**Q1. If a one dimensional integer type array with its size and length given, write the code in C++**

**language to create functions to perform the following operations (Please assume whatever**

**is necessary to exemplify the results) :**

**i. Display()**

**ii. Add/Append(x)**

**iii. Insert(index,x)**

**iv. Delete(x)**

**v. LinearSearch(s)**

**vi. Get(index); function to get value available on the given index**

**vii. Set(index,x); insert the value x at the given index**

**viii. Max()**

**ix. Min()**

**x. Reverse()**

**xi. Shift()**

**xii. Rotate(int number )**

**PROGRAM:**

#include<iostream>

using  namespace std;

void Display(int arr[], int n)

{

    for(int i=0;i<n;i++)

    {

        cout<<arr[i]<<" ";

    }

    cout<<endl;

}

void reverseArray(int arr[], int start, int end)

{

    while (start < end)

    {

        int temp = arr[start];

        arr[start] = arr[end];

        arr[end] = temp;

        start++;

        end--;

    }

}

int Append(int x[],int n,int y)

{

    n++;

    x[n-1]=y;

    Display(x,n);

    return n;

}

int linearSearch(int x[],int n,int n1)

{

    for(int i=0;i<n;i++)

    {

        if(n1==x[i])

        {

            cout<<"Element found and is at index no. "<<i;

            return 1;

            break;

        }

    }

    return 0;

}

void Get(int x[],int y,int n)

{

    if(y<n)

    cout<<"The value at index "<<y<<" is "<<x[y]<<endl;

    else

    cout<<"index number is invalid\n";

}

int Max(int arr[], int n)

{

    int max = arr[0];

    int i;

    for (i = 1; i < n; i++)

        {

            if (arr[i] > max)

           {max = arr[i];

           }

        }

    return max;

}

void Min(int arr[], int n)

{

    int min = arr[0];

    for(int i =1;i<n;i++)

   {

       min=arr[i]<min?arr[i]:min;

   }

   cout<<"Minimum value of the array is "<<min<<endl;

}

void shift(int x[],int n)

{

    int last = x[n-1];

    for(int i=n-2;i>=0;i--)

    {

        x[i+1]=x[i];

    }

    x[0]=last;

}

int Insert(int arr[],int n,int y,int index)

{

    if(index>=n)

    {

        cout<<"INVALID INDEX\n";

        return n;

    }

    n++;

    for(int i=n-2;i>=index;i--)

    {

        arr[i+1]=arr[i];

    }

    arr[index]=y;

    Display(arr,n);

    return n;

}

int Delete(int arr[],int n,int index)

{

    if(index>=n)

    {

        cout<<"INVALID INDEX\n";

        return n;

    }

    for(int i =index;i<n-1;i++)

    {

        arr[i]=arr[i+1];

    }

    n--;

    Display(arr,n);

    return n;

}

void rotate(int arr[],int n,int x)

{

    int i = 1;

    while(i<=x)

    {

        shift(arr,n);

        i++;

    }

}

int main()

{

    int arr[] = {1, 2, 3, 4, 5, 6};

    int n = sizeof(arr) / sizeof(arr[0]);

    //Display

    Display(arr, n);

    //Append

    cout<<"Enter the number you wish to append\n";

    int z;

    cin>>z;

    n = Append(arr,n,z);

    cout<<endl;

    cout<<"After adding a term the array as "<<n<<" elements"<<endl;

    Display(arr,n);

    //Linear Search

    int e;

    cout<<"Enter the element you want to find \n";

    cin>>e;

    int check = linearSearch(arr,n,e);

    if(!check)

    cout<<"Element not found\n";

    cout<<endl;

    //Get

    int n1;

    cout<<"Enter the index at whose position you want to find the element \n";

    cin>>n1;

    cout<<endl;

    Display(arr,n);

    Get(arr,n1,n);

    //Max

    int max=Max(arr,n);

    cout<<"Maximum element of the array is "<<max<<endl;

    cout<<endl;

    Display(arr,n);

    //Min

    Min(arr,n);

    cout<<endl;

    Display(arr,n);

    //Reverse

    reverseArray(arr, 0, n-1); // since address of the array is sent, direct changes in the array are made.

    cout<<endl;

    cout << "Reversed array is" << endl;

    Display(arr,n);

    shift(arr,n);

    cout<<endl;

    Display(arr,n);

    //Insert

    int num;

    cout<<"Enter the number which you want to insert\n";

    cin>>num;

    int pos;

    cout<<"Enter the index at which you want to insert the num"<<endl;

    cin>>pos;

    n = Insert(arr,n,num,pos);

    cout<<endl;

    //Delete

    cout<<"Enter the index number whose element you want to delete\n";

    cin>>pos;

    n=Delete(arr,n,pos);

    cout<<endl;

    Display(arr,n);

     cout<<endl;

    //Rotate

    int num1;

    cout<<"Enter the number of times you want the array to get rotated\n";

    cin>>num1;

