Usman Institute of Technology

Department of Computer Science – Fall 2018

CS-212 Data Structures and Algorithms
Lab Manual # 1

OBJECTIVE:

1. Introduction to Data Structures and Classification of Data Structures.
2. To understand Linear Array.
3. Develop method to take input from user in an array
4. Traversing in an Array
5. To find an element in an array.
6. To understand multidimensional Array
7. Implementation of the algorithm for finding largest element in two dimensional array.
8. Implementation of the algorithm for matrix multiplication using two dimensional arrays.

.

Name : _____

Roll No. : _____

Semester : _____ Section: _____

Date : _____

Remarks : _____

Signature : _____

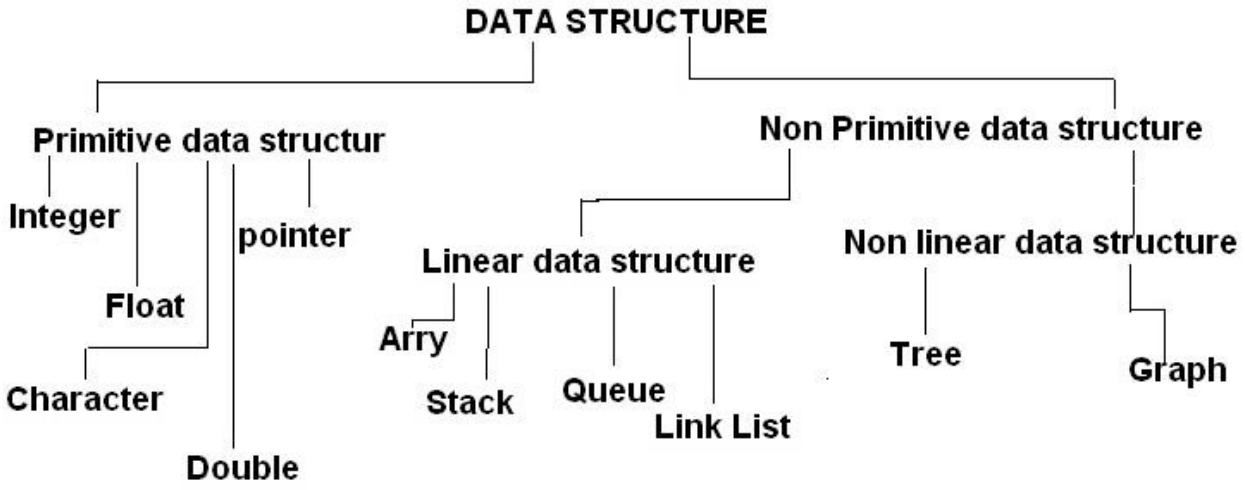
Experiment No. 01

Background

Data Structure:

Data may be organized in many different ways: the logical or mathematical model of a particular organization of data is called a data structure.

Classification of Data Structures:



Linear Data Structures:

A data structure is said to be linear if its elements form a sequence or a linear list. In linear data structures, the data is arranged in a linear fashion.
Example: Arrays, linked lists, stacks and queues.

Arrays:

The simplest type of data structure is a linear (or one-dimensional) array. By a linear array, we mean a list of a finite number n of similar data elements referenced respectively by a set of n consecutive numbers, usually 1, 2, 3, ..., n.

If we choose the name A for the array, then the elements of A are denoted by subscript notation i.e. A_1, A_2, A_3 or by the parenthesis notation i.e. A(1), A(2), A(3), …., A(N) or by the bracket notation A[1], A[2], A[3], ..., A[N].

Regardless of the notation, the number K in A[K] is called a subscript and A[K] is called a subscripted variable.

Operations performed on Linear Array:

The operations one normally performs on any linear structure. Whether it be an array or a linked list, include the following:

(a) **Traversal:** Processing each element in the list.
(b) **Search:** Finding the location of the element with a given value or the record with a given key.
(c) **Insertion:** Adding a new element to the list,
(d) **Deletion:** Removing an element from the list.
(e) **Sorting:** Arranging the elements in some type of order.
(f) **Merging:** Combining two lists into a single list.

Algorithm 1: (Traversing a linear array)

Here, A is a linear array with lower bound (LB) and upper bound (UB). This algorithm traverses A applying an operation PROCESS to each element of A.

Step I: [Initialize counter] Set I:= LB.
Step II: Repeat steps 3 and 4 while I <= UB:
Step III: Apply PROCESS to A. [Visit element.]
Step IV: Set I := I+1. [Increase counter.]
[End of step 2 loop.]
Step V: Exit.

