SVKM's NMIMS
Mukesh Patel School of Technology Management & Engineering
Computer Engineering Department
Program: B.Tech. Sem II

Course: Data Structure and Algorithm
List of Experiments

LAB Manual

PART A

(PART A : TO BE REFFERED BY STUDENTS)

Experiment No.01

A.1 Aim:

Introduction to Data Structures and implementation of Arrays

A.2 Prerequisite:

1. Knowledge of different operations performed on Array data structure
2. Fundamental concepts of C\C++.

A.3 Outcome:

After successful completion of this experiment students will be able to

1. Identify the need of appropriate selection of data structure
2. Identify the steps of Array data structure selection.
3. Implement various applications on Array.

A.4 Theory:

A.4.1. Introduction of Array

- An array is a collection of similar data elements.
- These data elements have the same data type.
- The elements of the array are stored in consecutive memory locations and are referenced by an index (also known as the subscript).
- The subscript is an ordinal number which is used to identify an element of the array.
- An array must be declared before being used. Declaring an array means specifying the following:

- *Data type*—the kind of values it can store, for example, int, char, float, double.
- *Name*—to identify the array.
- *Size*—the maximum number of values that the array can hold.

- **Traversing an Array**
- Traversing an array means accessing each and every element of the array for a specific purpose.
- Traversing the data elements of an array A can include printing every element, counting the total number of elements, or performing any process on these elements.
- Since, array is a linear data structure (because all its elements form a sequence), traversing its elements is very simple and straightforward. The algorithm for array traversal is given in Fig.

```
Step 1: [INITIALIZATION] SET I = lower_bound
Step 2: Repeat Steps 3 to 4 while I <= upper_bound
Step 3:       Apply Process to A[I]
Step 4:       SET  I = I + 1
        [END OF LOOP]
Step 5: EXIT
```

- **Inserting an Element in an Array**
- If an element has to be inserted at the end of an existing array, then the task of insertion is quite simple.
- We just have to add 1 to the upper_ bound and assign the value.
- Here, we assume that the memory space allocated for the array is still available.
- For example, if an array is declared to contain 10 elements, but currently it has only 8 elements, then obviously there is space to accommodate two more elements.
- But if it already has 10 elements, then we will not be able to add another element to it.

```
Step 1: Set upper_bound = upper_bound + 1
Step 2: Set A[upper_bound] = VAL
Step 3: EXIT
```

- Figure shows an algorithm to insert a new element to the end of an array.
- In Step 1, we increment the value of the upper_bound.
- In Step 2, the new value is stored at the position pointed by the upper_bound.
- **Algorithm to Insert an Element in the Middle of an Array**

- The algorithm INSERT will be declared as INSERT (A, N, POS, VAL).
- The arguments are
- A, the array in which the element has to be inserted
- N, the number of elements in the array
- POS, the position at which the element has to be inserted
- VAL, the value that has to be inserted

```
Step 1: [INITIALIZATION] SET I = N
Step 2: Repeat Steps 3 and 4 while I >= POS
Step 3:         SET A[I + 1] = A[I]
Step 4:         SET I = I - 1
        [END OF LOOP]
Step 5: SET N = N + 1
Step 6: SET A[POS] = VAL
Step 7: EXIT
```

- **Deleting an Element from an Array**
- Deleting an element from an array means removing a data element from an already existing array. If the element has to be deleted from the end of the existing array, then the task of deletion is quite simple.
- We just have to subtract 1 from the upper_bound. Figure shows an algorithm to delete an element from the end of an array.

```
Step 1: SET upper_bound = upper_bound - 1
Step 2: EXIT
```

- **Algorithm to delete an element from the middle of an array**
- The algorithm DELETE will be declared as DELETE(A, N, POS).
- The arguments are:
- A, the array from which the element has to be deleted
- N, the number of elements in the array
- POS, the position from which the element has to be deleted

```
Step 1: [INITIALIZATION] SET I = POS
Step 2: Repeat Steps 3 and 4 while I <= N - 1
Step 3:         SET A[I] = A[I + 1]
Step 4:         SET I = I + 1
        [END OF LOOP]
Step 5: SET N = N - 1
Step 6: EXIT
```

A.5 Procedure/Algorithm:

A.5.1:

TASK 1:

A click counter is a small hand-held device that contains a push button and a count display. To increment the counter, the button is pushed and the new count shows in the display. Clicker counters also contain a button that can be pressed to reset the counter to zero. Design and implement the Counter ADT that functions as a hand-held clicker.

