CSE 321 - Homework 4
Sena Özbelen - 1901042601

1-) This algorithm is based on recursion. The aim is to try all possible paths to find the maximum point. Since there are only 2 ways to move (down and right), the algorithm calls the function with these movements recursively if it is possible.

T(i, j) = T(i, j+1) + T(i+1, j) + c

Time complexity = O(2 ^ ((n-1)*(m-1)))

2-) This algorithm sorts the array first by using a decrease and conquer algorithm, insertion sort. After sorting the array, it finds the median. If the length is odd, then it divides the length by 2 (len = 3, 3/2 = 1.5 = 1 as integer) to find the median element's index. If the length is even, then it computes the average of median elements. Time complexity of this algorithm is equal to time complexity of insertion sort which is O(nlogn).

3-)

a. To have a linear time complexity by using circular linked list, this algorithm traverses the list by deleting the eliminated node. It starts from P1 and when we visit P1, it updates the next node as P3 since P2 got eliminated. Up until reaching the node itself again (since it is circular, if there is only one node, it should point itself), it traverses the list by deleting.

b. There is a pattern discovered in this problem.

number of players	winner
1	1
2	1
3	3
4	1
5	3
6	5
7	7
8	1

Winner's position increases by 2 and when it encounters a number which is powers of 2, then it goes back to 1. We can write numbers in terms of powers of 2.

If we have 5 players, then it would be written as 2^2 + 1. Since we know that powers of 2 is reset point, we will focus on the rest which is 1 here. The starting point is now 1 and we are 1 step ahead. Since the increase amount is 2, our current winner is 1 + 2*1 = 3 .

If we have 21 players, it is written as 2^4 + 5. The rest part shows that we are 5 step ahead. Therefore, the winner is 1 + 2*5 = 11.

We can write this as = 2*(rest) + 1

This algorithm calculates the biggest power of 2. Then, it calculates the rest part. Using the formula above, it determines the winner. Finding the biggest power of 2 takes logarithmic time so overall, we have logarithmic time complexity.

4-) Binary search recursion formula is:

T(n) = T(n/2) + c since we call the function with half of the input size according to key. This gives us O(logn base=2) complexity.

Ternary search divides it by 3 so its formula is:

T(n) = T(n/3) + c and this gives us O(logn base=3). The divisor determines the base of the logarithm since it determines the input size of each recursive call. However, the number of comparisons are more in ternary search so this makes the algorithm slower. If the split number increases, the number of comparison will increase too.

If it is divided into n parts, then it will be linear time because we should check all elements in order to find in the worst case.

5-)

a. Time complexity of interpolation for the best case is O(1) since this algorithm is based on binary search and we know that binary search has O(1) complexity for best case when the key is the first middle element. Similarly, when the key is the calculated middle element of interpolation search, it will be constant time.

b. Interpolation search is an improved version of binary search. Whereas we find the middle element to compare the key element by adding high and low numbers and dividing them by 2 in binary search, interpolation search decides its middle element according to the value of the key. It has this formula to compute the comparison element : mid = low + ((key - arr[low]) * (high - low) / (arr[high] - arr[low])). The rest of the operations is the same. It compares the key to the calculated middle element and calls the function recursively according to comparison result.

	Interpolation Search	Binary Search
Best Case	O(1)	O(1)
Average Case	O(log(logn))	O(logn)
Worst Case	O(n)	O(logn)

Their best case situations are the same as explained in the first part of the question. The worst case of interpolation search is slower than the binary search's worst case. When elements grow exponentially, this causes the worst case of interpolation since it has to traverse all elements.