UNIVERSITI TEKNOLOGI MARA
FINAL EXAMINATION

COURSE	:	DATA STRUCTURES AND ALGORITHMS
COURSE CODE	:	ECE532
EXAMINATION	:	JUNE 2019
TIME	:	3 HOURS

INSTRUCTIONS TO CANDIDATES

1. This question paper consists of five (5) questions.
2. Answer ALL questions in the Answer Booklet. Start each answer on a new page.
3. Do not bring any material into the examination room unless permission is given by the invigilator.
4. Please check to make sure that this examination pack consists of:

   i) the Question Paper
   ii) an Answer Booklet – provided by the Faculty

5. Answer ALL questions in English.

DO NOT TURN THIS PAGE UNTIL YOU ARE TOLD TO DO SO

This examination paper consists of 7 printed pages

QUESTION 1

a) The following is an array named **scores**, with 10 components of type **double**.

scores = {2.5, 3.4, 4.1, 6.2, 6.2, 7.1, 7.6, 8.5, 8.5, 9.5};

The code segment as shown in **Figure Q1a** is written to ensure that the elements of **scores** are in increasing order. However, there are errors in the code. Identify **THREE** (3) errors and show the correct code.

```
for (int i = 0; i <=10; i--)
    if(scores[i] >= scores[i+1])
    cout <<i <<" and " <<(i+1);
    cout <<" elements of scores are out of order.";
    cout <<endl;
```

Figure Q1a

(5 marks)

b) Linked-list is a commonly used data structure which consists a set of nodes in a sequence.

i) Compare **TWO** (2) main differences between **linked-list** and **array** in data structures.

(4 marks)

ii) Based on the structure declaration in **Figure Q1b** below, write a function definition for `int getCount(struct node *p)`. This function will count the number of nodes in a singly linked-list.

```
struct node
{
    int data;
    node *link;
};
```

Figure Q1b

(6 marks)

c) Illustrate a **BINARY TREE** from the given integers below with 59 as the root node. Then, list the nodes for **Post-order Traversal**.

59	69	6	3	48	55	52	73	12	60

(5 marks)

QUESTION 2

a) Based on the given structures declaration shown in **Figure Q2a:**

```
struct staff {
    char name[100];
    int staffID;
    char position;
};

struct unit {
    staff S;
    char unitName;
    string location;
};
```

Figure Q2a

i) Explain the purpose of declaring structure variable "`S`" for `struct unit`.

(2 marks)

ii) In the `main()` program, show C++ statements to declare an array of structure variable for `struct unit` with the size of 3. Using the declared structure variable, get input for staff name as well as staff ID for 3 workers.

(3 marks)

b) The conversion of an expression to a different type of expression can be performed using **STACK**. By applying its concept, answer the following questions:

i) For the following **PREFIX** expression, produce its **INFIX** equivalent using conversion table.

- * 3 ^ + 4 2 5 1

(5 marks)

ii) Analyze the following **INFIX** expression and produce its **POSTFIX** equivalent using conversion table.

$$A - B * C + D ^\wedge E / F$$

(5 marks)

c) Based on the set of numbers given below, deduce the **Binary Search Tree** with the first number as its root.

56 34 21 89 76 55 4 90 41 88

(5 marks)

QUESTION 3

a) Based on the array declaration as shown below, show the function definition for `void odd_numbers(int n[])` that will display all the odd numbers in the array.

```
int num[12] = {25, 5, 11, 6, 16, 45, 37, 9, 10, 2, 211, 18};
```

(5 marks)

b) Fibonacci series is a set of numbers that starts with 0 and 1, and the next number is the summation of the two previous numbers (0, 1, 1, 2, 3, 5, 8, ….). Based on the sample codes shown in **Figure Q3b**, show a recursive function for `int Fibonacci_number(int num)`.

```
int i, num, first=0, second=1, next;
cout<<"Enter number of terms for Series: ";
cin>>num;
cout<<"Fibonacci series are: \n";
for(i=0; i<num; i++)
{
    cout<<"\n"<<first;
    next = first + second;
    first = second;
    second = next;
}
```

Figure Q3b

(5 marks)

c) Bubble sort is a simple and well-known sorting algorithm. It is used in practice as an introduction to the sorting algorithms because the method is stable and adaptive. By using **BUBBLE** sorting concept, develop the content of the array in ascending order for each iteration for a set of numbers shown in **Figure Q3c**.

25	12	23	7	5

Figure Q3c

(10 marks)

QUESTION 4

a) In data structures, stack and queue are very useful in computer applications.

i) Explain **TWO** (2) main differences between static and dynamic **STACK**.

(4 marks)

ii) Describe the **THREE** (3) main steps performed for traversal algorithm in **LINKED-LIST**. This algorithm will travel from the head node until the last node.