TASK 2:

Write a C/C++ program of array to perform following (1D Array)

i. Find the sum and Average of all the elements.

ii. Find highest and lowest element in an array.

iii. Insert and delete an element in an array (by passing array to the function).

PART B

(PART B : TO BE COMPLETED BY STUDENTS)

(Students must submit the soft copy as per following segments within two hours of the practical. The soft copy must be uploaded on the Blackboard or emailed to the concerned lab in charge faculties at the end of the practical in case the there is no Black board access available)

Roll No. N230	Name: Daksh Gehlot
Class : MBA Tech CE (Sem 2)	Batch : A1
Date of Experiment: 21 January 2022	Date of Submission: 6 February 2022
Grade :	Time of Submission: 02:22 AM
Date of Grading:	

B.1 Software Code written by student: (Task 1)

(Paste your code completed during the 2 hours of practical in the lab here)

Task1:

```cpp
#include <iostream>
using namespace std;
#include <string.h>

// Daksh Gehlot - N230
int main()
{
    int click[100], count=1, i=0, x=0;
    char choice;
    cout << string(20, '*');
    cout << "\nA Click Counter\n";
    cout << string(20, '*');
    cout << "\nPress P to increment the counter."
    "\nPress R to reset the counter."
    "\nPress E to exit the counter."
    "\n\nCurrent count: 0\n";
    cout << "\nEnter your choice: ";
    cin >> choice;

    while(true){
        if (choice == 'P' || choice == 'p'){
            click[x] = count++;
            cout << "Current count: " << click[x] << endl;
            x++;
        }
        else if (choice == 'R' || choice == 'r'){
            cout << "Last count: " << click[x-1] << endl;
            for(i=0;i<x;i++){
                click[i] = 0;}
            x=i=0;
            cout << "Counter reset to " << click[i] << endl;
        }
        else if (choice == 'E' || choice == 'e'){break;}
        else{
            cout << "Enter a valid choice";
        }

        cout << "Enter new choice: ";
        cin >> choice;
    }
    return 0;
}
```

Task2:

(i)

```cpp
#include <iostream>
using namespace std;

// Daksh Gehlot - N230
int main()
{
    int n, sum=0, avg;
    cout << "\nHow many elements: ";
    cin >> n;

    int arr[n];
    cout << "\nEnter the elemets: ";
    for(int i=0;i<n;i++){
        cin >> arr[i];
        sum += arr[i];
    }
    avg = sum / n;
    cout << "\nThe sum of the elements are: " << sum;
    cout << "\nAverage is: " << avg;
    return 0;
}
```

(ii)

```cpp
#include <iostream>
using namespace std;

// Daksh Gehlot - N230
int main()
{
    int n, i, min, max;
    cout << "\nHow many input items are there? ";
    cin >> n;
    int a[n];

    cout << endl;
    for(i=0;i<n;i++){
        cout << "Enter element " << i+1 <<": ";
        cin >> a[i];}

    cout << "\nThe input array is: ";
    for(i=0;i<n;i++){cout << a[i] << " ";}

    max=a[0],min=a[0];
    for(i=0;i<n;i++){
        if (a[i]>max){max=a[i];}
        else if (a[i]<min){min=a[i];}}

    cout << "\nThe min and max value from the array is " <<
    min << " and " << max << " respectively.";

    return 0;
}
```

