HNDIT 1211: Data Structures and Algorithms

Model Answers – 2018 Second Semester

Question 01

a) Briefly explain the term “Data Structure” in computer programming.

- ***Data structure is the arrangement of data in a computer memory/storage,*** (01 mark)
- ***It is the implementation of ADT such as stack, queue, etc***
- ***It helps for efficient programming and reduces complexity of the program and its calculation*** (For any suitable two points 01 mark)

(02 Marks)

b) **Explain** following terms by **using suitable examples**.

I. Primitive Data type

- ***Basic data types of programming language.***
- ***It refers two things: a data item with certain characteristics and permissible operations of data***
- ***Example: integer, float, char, boolean*** (Description -01 mark, example – 01 mark)

II. Abstract Data type

- ***ADT is a specification of a mathematical set of data and the set of operation s that can be performed on the data. (or any other definition gives similar meaning)***
- ***Example: stacks, queue, etc*** (Description -01 mark, example – 01 mark)

(04 Marks)

c) Identify the difference between Linear Data Structures and Non-Linear Data Structures. Give suitable examples for each type.

Linear Data Structure	**Non-Linear Data Structure**
Data organized sequentially (one after another)	Data organized non-sequentially
Easy to implement (Because the computer memory is also organized as linear fashion)	Difficult to implement
Eg: Array, Stack, Queue, Linked List	Eg: Tree, Graph

(For any two comparison points -02 marks
Two examples -02 marks [01 mark per each])

(04 Marks)

d) Fill the blanks by **using suitable words given in the brackets**.

(05 x 01 Marks)

I. ……………………… efficiency explains the minimum number of steps that an algorithm can take with any collection of data values. (***Best Case***, Worst Case, Average Case)

II. If the number of operations in an algorithm is $n^3 + 2n + 10$, the big O' notation of this algorithm is …………….. ($O(1)$, $O(n)$, $O(n^2)$, **$O(n^3)$**)

III. …………………….. is an example for non-linear data structure. (Array, Linked List, **Graph**)

IV. $O(n)$ is faster than ………………. ($O(n)$, **$O(n^3)$**)

V. ………………. has last-in-first-out behavior. (***Stack***, Array, Queue, Tree)

e) Write suitable C++ codes to the followings.

I. Create the following array with the name "**first**" using a **single statement**.

3	20	10	7	11

```
int first [5] = {3,20,10,7,11};  or
int first []= {3,20,10,7,11};
```

(declaration – 01 mark and value assignment -01 mark)

(02 Marks)

II. Declare an integer array with the name "**second**", with the size 5 and without assigning any value.

```
 int second [5];
```

(01 Mark)

III. Write a C++ code segment to assign values **taken as keyboard inputs** to the "**second**" array using a **'for'** loop.

```
for (int x=0; x<=4; x++)
{
        cin>>second[x];
}
```

(for loop – 01 mark and input reading- 01 mark)

(02 Marks)

IV. Write a C++ code to **compare the two arrays** "first" and "second".
Note: If two arrays are similar, you need to display "Arrays are matching" and if not, you need to display "Arrays are not matching"

```
int mismatch=0;
for (int x=0; x<=4 && !mismatch; x++)
        if (first[x] != second[x])
                mismatch=1;
if (mismatch)
        cout<<"Arrays are not matching";
else
        cout<<" Arrays are matching";
```

or any other answer which gives correct result.

(for comparison loop – 03 marks
For statement displaying scenario – 02 marks)

(05 Marks)

Question 02

a) What is Linked List?

- ***Linked list is a <u>linear data structure</u> which <u>consists of nodes which are connected to other nodes</u>.***
- ***Or any other acceptable definition***

(02 Marks)

b) State an advantage of Linked List over Arrays

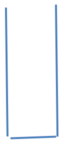
- ***Arrays are having fixed memory allocation. So, once array has declared cannot change the size.***
  But with linked list can have dynamic memory allocation. New nodes can be created or deleted instantly.

