

“Start by doing what's necessary; then do what's possible; and suddenly you are doing the impossible”- Francis of Assisi

DATA STRUCTURE

Type Arrays [1D, 2D]

Deadline **IN CLASS**

Weighting TBA

LAB PERFORMANCE

1A

OBJECTIVES

This assessment item is designed to test your skills on single and multi-dimensional arrays and loops.

OUTCOMES

After this Lab, you should have clear concept on the following topics:

- **Array**
- **One dimensional/ 1D Array**
 - i. When to use.
 - ii. Understanding the index of any elements.
 - iii. Data Storing in and Extracting from 1D array.
 - iv. Passing an entire array to a function.
 - v. Finding minimum and maximum values.
 - vi. Searching a particular element from the array.
 - vii. Perform different operations both on data and indices.
- **Two dimensional/ 2D Array**
 - i. When to use.
 - ii. Understanding the index of any elements in 2D array.
 - iii. Data Storing in and Extracting from 2D array.
 - iv. Passing an entire 2D array to a function.
 - v. Finding minimum and maximum values.
 - vi. Searching a particular element from the array.
 - vii. Perform different operations both on data and indices.

PRELIMINARIES

- Passing an entire 1D array to a function.

<pre>// function declaration: void printArray(int arr[], int size); int main () { // an int array with 5 elements. int balance[5] = { 1000, 2, 3, 17, 50}; // pass pointer to the array as an argument. printArray (balance, 5); return 0; }</pre>	<pre>void printArray (int arr[], int size) { int i; for (i = 0; i < size; i++) { cout<<arr[i]<<"\n"; } }</pre>
--	---

- Passing an entire 2D array to a function.

```
int array[10][10];
void passFunc(int a[][10])
{
    // ...
}
passFunc(array);
```

ATTENTIONS

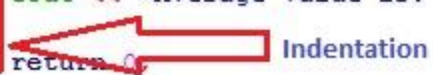
- **Use meaningful variable's name.**
- **Maintain indentation properly.**

```
int main ()
{
    // an int array with 5 elements.
    int balance[5] = {1000, 2, 3, 17, 50};
    double avg;

    // pass pointer to the array as an argument.
    avg = getAverage( balance, 5 ) ;

    // output the returned value
    cout << "Average value is: " << avg << endl;

    return 0;
}
```



- **Don't copy-paste.**

SCENARIO

Suppose, you are a **Jr. Software Engineer** at ABC solutions, one of the leading software firm in the country. At present, they are working in a software project whose objective is to develop a Virtual University Management System. As you are a fresh graduate, so they have given you the task to develop the quiz marking system as the part of the whole system. The requirements of the quiz marking system are given below:

1. **First, you have to store and display the result of the quizzes. Let's assume there are 10 students in each section. You have to implement the program only for one section and one quizzes at this point. Which data structure would you use?**

- Implement the program with your proposed data structure with 10 (integer/float) values. Display/print the content of your data structure. Also display the data structure in reverse order.

Input: A []: 0 1 2 3 4 5 6 7 8 9

Output 1: A []: 0 1 2 3 4 5 6 7 8 9

Output 2: A []: 9 8 7 6 5 4 3 2 1 0

2. **After publishing the initial result of the quiz, the teacher may want to add some bonus marks after giving some extra tasks to the students. So you have to extend your previously written program to provide this functionality.**

- Add a function in your existing program which will take **the array** and **bonusMarks** as the arguments. Display the marks of the students after adding the bonusMarks.
- ```
void addBonus(int marks[], int arraySize, int bonusMarks)
{
 //Implement your code here
}
```

*Input:* A []: 0 1 2 3 4 5 6 7 8 9

*Input:* bonusMarks= 3

*Output:* A []: 3 4 5 6 7 8 9 10 11 12

3. **So far, you have done well! Now, to enrich this software we have to add more functionalities. At this stage, finding the highest and lowest marks in the quiz would be highly appreciable.**

- Let's add two functions to display the highest and lowest number in the quiz separately.

*Input:* A []: 0 1 2 3 4 5 6 7 8 9

*Output 1:* Highest Marks is: 9

*Output 2:* Lowest Marks is: 0

4. **After the adding/dropping period, a new student can be added into a section. So you have to provide the functionality to add the marks of a student in any position of the data structure you have used.**

- Now enhance your existing program by adding a function which can be used to insert a value at the given position in the data structure.
  - Suppose an array is initialized as: 22 12 13 15 10 17
  - Insert value 21 in middle ( $K^{\text{th}}$  position) of array
  - Print the array to see the changes

*Sample Execution:*

*Enter the position to insert: 4*

*Enter the value to insert: 21*

*22 12 13 21 15 10 17*

- 5. After the adding/dropping period or midterm, a new student can drop the section of a course. So you have to provide the functionality to remove the marks of a student from the data structure you have used.**

- Now enhance your existing program by adding a function which can be used to remove a value from the given position in the data structure.
  - Suppose an array is initialized as: 22 12 13 15 10 17
  - Delete value 12 from ( $K^{\text{th}}$  position) of array
  - Print the array to see the changes

*Sample Execution:*

*Enter the position to delete: 4*

*22 13 21 15 10 17*

## **Bonus Problems**

- 6. Find the highest and lowest value from a 2D array and display their location.**
- 7. Show the sum of the boundary elements and diagonal elements for a 5\*5 matrix.**