_len
1	1	1
1	1	1
1	1	1
1	1	1
		2arr_len+7

$T_w = \Theta(n)$
$= \Theta(arr\_len/2)$

$T_b = \Theta(n)$
$= \Theta(arr\_len/2)$

$T_{av} = \Theta(n)$
$= \Theta(arr\_len/2)$

$T(n) = \Theta(n)$

* It has two 'for' loop but these loops don't nested loop. And this function has to go to the end of the loops. Hence, the best case and the worst case is the same. And both of them are $\Theta(n)$.

4)
```
somefunction(n) {
    c = 0
    for(i = 1 to n*n)
        for( j=1 to n)
            for( k=1 to 2*j)
                c = c+1
    return c
}
```

steps	freq	total
1	1	1
1	n^2+1	n^2+1
1	$n^2(n+1)$	n^3+n
1	$n^3(2j+1)$	$2n^3j+n^3$
2	$2n^3j$	$4n^3j$
1	1	1
		$6n^3j+2n^3+n^2+n+3$

$T_w = \Theta(n^3)$

$T_b = \Theta(n^3)$

$T_{av} = \Theta(n^3)$

$T(n) = \Theta(n^3)$

* This function includes three nested 'for' loops. Also, it has to go to the end of the loops. Hence, the best case and the worst case is the same. This function's complexity is $\Theta(n3)$, because it has three nested 'for' loops.

5)
```
otherfunction(xp, yp){
    temp = xp
    xp = yp
    yp = temp
}
some_function(arr[], arr_len){
    for(i=0; i<arr_len-1; i++){
        min_idx = i
        for(j=i+1; j<arr_len; j++)
            if(arr[j] < arr[min_idx]
                min_idx = j
        otherfunction(arr[min_idx], arr[i])
    }
}
```

$T_w = \Theta(n^2)$
$= \Theta(arr\_len * arr\_len - i - 1)$

$T_b = \Theta(n^2)$
$= \Theta(arr\_len * arr\_len - i - 1)$

$T_{av} = \Theta(n^2)$
$= \Theta(arr\_len * arr\_len - i - 1)$

$T(n) = \Theta(n^2)$

* The complexity of 'otherfunction' is $\Theta(1)$.

* The complexity of 'somefunction' is $\Theta(n^2)$, Because it has two nested loops And complexity of each of them is $\Theta(n)$.

---

6)
```
otherfunction(a,b){
    if(b == 0)
        return 1
    answer = 0
    increment = a
    for(i=1 to b){
        for(j=1 to a){
            answer += increment
        }
        increment = answer
    }
    return answer
}
somefunction(arr, arr_len){
for(i=0 to arr_len)
    for(j=i to arr_len)
        ? if((otherfunction(arr[i], 2) == arr[j])
            if(otherfunction(arr[i], arr[j])
            ~~else if(otherfunction(arr[j], 2) == arr[i])~~
            ~~print(arr[j], arr[i])~~
}
```

$T_w = \Theta(n^2)$
$T_b = \Theta(1)$

$T_w = \Theta(n^4)$
$= \Theta(arr\_len * arr\_len - i * b * a)$

$T_b = \Theta(n^2)$
$= \Theta(arr\_len * arr\_len - i)$

$T_{av} = \Theta(n^4)$
$= \Theta(arr\_len * arr\_len - 1 - 1 * b * a)$

$T(n) = O(n^4)$
$= \Omega(n^2)$

* The best case for the 'otherfunction' is $\Theta(1)$ because it has 'if' and 'return' statement at the beginning of the function. The worst case is $\Theta(n^2)$ because it has two nested loops.

* The best case for the 'somefunction' is $\Theta(n^2)$ and the worst case is $\Theta(n^4)$.

7)
```
otherfunction(X,i){
  //Θ(1) s=0
  //Θ(logn) for(J=1; J<=i; J=J*2)
      //Θ(1) s = s + X[J]
  //Θ(1) return s
}

somefunction(arr[], arr_len){
  //Θ(n) for(i=0; i<=arr_len-1; i++)
    //Θ(logn) A[i] = otherfunction(arr, i) / (i+1)
    //Θ(1) return A
}
```

$T_w = \Theta(\log n)$

$T_b = \Theta(\log n)$

$T_w = \Theta(n.\log n)$
$= \Theta(arr\_len - 1 * i)$

$T_b = \Theta(n.\log n)$
$= \Theta(arr\_len - 1 * i)$

$T_{av} = \Theta(n.\log n)$
$= \Theta(arr\_len - 1 * i)$

$T(n) = \Theta(n.\log n)$

* The complexity of the 'otherfunction' is $\Theta(\log n)$. Because J is multiplied by 2 each time

* The complexity of the 'somefunction' is $\Theta(n \log n)$. It has one 'for' loop and this for loop includes the 'otherfunction'. Since the complexity of 'otherfunction' is $\Theta(\log n)$, the function's complexity is $\Theta(n \log n)$

