SLIATE
SRI LANKA INSTITUTE OF ADVANCED TECHNOLOGICAL EDUCATION
(Established in the Ministry of Higher Education, vide in Act No. 29 of 1995)

Higher National Diploma in Information Technology
First Year, Second Semester Examination-2017
HNDIT1211: Data Structures and Algorithms

Instructions for Candidates:
Answer four (04) questions only.

No. of questions : 05
No. of pages : 06
Time : Two hours

Question 01

a. What is Data Structure? (03 Marks)

b. State whether the following data structures are leaner or nonlinear. (04 Marks)

i. Tree
ii. Stack
iii. Queue
iv. Graph

c. Briefly explain the following terms. (05 Marks)

i. Abstract Data Type (ADT)
ii. Big O Notation
iii. Best Case Efficiency
iv. Worst Case Efficiency
v. Average Case Efficiency

d. Write C++ code to initialize an array with a word having seven (7) characters and print whether the word is palindrome or not. [NOTE: A palindrome is a word, phrase, number, or other sequence of characters which reads the same backward as forward, such as 'madam' or 'tacocat'] (06 Marks)

e. Write C++ code to create the following two dimensional array and display its elements.

5	3	5
8	4	2
2	6	3
7	4	1

(07 Marks)

Question 02

a. Define Stack Data Structure and mention two (02) ways of implementing Stack. (03 Marks)

b. Graphically illustrate the following stack operations sequentially. (04 Marks)

i. initializeStack()

ii. push(5)

iii. push(10)

iv. a = topEement( )

v. push(2)

vi. push(7)

vii. b = pop()

viii. displayElements()

c. Graphically illustrate the implementation of the following stack operations. (05 Marks)

i. initializeStack()

ii. push(50)

iii. y = topElement( )

iv. push(60)

v. x = pop()

d. Carefully read and understand the following class definition for implementing a stack data structure.

```
const STK_SIZE=5;
class Stack
{
private:
   int top;
   int stk[STK_SIZE];
public:
      ...
      void push(int);
      int pop();
      int topElement();
```

...

}

Now write C++ code to implement the following operations.

i. Adding a new element in on top of the stack. (06 Marks)

ii. Removing the element from the top of the stack.

iii. Referring the top element of the stack without removing it.

e. Carefully observe the following output. Here user input is formatted as bold and underlined.

1.Push 2.Pop 3.Display 4.Exit

Enter your choice: **1**

Enter an element to be pushed: **50**

1.Push 2.Pop 3.Display 4.Exit

Enter your choice: **1**

Enter an element to be pushed: **25**

1.Push 2.Pop 3.Display 4.Exit

Enter your choice: **2**

25 has been popped out.

1.Push 2.Pop 3.Display 4.Exit

Enter your choice: **1**

Enter an element to be pushed: **60**

1.Push 2.Pop 3.Display 4.Exit

Enter your choice: **3**

60 50

1.Push 2.Pop 3.Display 4.Exit

Enter your choice: **4**

Considering that the Stack class has been completely defined, write only the main method in order to obtain the output as shown in the above sample output. (07 Marks)

Question 03

a. How a queue differ from stack in operation and in static implementation? (03 Marks)

b. Diagrammatically illustrate the following Queue Operations. (04 Marks)

i. initializeQueue()

ii. p = isEmpty()

iii. enQueue(A)

iv. enQueue(B)

v. x = deQueuer()

vi. enQueue(C)

vii. y = frontElement()

viii. displayQueue()

c. Diagrammatically illustrate the implementation of the following queue operations. (05 Marks)

i. initializeQueue()

ii. enQueue(l)

iii. enQueue(m)

iv. x = deQueue()

v. enQueue(n)

d. Carefully read and understand the following class definition for the Queue data structure.

```
#include<iostream.h>
const Q_SIZE=5;
class Queue
{
private:
 int front, rear, size;
 int que[Q_SIZE];
public:
 ...
 void enQueue(int);
```

```
int deQueue();
void displayQueue();
...
}
```

Now write code to implement the methods for the following operations. (06 Marks)

i. Inserting an item to the queue

ii. Removing an item from the queue

iii. Displaying all the items in the queue

e. Match the following Linked List Operations with the corresponding Queue operations.

	Linked List Operations		***Queue Operation***
i.	deleteFirstElement()	A	initializeQueue()
ii.	firstElement()	B	enQueue()
iii.	initializeList()	C	deQueue()
iv.	isEmpty()	D	frontElement()
v.	isFull()	E	isEmpty()
vi.	displayList()	F	isFull()
vii.	insertAtRear()	G	displayQueue()

(07 Marks)

Question 04

a. Briefly explain the following terms related to tree data structure with the help of diagrams.

(03 Marks)

i. Tree

ii. Root of a Tree

iii. Parent of a Node

iv. Children of a Node

v. Siblings of a Node

vi. Leaves of a Tree

b. Define the following terminologies related to tree data structure. (04 Marks

i. Size of a Tree

ii. Depth of a Node

iii. Degree of a Tree

iv. Path between two nodes.

c. Explain the following special trees (02 Marks)

i. Binary Tree (03 Marks)

ii. Binary Search Tree

d. Insert the following data set into a binary search tree. (06 Marks)

45, 30, 24, 10, 37, 20, 15, 60, 33, 23

e. Diagrammatically explain the implementation of a Binary Tree. (07 Marks)

Question 05

a. Give a single line explanation for the following terms. (03 Marks)

i) Sorting

ii) Selection Sort

iii) Bubble Sort

b. Sort the data set 5, 1, 4, 2, 8 using Selection Sort Method and show your work (04 Marks)

c. Briefly explain the following terms.

i. Searching (01 Marks)

ii. Sequential Searching (02 Marks)

iii. Binary Searching (02 Marks)

d. Give C++ implementation for Sequential Search. (06 Marks)

e. Write recursive pseudo code for Binary Search. (07 Marks)