(6 marks)

b)

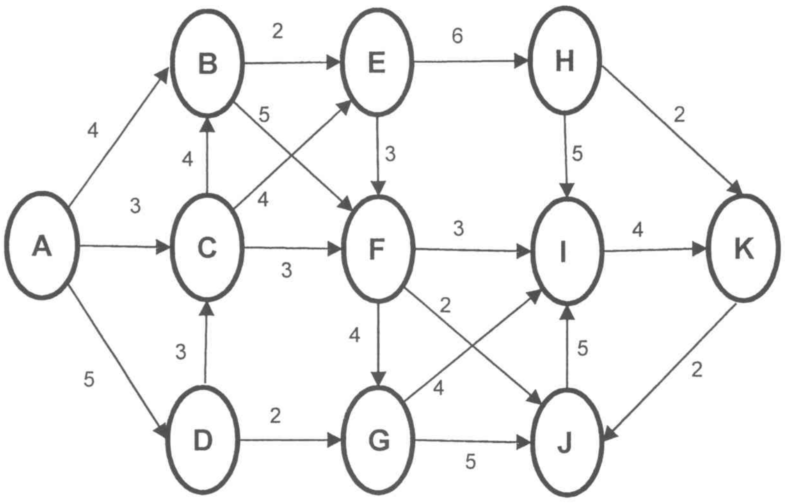


Figure Q4b

Figure Q4b shows a Directed-Acyclic-Graph (DAG) consisting eleven vertices. These vertices represent eleven different locations. The distance between two locations are given as shown in the figure above. Starting from location A, discover the shortest path to all other locations by using Dijkstra's Algorithm.

(10 marks)

QUESTION 5

a) Develop the function definition for insertion sorting algorithm using C++ programming codes based on the following function prototype:

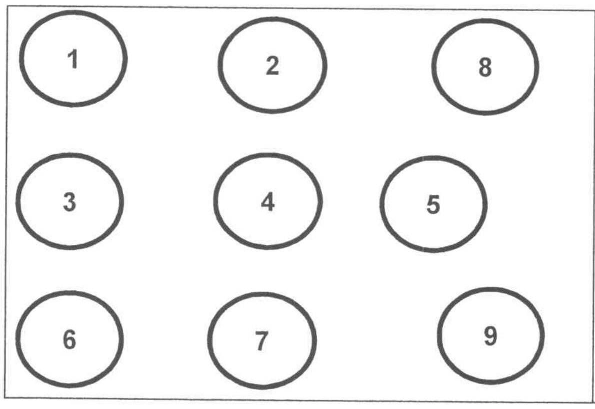
```
void insertionSort(int Array[], int n);
 //n is the number of element in the array
```

(5 marks)

b) A minimum spanning tree (MST) or minimum weight spanning tree is a subset of the edges of a connected, edge-weighted undirected graph. It connects all the vertices together, without any cycles and with the minimum possible total edge weight. Based on the tabulated nodes and adjacency matrix shown in **Figure Q5b,**

i) Draw a complete graph by inserting the edges (undirected graph).

ii) Determine the minimum spanning tree, and calculate the total minimum cost by using **Kruskal's Algorithm**.



$$\begin{pmatrix}
0, & 29, & 24, & 10, & 18, & 0, & 0, & 0, & 0, \\
29, & 0, & 0, & 0, & 19, & 0, & 0, & 14, & 0, \\
24, & 0, & 0, & 19, & 0, & 16, & 0, & 0, & 0, \\
10, & 0, & 19, & 0, & 17, & 18, & 28, & 0, & 0, \\
18, & 19, & 0, & 17, & 0, & 0, & 14, & 23, & 25, \\
0, & 0, & 16, & 18, & 0, & 0, & 23, & 0, & 0, \\
0, & 0, & 0, & 28, & 14, & 23, & 0, & 0, & 25, \\
0, & 14, & 0, & 0, & 23, & 0, & 0, & 0, & 23, \\
0, & 0, & 0, & 0, & 25, & 0, & 25, & 23, & 0,
\end{pmatrix}$$

Figure Q5b

(10 marks)

c) Linear search is a simple searching algorithm. It searches an element in each array by traversing the array from the starting, until the desired element is found. Based on sample codes shown in **Figure Q5c**, develop a function definition for `int Linear_Search(int array_linear[], int key_item, int size);` where `array_linear[ ]` is the set of numbers, `key_item` is the special member that uniquely identifies the item in the data set and `size` is the array size.

```
int arr[] = { 2, 3, 4, 10, 40 };
int x = 10;
int n = sizeof(arr) / sizeof(arr[0]);
int result = Linear_search(arr, x, n);
if (result == -1)
cout<<"Element is not present in array";
else
cout<<"Element is present at index" <<result;
```

Figure Q5c

(5 marks)

END OF QUESTION PAPER