(02 Mark)

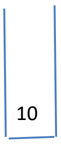
c) Graphically illustrate the following stack operations sequentially.

(03 Marks)

I. initializeStack()

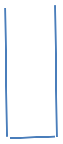


II. push (10)



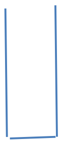
III. a = pop ()

a=10

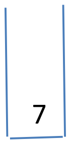


IV. y = isEmpty()

y= true

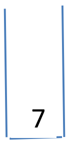


V. push (7)



VI. x = topElement()

x=7

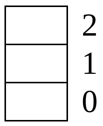


d) Graphically illustrate the static (array based) implementation for the following stack operations.
Note: Size of the array is 3

(06 Marks)

I. initializeStack ( )

Top= -1



II. push (100)

Top= 0

	2
	1
100	0

III. push (50)

Top= 1

	2
50	1
100	0

IV. a= pop ()

a = 50
top=0

	2
	1
100	0

V. push (70); push (80)

Top = 2

80	2
70	1
100	0

VI. b = isFull()

b= true
top=2

80	2
70	1
100	0

e) Consider following stack operations

- isFull ()
- isEmpty ()

I. Write C++ code for the array based (static) implementations of the above operations.

```
int isFull()
{
    if (top == SIZE -1)
        return 1;
    else
        return 0;
}
int isEmpty()
{
    if (top == -1)
        return 1;
    else
        return 0;
}
```

(2 x02 marks=04 Marks)

II. Write C++ code for the linked list based (dynamic) implementation of the above operations.

```
int isFull()
{
      return 0;
}
int isEmpty()
{
      if (top == NULL)
         return 1;
      else
         return 0;
}
```

(2 x 02marks =04 Marks)

f) What do you understand by the term “Stack Overflow”?

- ***Stack over flow is a situation in which a particular computer program tries to use more memory space than the stack is available.***

(02 Marks)

g) Give two examples for applications of stacks.

- ***Undo sequence in a text editor***
- ***Chain of method calls in a programming language***
- ***Reverse operations***

(For any two answers – 02 marks**)**

(02 Marks)

Question 03

a) What is Queue Data Structure?

Queue is a data structure which is used to handle data in first -in-first-out (FIFO) method.

(02 Marks)

b) Give two examples for applications of queues.

- ***CPU Scheduling***
- ***Resource scheduling***

(02 Marks)

c) Graphically illustrate the following scenario using a Queue. Assume that the queue has already initialized and it is empty at the beginning.

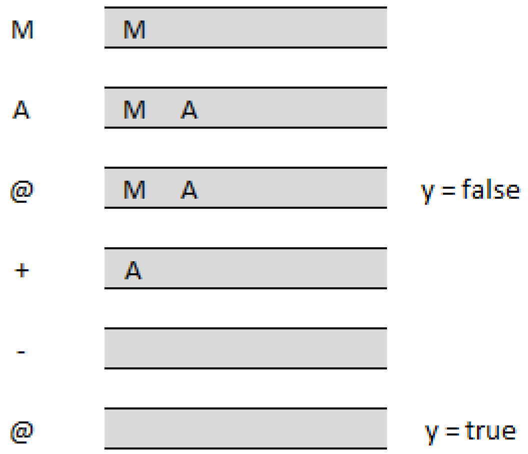
Scenario:

Consider the following sequence.

M A @ + @ - @

In the above sequence;

- Each alphabetic letter inserts the letter into the queue.
- Each operator (+, -, etc.) delete an item from the queue.
- Each @ symbol represents y=isEmpty()



M	M	
A	M A	
@	M A	y = false
+	A	
-		
@		y = true

(06 Marks)

d) Graphically illustrate the static (array based) implementation of the following Queue operations sequentially. **Note: Size of the array is 5.**

(5 x 01 mark = 05 Marks)

I. initializeQueue()

0	1	2	3	4

Front = -1
Rare= -1
Size= 0

II. enQueue (M)

0	1	2	3	4
M				

Front = -1
Rare = 0
Size = 1

III. enQueue (S)

0	1	2	3	4
M	***S***			

Front = -1
Rare = 1
Size = 2

IV. q = isFull()

0	1	2	3	4
M	***S***			

Front =-1
Rare = 1
Size = 2
q=False

V. x = deQueue ()

0	1	2	3	4
	S			

Front =0
Rare = 1
Size = 1
x = M

e) Write down the C++ implementation of a Node which can be used in Dynamic (Linked list) implementation of a Queue.

```
struct Node
{
        int data;
        Node* next;
};
```

(04 Marks)

f) Give static (Array based) implementation of the following queue operations.