(iii)

```cpp
#include <iostream>
using namespace std;

// Daksh Gehlot - N230
int insertIntoArray(int newArr[], int x){
    int num;
    cout << "\nEnter the element to add: ";
    cin >> num;
    newArr[x] = num;
    cout << "\nThe new array is: ";
    for(int i=0;i<x+1;i++){
        cout << newArr[i] << " ";
    }
}

int delFromArray(int newArr[], int x){
    int num;
    cout << "\nEnter the element to delete: ";
    cin >> num;

    for(int i=0;i<x+1;i++){
        if(newArr[i] != num){
            cout << newArr[i] << " ";
        }
    }
}

int main()
{
    int n, i;
    cout << "\nEnter the number of elements: ";
    cin >> n;

    int arr[n+1];
    cout << "\nEnter the elemets: ";
    for(i=0;i<n;i++){
        cin >> arr[i];
    }

    cout << "\nThe entered array is: ";
    for(i=0;i<n;i++){
        cout << arr[i] << " ";
    }

    insertIntoArray(arr, n);
    delFromArray(arr, n);
    return 0;
}
```

B.2 Input and Output: (Task 1)

(Paste your program input and output in following format, If there is error then paste the specific error in the output part. In case of error with due permission of the faculty extension can be given to submit the error free code with output in due course of time. Students will be graded accordingly.)

Task1:

```
********************
A Click Counter
********************
Press P to increment the counter.
Press R to reset the counter.
Press E to exit the counter.

Current count: 0

Enter your choice: P
Current count: 1
Enter new choice: p
Current count: 2
Enter new choice: p
Current count: 3
Enter new choice: P
Current count: 4
Enter new choice: p
Current count: 5
Enter new choice: R
Last count: 5
Counter reset to 0
Enter new choice: E
```

Task2:

(i)

```
PS C:\NMIMS\SEM 2\DSA\Codes> cd "c:\NMIMS\SEM

How many elements: 7

Enter the elemets: 1 2 3 4 5 6 7

The sum of the elements are: 28
Average is: 4
PS C:\NMIMS\SEM 2\DSA\Codes\Arrays-A1>
```

(ii)

```
PS C:\NMIMS\SEM 2\DSA\Codes\Arrays-A1> cd "c:\NMIMS\SEM 2\DSA\Co

How many input items are there? 7

Enter element 1: 4
Enter element 2: 3
Enter element 3: 2
Enter element 4: 7
Enter element 5: 13
Enter element 6: 2
Enter element 7: 8

The input array is: 4 3 2 7 13 2 8
The min and max value from the array is 2 and 13 respectively.
PS C:\NMIMS\SEM 2\DSA\Codes\Arrays-A1>
```

(iii)

```
PS C:\NMIMS\SEM 2\DSA\Codes> cd "c:\NMIMS\SEM

Enter the number of elements: 7

Enter the elemets: 1 2 3 4 5 6 7

The entered array is: 1 2 3 4 5 6 7
Enter the element to add: 8

The new array is: 1 2 3 4 5 6 7 8
Enter the element to delete: 3
1 2 4 5 6 7 8
PS C:\NMIMS\SEM 2\DSA\Codes\Arrays-A1>
```

B.3 Observations and learning [w.r.t. all tasks]:

Task 1: Learnt about incrementing operation and its mechanism with the example of click counter device example which can be done using while loop and if else statements.

Task 2: Learnt about finding the sum and average of all the elements. How to find highest and lowest element in an array. Insert and delete an element in an array. And to perform all the above into one program which can be done using for loop and if else statements and also switch case to give user choice to perform a particular task

B.4 Conclusion:

C++ provides a data structure, the array, which stores a fixed-size sequential collection of elements of the same type. An array is used to store a collection of data, but it is often more useful to think of an array as a collection of variables of the same type. We can use normal variables (v1, v2, v3, ...) when we have a small number of objects, but if we want to store a large number of instances, it becomes difficult to manage them with normal variables. The idea of an array is to represent many instances in one variable.

****