Sample Program in C#: Array Initialization and Traversal

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApplication5
{
    class Program
    {


        static void Main(string[] args)
        {
            int MAX = 10;
             int[] numbers = new int[MAX]; // declare numbers as an int array of any size
            Methods mm = new Methods();
            mm.writeArray(numbers,MAX);
            mm.PrintArray(numbers);
        }
    }
```

```
public class Methods
 {

     public void writeArray(int[] arr, int n)
     {

         for (int i = 0; i < n; i++)
         {
             arr[i] = Convert.ToInt16(Console.ReadLine());
         }
}
public void PrintArray(int[] arr)
{
    for (int i = 0; i < arr.Length; i++)
    {
        System.Console.Write("{0}\n",arr[i]);
    }
    System.Console.WriteLine();
    System.Console.ReadKey();
}


  }
}
```

Sample Program in JAVA: Array Initialization and Traversal

```
class Testarray{
public static void main(String args[]){

int a[]=new int[5];//declaration and instantiation
a[0]=10;//initialization
a[1]=20;
a[2]=70;
a[3]=40;
a[4]=50;

//traversal array
for(int i=0;i<a.length;i++)//length is the property of array
System.out.println(a[i]);
            }
}
```

Algorithm 2 :(Searching)

A non-empty array DATA with N numerical values is given. This algorithm finds the location LOC of the searching value X. The variable K is used as counter.

Step 1. [Initialize] set K: =1, LOC: =1 and X: = [Element to find]
Step 2. Repeat Step 3 and 4 while K less than and equal to N
Step 3. If X==DATA [K], then Set LOC: =K [Go to step 5 and Exit]
Step 4. Set K: =K+1
Step 5. Write LOC, X
Step 6. Exit

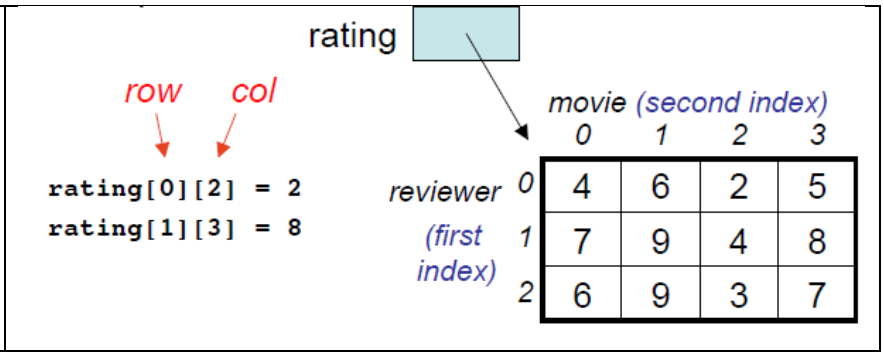
Multidimensional Arrays:

Two-Dimensional Arrays:

Two-dimensional (2D) arrays are indexed by two subscripts, one for the row and one for the column.

Example:

- Movie ratings by multiple reviewers.
- Each row is a different reviewer
- Each column is a different movie



Algorithm 3: (Traversing a 2D-array.)

Here A is a two – dimensional array with M rows and N columns. This algorithm traverses array A and applies the operation PROCESS to each element of the array.

1. Repeat For I = 1 to M
2. Repeat For J = 1 to N
3. Apply PROCESS to A[I][J]
   [End of Step 2 For Loop]
   [End of Step 1 For Loop]
4. Exit

Sample Program: 2D Array Initialization and Traversal

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace Lab2Example
{
    class Program
    {
        static void Main(string[] args)
        {
            int [,] abc = new int[2,2];

            Console.WriteLine("Insert Elements in 2D Array (2X2 array)\n");
                    for (int row = 0; row < 2; row++)
                    {
                        for (int col = 0; col < 2; col++)
                        {
                            abc[row, col] = Convert.ToInt16(Console.ReadLine());
                        }
                    }
```

```
            Console.WriteLine("\n============  Output =========\n");

            for (int i = 0; i<2; i++)
                for (int j=0; j<2; j++)
                Console.WriteLine("{0} \n", abc[i,j]);

            System.Console.WriteLine();
            System.Console.ReadKey();
        }
    }
}
```

ALGORITHM FOR MATRIX MULTIPLICATION USING 2D ARRAYS.

Algorithm 4:
MATMUL (A, B)
Here A and B both are of order nxn.
1. For i = 0 to n-1
2. For j = 0 to n-1
3. C[i][j] = 0
4. For k = 0 to n-1
5. C[i][j] = C[i][j] +A[i][k]*B[k][j]
6. End for
7. End for
8. End for
9. Return

FINDING LARGEST ELEMENT IN 2D ARRAY

Algorithm 5:
MAX is a two – dimensional array with M rows and N columns.
X is the element to search
This algorithm finding largest element in array MAX and applies the operation (Searching) PROCESS to each element of the array.
[Initialize] Set LOC: =1, Data [2][2] = {Some values} and MAX: = DATA [0][0]
1. Repeat For I = 1 to M
2. Repeat For J = 1 to N
3. If MAX<DATA [I][J], then Set MAX: = DATA [I][J]
   [End of Step 2 For Loop]
   [End of Step 1 For Loop]
4. Write MAX
5. Exit

Tasks:

1. Implement algorithm 1 and show the output.
2. Implement above algorithm 2 and show the output.
3. Implement Algorithm 3 and show the output
4. Implement Algorithm 4 and show the output
5. Implement Algorithm 5 and show the output.
6. Modify Algorithm 5 to find the smallest number in 2D array

Home Task: Implement all algorithms in Object Oriented structure using JAVA and C++ programming language.