United International University
Department of Computer Science and Engineering

CSI 217:Data Structure Mid Exam : Summer 2018

Total Marks: 30 Time: 1 hour 45 minutes

There are FIVE questions. Figures in the right-hand margin indicate full marks.

1. (a) Suppose you have a 350-element **unsorted array**. You need to execute exactly six search operations on this array. Show with calculations which of the search algorithms will perform better in this case in terms of worst case runtime, linear or binary search. (Assume that sorting requires runtime nlog n). (2)

   (b) Consider the following algorithm for insertion sort and answer the following questions.

```
INSERTION-SORT(A)
    for j ← 2 to n
        do key ← A[ j ]
         Insert A[ j ] into the sorted sequence A[1 . . j -1]
          i ← j - 1
          while i > 0 and A[i] > key
              do A[i + 1] ← A[i]
                 i ← i – 1
          A[i + 1] ← key
```

   i. In which order does this algorithm sort the array elements, ascending or descending? (1)

   ii. What changes should you make in this algorithm if you want to sort in reverse order? (2)

2. (a) Snape is a faculty member of the computer science department at Hogwarts. He instructed his students to design a computer program to store the information of 10000 students. As a hint, he told that the following structure can be used: (3)

```
struct student {
        char name[20];
        int id;
        float cgpa;
        char hometown[30];
};
```

Hermione and Ron are two CS students. Hermione used a linked list based implementation but Ron used Array based implementation to store the information. The program that is implemented by Ron crashed. Can you discuss the possible causes of crashing the program? What are the benefits of Hermione's implementation over Ron's implementation?

   (b) Suppose there is a singly linked node declared as below: (2)

```
struct SNode {
        int data;
        struct SNode *next;        //will keep address of next SNode
};
```

Now write a function **int max(struct SNode* first)** that returns the max element of a given linked list.

Sample Input:

For Linked List: 10 → 15 → 13 → 14 → 12 Output: 15

3. (a) Is the following statement true or false? If true, establish your claim. (1)
$(n/500)^2$ is $O(n)$

(b) Write a code (pseudocode/in any language) that arranges n data in an array in a way similar to the to-and-fro movement of a Pendulum. (3)

Sample Input	Sample Output
1 3 2 5 4	5 3 1 2 4
11 12 31 14 5	31 12 5 11 14

- The minimum element must come in center position of array. If there are even elements, then minimum element should be moved to (n-1)/2 index (considering that indices start from 0)
- The next number (next to minimum) in the ascending order goes to the right, the next to next number goes to the left of minimum number and it continues like a Pendulum.

(c) Consider the declaration of a two-dimensional array in C: char a[100][100]; Assuming that the main memory is byte-addressable and the address of a[0][0] is 1001, find the address of a[40][50]. Note that a 2-D array is stored in row major order in C language. (1)

4. Suppose that you are given the following doubly linked list implementation: (5)

```
struct listNode {
        int item;          //will store data
        struct listNode *next;          //will keep address of next node
        struct listNode *prev;          //will keep address of previous node
};
struct listNode * head;          //points to the first node of the list
struct listNode * tail;          //points to the last node of the list
```

Now design a function to print alternate nodes of the given Linked List, first from head to tail, and then from tail to head.
For Linked List: 1 → 2 → 3 → 4 → 5 Output: 1 3 5 5 3 1
For Linked List: 1 → 2 → 3 → 4 → 5→ 6 Output: 1 3 5 6 4 2

5. (a) Write a program that will create and initialize and one queue and two stacks. Then it will perform the following operations: (4)

- Write a loop that will run 10 times and each time insert an integer into one of the stack. The value of the integer is equal to the value of the loop variable.
- Then, there will be another loop that will run until the stack you used earlier is emptied and each time it will pop elements from the stack and insert it to the queue or to the second stack alternatively.
- After that, there will be another loop that will each time remove elements simultaneously from the stack and the queue and compare the numbers removed from them.
- At the end of the program, it should output how many times the numbers removed from the queue were greater than the numbers removed from the stack.

(b) Suppose Stack is implemented using an array and Queue is implemented using a circular array. Now consider the following sequences of push, pop, enqueue and dequeue and find the value of top of the stack and rear, front and size of the queue after the series of the operations. Also show the elements in the Stack and the Queue after the operations. (3)
push(5), enqueue(7), push(10), enqueue(12), push(dequeue()), enqueue(37)
enqueue(pop()), push(43), push(dequeue()), enqueue(pop()), pop(), dequeue()

(c) Write a recursive function to empty a Stack. Assume all necessary functions and data structures are implemented accordingly. Your function should take the reference of the stack as input parameter. (3)

OR,

Write an algorithm to represent the **Enqueue** and **Dequeue** methods of implementing a Queue using two Stacks, named **inputStack** and **outputStack**.