



NATIONAL SCHOOL OF BUSINESS MANAGEMENT

UGC/UCD//PLY 18.2

1st Year 2nd Semester Examination

22/07/2019

CS106.3 – Data Structures and Algorithms







Instructions to Candidates

- 1) Answer PART A, any three questions from PART B and one question from PART C.
- 2) Time allocated for the examination is three (03) hours.
- 3) Total number of pages - Six (06)
- 4) If a page or a part of this question paper is not printed, please inform the supervisor immediately.
- 5) Write your index number in all pages of answer script.
- 6) Staple all answer sheets at the end of the examination.

PART A: Mandatory Question

- An algorithm can be defined as a detailed step-by-step method for solving a problem so one can relate the techniques to solve day-to-day life problems with them. (20 + 20 = 40 Marks)

I. Identify the following situations and explain applications/ usages of data structures and algorithms. (Transfer the answers to the answer booklet)

Algorithm /Data Structure	Sample	Day-to-day life Application
Linear Search Algorithm		When finding a lost phone checking all places in the room.
1.		2.
3.		4.
5.		6.
7.		8.
9.		10.

- Suppose you have been hired as a detective to investigate a computer related crime. A hacker had stolen important documents and hid it inside his computer. Your task is to find and recover these classified documents. Answer the following questions based on the algorithms and data structures knowledge.

- What type of data structure does the file system represent?

- ii. What are the different mechanisms to read that data structure? Explain your answer with illustrations.
- iii. What type of algorithm can you suggest if these documents were printed and hidden inside the hacker's room? Explain your answer with analysis.

PART B: Answer only 03 (three) Questions

2. In computer science, a sorting algorithm is an algorithm that puts elements of a list in a certain order. The most frequently used orders are numerical order and lexicographical order. **(3 + 5 + 7 = 15 Marks)**

- I. Compare (look for similarities) and contrast (look for difference) a basic searching algorithm (bubble or selection sort) and an advanced searching algorithm (merge or insertion sort).
- II. Sort the [93, 36, 12, 43, 13, 31, 52, 10] array using bubble or selection sort algorithm. Note that the illustrations and labels are mandatory.
- III. Sort the above array using merge or insertion sort algorithm. Note that the illustrations and labels are mandatory.

3. Searching algorithms aim to find position of a target value within an array/list.

(3 + 4 + 8 = 15 Marks)

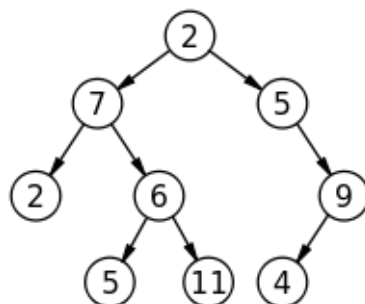
- I. Identify the errors of the below code snippet. Write down the correct statements on the answer booklet.

```
int binarySearch(int array[], int size, int value)
{
    int first = 0, last = size - 1, middle, position = 0;
    boolean found = false;

    while (!found && first <= last)
    {
        middle = (first + last) / 2;
        if (array[middle] = value)
        {
            found = true;
            position = middle;
        }
        else if (array[middle] > value)
            last = middle - 1;
        else
            first = middle -1;
    }
    return position;
}
```

- II. Compare (look for similarities) and contrast (look for differences) linear search and binary search algorithms.

- III. Write a function using pseudo or source codes for searching an integer variable called *item* using linear search in an array called *unorderedArray*.
4. Stacks and queues are linear data structures that allow one to access single data item at a time. **(3 + 3 + 4 + 5 = 15 Marks)**
- Compare (look for similarities) and contrast (look for differences) stacks and queues in terms of adding, removing, size, and emptiness. Note the wording is important; e.g.: *add* \rightarrow *push*
 - Illustrate how would you *push* [7, 6, 3, 5, 9, 0] to a stack data structure.
 - Explain how you would *push* "2" in between "6" and "3."
 - Write pseudo or source code to check if a stack is empty and full.
5. Asymptotic analysis refers to defining the mathematical bound/framing of an algorithm's runtime performance. **(1 + 4 + 10 = 15 Marks)**
- What is the symbolic notation for upper bound of a growth rate?
 - Discuss the "best" and "worst" case scenarios when finding an item using linear search and binary search algorithms.
 - Simplify the following expressions with **reasoning**.
 - $n^4 + 100n^2 + 10n + 50$
 - $10 \cdot n^2 + \log n$
 - $n \cdot \log n + 12 \cdot n^2 + 90,000 \cdot n$
 - $n! + K \cdot n^2$
 - $9000 \cdot n! + 2500 \cdot n^n + 5000 \cdot 2^n$
6. A tree is a widely used abstract data structure that is also non-linear format storing data in a hierarchical structure. **(3 + 6 + 3 + 3 = 15 Marks)**



- Identify the root, siblings, leaves, descendants, ancestors, edges, and paths of the above tree data structure.
- Derive the preorder, post order and in order traversal output of the above tree structure.

- III. Draw a binary search tree (BST) by inserting these numbers from left to right. [19, 33, 27, 8, 12, 36, 45, 2, 9]
- IV. With the help of the above BST, illustrate how you can find the item "2." (Hint: just like binary search mechanism)

PART C: Answer only 01 (one) question.

7. A recursive algorithm is an algorithm that calls itself with "smaller (or simpler)" input values. **(3 + 4 + 8 = 15 Marks)**

- I. Name three situations where you can use recursive approach to solve mathematic curiosities?
- II. Find the greatest common divisor (GCD) of 45 and 180 using prime factorization and Venn diagrams.
- III. In mathematics, factorials can be defined as $n! = n \cdot (n-1)!$ where $0! = 1$. The following code snippet is a factorial implementation in C language using the recursive approach. Write a similar program to compute GCD. You need to start with the mathematical formulas. (Hint: Euclidian theorem)

```
long factorial(int n)
{
    // base case 0! = 1! = 1
    if (n <= 0)
        return 1;
    return(n * factorial(n-1));
}
```

8. A recursive algorithm is an algorithm that calls itself with "smaller (or simpler)" input values. **(7+ 8 = 15 Marks)**

- I. Derive the output of the following code snippets. For each code block illustrate how the activation records create/remove at the memory stack.

b.

```
void A(int a){
    if(a>0){
        printf("%d", a);
        A(a-1);
    }
}
void main(){
    int x=3;
    A(x);
}
```

a.

```
void A(int a){
    if(a>0){
        A(a-1);
        printf("%d", a);
    }
}
void main(){
    int x=3;
    A(x);
}
```

- II. The Fibonacci sequence is, by definition, the integer sequence in which every number after the first two is the sum of the two preceding numbers.

To simplify:

0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, ... It has many applications in mathematics and even trading.

- a) Derive a recursive function to find Fibonacci value of a given integer
- b) Show the function call tree structure if $n=5$ and derive the answer (Fibonacci value).