United International University (UIU)

Dept. of Computer Science and Engineering (CSE)
Mid Exam   Year: 2023   Trimester: Summer
Course: CSE 2215 Data Structure and Algorithms-I
Total Marks: 30, Time: 1 hour 45 minutes

(Any examinee found adopting unfair means will be expelled from the trimester/program as per UIU disciplinary rules)

There are FOUR questions. Answer all of them. Figures in the right-hand margin indicate full marks.

---

1. a) Demonstrate how **Descending Order Merge Sort** will work on the following data?   [3]

    y p z x r s

    Here, x=last two digits of your student id+1, y=x+3, z=x+y, p=y+z, r=x+2, s=y+9
    You must show the entire sorting procedure using a recursion tree.

    b) Discuss the time complexity of the following algorithm.   [3]

    ```
    sum=0;
    for(i=2; i<=n; i++){
        for(j=2; j<=i; j++){
            sum=sum+i+j;
        }
    }
    printf("%d", sum);
    ```

2. a) What will be the index returned by the partition function PARTITION (A,1,8) for the following array elements in **Ascending Order Quicksort**?   [3]

    18 23 56 26 89 37 28 48

    Also, show the condition of the given array once the partition function is executed.

    > **Partition Function:**
    >
    > ```
    > PARTITION(A,p,r)
    >   1  x ← A[r]
    >   2  i ← p-1
    >   3  for j ← p to r-1
    >   4        do if A[j]<=x
    >   5               then i ← i+1
    >   6  exchange A[i] ↔ A[j]
    >   7  exchange A[i+1] ↔ A[r]
    >   8  return i+1
    > ```

    b) Consider the following array of 5 elements   [3]

    Array: 30,10,40,20,15

    I.   How many times will the condition of the while loop in the Descending Order Insertion Sort Algorithm be executed for the following data?
    II. How many times will the while loop be executed if the above array was already sorted in ascending order?
    III. How many times will the while loop be executed if the above array was already sorted in Descending order?

c) Consider the following array declaration in C. [3]
```
double A[80][90];
```
Find the memory location of `A[60][70]` if `loc(A[15][20])`=x+1200, where x=last four digits of your student ID.
Assume column-wise memory is allocated for the array, where each data of type double is 8 bytes.

3. a) Show the step-by-step simulation for searching character 'w' using a **binary search algorithm** in the following sorted array of characters. For each step, you must mention the high, low and mid values indices. [3]

$$b, d, f, g, h, k, l, q, r, s, w, y$$

b) Prove that running time of $f(n) = 3n^3 + 2n^2 + 5n + 1$ is $O(n^3)$ [2]

c) Given a linear/single linked list containing six nodes whose data values are 2, 7, 5, 10, 8, and 0 respectively. Show the step-by-step simulation and pseudocode for the following operations. [2]
    I.   Delete the element at the second position of the linked list.
    II. Find the minimum of the linked list.
    III. Insert the square of the first element into the end of the linked list.

4. a) Consider the following code snippet. You will have to modify the code so that you can perform the "deleteLast" operation in O(1) time. After the necessary modifications, complete the implementation of both the" deleteLast" and "insertAtMiddle" functions. [4]

Note: The "deleteLast" function will delete the last element of the linked list whilst the "insertAtMiddle" function will insert an element in the middle of the linked list.

```
struct node {
      int value;
      struct node *next;
      struct node *prev;
};
struct node *head;
void deleteLast(){
}
void insertAtMiddle(int value){
}
int main() {
      return 0;
}
```

b) Show the status of a **STACK** implemented by a linear linked list for the operations given below. Here, x = -12 , y = x - 5 + 2x , and z = y + x. [2]
Pop(), Push(y), Push(y+z+x), Pop(), Pop(), Push(z), Push(z), Push(x*y), Push(x+y), Push(y - z), Pop(), Pop().

c) Show the status of a **QUEUE** of size 5 implemented by an array for the operations given below. Here, x = 43 , y = x + 13, and z = 2y. [2]
Here, Enqueue and Dequeue are meant by insertion and deletion, respectively.
Enqueue(2x+y), Enqueue(y), Enqueue(z+5), Dequeue (), Enqueue(y*x), Enqueue(y+z), Enqueue(z-y), Dequeue (), Dequeue ().