I. Insert operation

```
void enQueue(int elt)
{
    if (size <Q_size)                01 mark
    {
      rare= (rare+1)%Q_size;         01 mark
      que[rare]=elt;                 0.5 mark
      size++;                        0.5 mark
    }
}
```

Or any other correct answer

II. Delete operation

```
int deQueue()
{
    if (size>0)                      01 mark
    {
      front =(front +1)%Q_size;      01 mark
      size --;                       0.5 mark
     return que[front];              0.5 mark
    }
}
```

Or any other correct answer

(06 Marks)

Question 04

a) Explain "Tree Data Structure" with a suitable graphical example.

- ***Tree is a hierarchical data structure, which is a set of connected nodes***
- ***Rooted tree has a distinguished node called root*** (any two points – 02 marks)
  (for any suitable tree example diagram – 01 mark)

(03 Marks)

b) Explain how binary search trees are different from binary trees. Use suitable graphical examples.

- ***Binary trees are the trees where no node have more than two children, and children have distinguished as left and right.*** (01 mark)
- ***Binary search trees also have binary tree features as stated above.***
  But BSTs have additional features which do not own by the ordinary binary trees. Such as; (01 mark)
  - ***All keys in N's left sub tree are less than key in N*** (01 mark)
  - ***All keys in N's right sub tree greater than the key in N*** (01 mark)

  Any suitable example of Binary tree – 01 mark and
  Any suitable example of BST – 01 mark

(06 Marks)

c) Briefly explain following terms related to tree data structures.

I. Root – ***the unique node which do not have a parent***
II. Leaf – ***a node which has no children***
III. Size of a tree – ***number of nodes that a tree has***
IV. Depth of a node – ***number of edges from the root to the given node***
V. Degree of a node – ***number of children that owned by the node***
VI. Degree of a tree – ***the maximum degree of any of nodes within the tree***

(If any student uses suitable examples to explain the terms, full marks can be provided for the correct examples)

(6 x 01 marks =06 Marks)

d) Briefly explain the following types of binary trees.

I. Full binary tree – ***a binary tree in which every node other than the leaves has two children***.
II. Complete binary tree – ***a binary tree which is completely filled, with the possible exception of the bottom level, which is filled from left to right***
III. Perfect binary tree – ***binary tree with all leaf nodes at the same level while all internal nodes have degree 2***

(If any student has use only diagrams rather than explanations, can give only half marks)

(3 x 01 mark =03 Marks)

