United International University
Department of Computer Science and Engineering

CSI 217: Data Structures Mid-term Examination : Spring 2019

Total Marks: 30 Time: 1 hour and 45 minutes

Answer all 5 questions. Numbers to the right of the questions denote their marks.

1. (a) Find out the run time complexity of the following algorithm: 3

```
int factorial(int n) {
    if(n==1)
        return 1;
    else
        return n * factorial(n - 1);
}
```

(b) Show the manual tracing of **selection sort** for the array $A = \{10, 4, 1, 2, 3, 5, 6, 15, 7, -5\}$ if we want to sort the array in **ascending order**. 3

2. (a) In a **sorted array with duplicate elements**, write an efficient searching algorithm to **count the number of occurrences of an element**. Some examples are as follow: 3

Array	Number to search	Occurrence
$\{2, 5, 5, 5, 6, 6, 8, 9, 9, 9\}$	5	3
$\{2, 2, 4, 4, 5, 6, 7, 8, 11, 12\}$	6	1
$\{1, 1, 2, 4, 7, 10, 12, 12, 14\}$	13	0

(b) Which of the searching operation in **question 2(a)** did you use? Fill up the following table with the element comparisons if the arrays were like the following: 3

Array	Number to search	Element comparisons to find occurrences using linear search	Element comparisons to find occurrences using binary search
$\{1, 1, 1, 1, 1, 1, 1\}$	1	?	?
$\{1, 2, 2, 3, 5, 5, 12\}$	12	?	?

3. Suppose there is a singly linked node declared as below: $3 + 3 = 6$

```
struct node{
    int value;
    node *next;
}
```

(a) Given a pointer to the **first node** of the linked list (head), you need to write a code to print the **values** of the **first three nodes** and the **last three nodes**. For this problem, you can assume that the linked list contains more than three nodes. Example:

Linked List	Output
$9 \rightarrow 8 \rightarrow 7 \rightarrow 6 \rightarrow 5$	First three: 9, 8, 7 Last three: 7, 6, 5

(b) For the same linked list, write a code to **insert** the **minimum** node from the list, in front of the list. See the example below for clarification.

Linked List	Nodes after inserting minimum
$9 \rightarrow 6 \rightarrow 8 \rightarrow 5 \rightarrow 7$	$5 \rightarrow 9 \rightarrow 6 \rightarrow 8 \rightarrow 5 \rightarrow 7$

4. (a) Suppose there is a doubly linked node declared as below: 3

```
struct node {
    int value;
    node *next;
    node *prev;
}
```

Given a pointer to the **last node** of the doubly linked list only, write a code to **sequentially** print the nodes starting from the **first node** to the **last node**.

(b) Given the same node structure, now assume that the doubly linked list is **circular**. Given a pointer to the **first** node of this list, write a code to sequentially print the nodes starting from the **last node** to the **first node**. 3

OR,

4. Using the structure of **question 3**, show the effect of each of the statements given below: 6

```
temp = (node*)malloc(sizeof(node));
temp1 = (node*)malloc(sizeof(node));
temp2 = (node*)malloc(sizeof(node));
temp->value = 10;
temp1->value = 20;
temp2->value = 30;
temp2->next = temp;
temp->next = temp1;
temp1->next = NULL;
free(temp2);
temp3 = (node*)malloc(sizeof(node));
temp3->value = 15;
temp3->next = temp1
```

5. (a) Your task is to implement stack two ways – using **array** and using **linked list**. Write down pseudocodes to demonstrate how you can implement **push** and **pop** functions for these cases. $2 + 2 = 4$

(b) Your task is to implement the function $pop\_specific(stack, key)$ from a stack. The procedure is simple – you have to pop elements one-by-one from the stack until you pop key. If you cannot find key in the stack, you should return **false**. Write down a pseudocode to demonstrate this function. 2