8)
```
somefunction(n){
  //Θ(1) res = 0
  //Θ(1) J = 1
  //Θ(1) if(n<10)
    //Θ(1) return n+10

  //Θ(1) for(i=0; i>1; i--)
    //Θ(1) while(n%i == 0)
      //Θ(1) n = n/i
      //Θ(1) res = res + J*i
      //Θ(1) J *= 10
  //Θ(1) if(n>10)
    //Θ(1) return -1

  //Θ(1) return res
}
```

$T_w = \Theta(1)$

$T_b = \Theta(1)$

$T_{av} = \Theta(1)$

$T(n) = \Theta(1)$

* Even if it has two nested loops, since these loops' time are constant time, their complexity are $\Theta(1)$. Also the best case and the worst case is the same.

PART 2

```
1) distance (x1, y1, x2, y2){
//Θ(1) return sqrt( pow(x1-x2, 2) + pow(y1-y2, 2) )
}
question_1(arr[][], arr_length, x1, y1) {
//Θ(1) if(arr_length == 0)
//Θ(1) throw Exception
num = distance(x1, y1, arr[0][0], arr[0][1])
//Θ(n) for(i = 0 to arr_length)
//Θ(1) if( distance(x1, y1, arr[i][0], arr[i][1]) < num)
//Θ(1) num = distance(x1, y1, arr[i][0], arr[i][1])
//Θ(1) return num
}
```

$T_w = \Theta(n)$

$T_b = \Theta(1)$

$T_{av} = \Theta(n)$

$T(n) = O(n)$
$= \Omega(1)$

* The 'question_1' function's complexity is $O(n)$ because of it's worst and best cases.

```
2)a) question_2_1(arr[], arr_length) {
//Θ(1) if(arr_length <= 2)
//Θ(1) throw Exception
//Θ(n) for(i=1 to arr_length-1)
//Θ(1) if(arr[i] <= arr[i+1] && arr[i] <= arr[i-1])
//Θ(1) return arr[i]
return -1
}
b) question_2_2(arr[], arr_length) {
//Θ(1) if(arr_length <= 2)
//Θ(1) throw Exception
//Θ(1) qarr[]
//Θ(n) for(i=1 to arr_length-1)
//Θ(1) k = 0
//Θ(1) if(arr[i] <= arr[i+1] && arr[i] <= arr[i-1])
//Θ(1) qarr[k] = arr[i]
//Θ(1) k++
//Θ(1) return qarr
}
```

a) $T_w = \Theta(n)$
$T_b = \Theta(1)$
$T_{av} = \Theta(n)$
$T(n) = O(n)$
$\Omega(1)$

b) $T_w = \Theta(n)$
$T_b = \Theta(n)$
$T_{av} = \Theta(n)$
$T(n) = \Theta(n)$

* First function's complexity is $O(n)$ and second function's complexity is $\Omega(n)$.

3) question_3(arr[], arr_length, b) {
//θ(n) for(i=0 to arr_length){
//θ(1) if(b > arr[i]){
//θ(1) int m = arr[i]
//θ(n) for(j=0 to arr_length)
//θ(1) if (m + arr[j] == b)
//θ(1) return true
}
}
//θ(1) return false
}

$T_w = \theta(n^2)$

$T_b = \theta(1)$

$T_{av} = \theta(n^2)$

$T(n) = O(n^2)$
$= \Omega(1)$

* The worst case is $\theta(n^2)$ because it has two nested loops.

* In case of entering to 'if' statements in the first entry. The best case happens. It's complexity is $\theta(1)$

* The complexity of function is $O(n^2)$

---

4) bubbleSort(arr[], len){
//θ(n) for (i=0 to len-1){
//θ(1) swapped = false
//θ(n) for (j=0 to len-i-1){
//θ(1) if (arr[j] > arr[j+1]) {
//θ(1) swap (arr[j], arr[j+1])
//θ(1) swapped = true
}
//θ(1) if (swapped == false)
//θ(1) break
}
}
question_4(arr[], arr_length){
//θ(1) if(arr_length <= 1)
//θ(1) throw Exception
//θ(n²) bubbleSort (arr, arr_length)
//θ(n) for(i=1 to arr_length){
//θ(1) q4arr[]
//θ(n) for (j=0 to i)
//θ(1) q4arr[j] = arr[j]
//O(n²) if(question_3(q4arr, q4arr.length, arr[i]) == false)
//θ(1) return false
}
//θ(1) return true
}

$T_w = \theta(n^2)$
$T_b = \theta(n)$

$T_w = \theta(n^2)$

$T_b = \theta(n)$

$T_{av} = \theta(n^2)$

$T(n) = O(n^2)$
$= \Omega(n^2)$

* The question4's complexity is $O(n^2)$ because it's best case is $\theta(n)$ and it's worst case is $\theta(n^2)$.