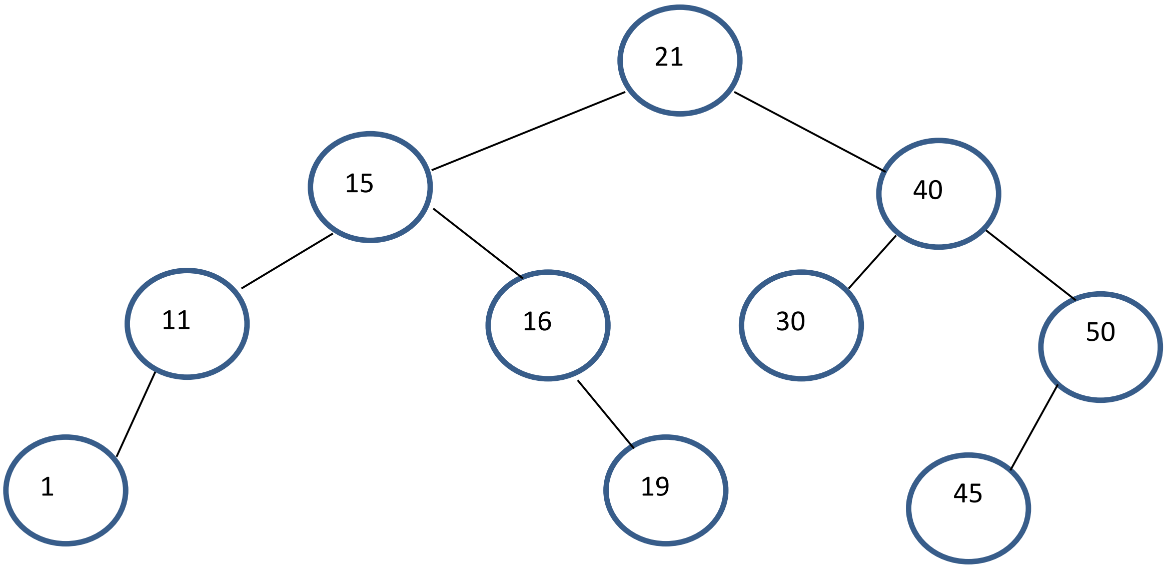
e) State one advantage and one disadvantage of binary trees.

Advantages
- ***Quick search***
- ***Quick insert*** (any one answer – 01 mark)

Disadvantages
- ***Deletion algorithm is complex*** (any one answer – 01 mark)

(02 Marks)

f) Insert following data set in to a binary search tree.
Data Set: 21, 40, 15, 30, 16, 11, 19, 1, 50, 45
(05 Marks)



Question 05

a) What is sorting?
Arranging items in ascending or descending order

(01 Mark)

b) Consider the following data set.
Data Set: 65, 20, 40, 6, 15

I. Sort above data set using selection sort.

Pass 0 (Original Array)

65	*20*	*40*	*6*	*15*

Pass 1 – 01 mark

6	*20*	*40*	*65*	*15*

Pass 2 -01 mark

6	*15*	*40*	*65*	*20*

Pass 3 - 01 mark

6	*15*	*20*	*65*	*40*

Pass 4 – 01 mark

6	*15*	*20*	*40*	*65*

II. Sort the above data set using bubble sort.

Pass 0 (Original Array)

65	*20*	*40*	*6*	*15*

Pass 1 – 01 mark

20	*40*	*6*	*15*	*65*

Pass 2 – 01 mark

20	***6***	***15***	***40***	***65***

Pass 3 – 01 mark

6	***15***	***20***	***40***	***65***

Pass 4 – 01 mark

6	***15***	***20***	***40***	***65***

(2 x 4 Marks)

c) Write a C++ code to implement swap function which can be used in selection sort algorithm.

```
void swap (int *x, int *y)
{
        int t;
        t= (*x);
        *x =(*y);
        *y=t;
}
```

(03 Marks)

d) What do you understand by the term Searching Algorithm?

It is an algorithm which is used to find an item among a collection of items

(01 Mark)

e) Briefly explain following searching algorithms.

I. Sequential Search – ***It examines the first element in the list and then second element and so on until a match is found*** (02 marks)

II. Binary Search – ***this algorithm finds the middle item of a sorted array (in ascending ordered array), compare it against the searched value, then decide which half of the list must contain the searched value, and repeat with that half.*** (02 marks)

f) Write down the pseudo codes for following algorithms.

I. Linear search

```
int sequentialSearch (a[], n, t)
    for i=0 to n-1
    if (a[i]=t)
        return i
    next i
    return -1
```

II. Binary search

```
int binarySearch (a[], l, u , t)
    p= (l+u)/2
    while (a[p] not equal t AND l<=u)
        if (a[p]>t)
                u=p-1
        else
                l=p+1
        p=(l+u)/2
```

```
end while
if (l<=u)
    return p
else
    return -1
```

(2 x 04 Marks)

*** End of the Marking Scheme ***