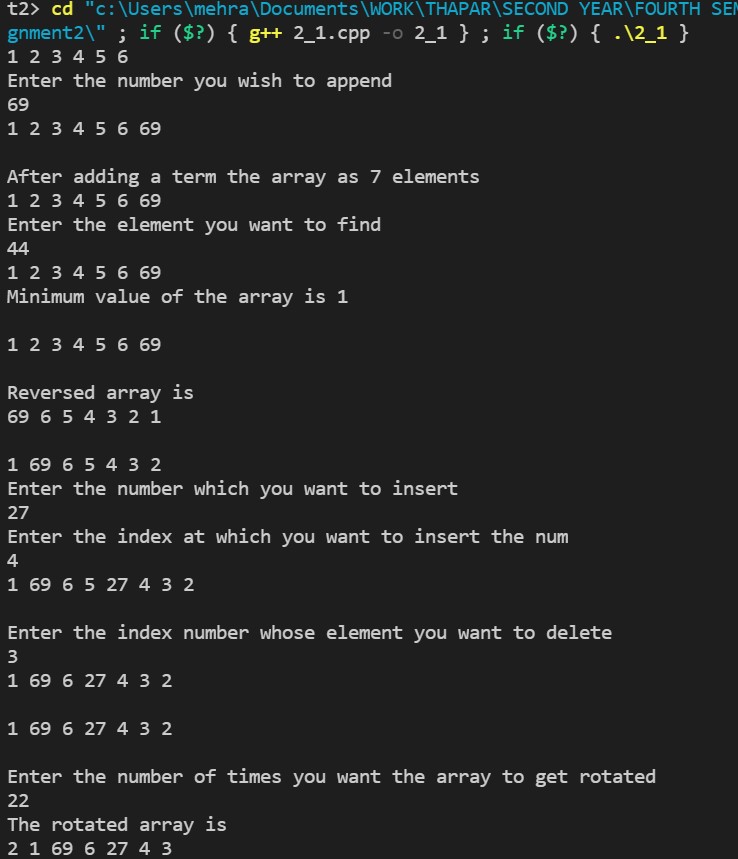
    rotate(arr,n,num1);

    cout<<"The rotated array is\n";

    Display(arr,n);

    return 0;

}

****

**Q2. For a given array, write functions to perform the following:**

**i. Check if an array is sorted**

**ii. Merge arrays**

**iii. Set operations on array: Union, Intersection**

**PROGRAM**

#include<iostream>

using namespace std;

bool check\_sorted(int x[],int n)

{ int y;

    for(int i =0;i<n-1;i++)

    {

        if(x[i]<x[i+1])

        y=1;

        else

         {

             y=0;

             break;

         }

    }

    return y;

}

int\* merge(int x[],int y[],int arr3[],int n1,int n2,int n3)

{

    for(int i=0;i<n1;i++)

    {

        arr3[i]=x[i];

    }

    for(int i=n1,j=0;i<(n3);i++,j++)

    {

        arr3[i]=y[j];

    }

    return arr3;

}

int remove\_duplicate(int arr[],int n)

{

    if(n==1||n==0)

    {return n;}

    int temp[n];

    int j=0;

    for(int i=0;i<n-1;i++)

    {

        if(arr[i]!=arr[i+1])

        {temp[j++]=arr[i];}

    }

        temp[j++]=arr[n-1];// this step is done to add the last element of the array to temp array

        //we feed the elements stored in the temp array back to the original one

        for(int i=0;i<j;i++)

        {

            arr[i]=temp[i];

        }

    return j;// this returns the value of the number of terms in the array without duplicate elements

}

int unsorted\_duplicate(int arr[],int n)

{

    int i,j,k,key,exists;

    int arr\_2[100]={0};

        for(i =0,j=0;i<n;i++)

        {

            key = arr[i];

            exists=0;

            for(int k =0;k<i;k++)

            {

                if(arr\_2[k]==key)

                {

                    exists=1;

                }

            }

            if(!exists)

            {

                arr\_2[j]=arr[i];

                j++;

            }

        }

    n=j;

    for(int z=0;z<n;z++)

    {

        arr[z]=arr\_2[z];

    }

    return n;

}

int\* sort(int x[],int n)

{

    for(int i=0;i<n-1;i++)

    {

        for(int j=0;j<n;j++)

        {

            if(x[j]>x[j+1])

            {

                int temp;

                temp=x[j];

                x[j]=x[j+1];

                x[j+1]=temp;

            }

        }

    }

    return x;

}

void union\_operation(int x[],int y[],int z[],int n1,int n2,int n3)

{

    int\* merged\_array = merge(x,y,z,n1,n2,n3);

    int\* sorted\_arr = sort(merged\_array,n3);

    n3 = remove\_duplicate(merged\_array,n3);

    for(int i=0;i<n3;i++)

    {

        cout<<\*(sorted\_arr+i)<<" ";

    }

}

void intersection(int input1[], int input2[], int size1, int size2)

{

    /\* Don't write main().

     \* Don't read input, it is passed as function argument.

     \* Print the output and don't return it.

     \* Taking input is handled automatically.

     \*/

    int i,j;

    for(i=0;i<size1;i++){

     for(j=0;j<size2;j++){

         if(input1[i]==input2[j]){

             cout<<input1[i];

             cout<<endl;

             input2[j]=-9999999999999999999;

             break;

         }

     }

     }

}

int main()

{

    int arr1[] = {23,78,9,11,3};

    int n1 = sizeof(arr1)/sizeof(arr1[0]);

    int arr2[] = {1,2,3,4,5,9};

    int n2 = sizeof(arr2)/sizeof(arr2[0]);

//checking if sorted

    int check = check\_sorted(arr1,n1);

    if(check==1)

    cout<<"Array 1 is sorted\n";

    else

    cout<<"Array 1 is not sorted\n";

    check = check\_sorted(arr2,n2);

    if(check==1)

    cout<<"Array 2 is sorted\n";

    else

    cout<<"Array 2 is not sorted\n";

//merging of array

    int n3=n2+n1;

    int arr3[n3];

    int\* merged\_arr = merge(arr1,arr2,arr3,n1,n2,n3);

    cout<<"The merged array is \n";

    for(int i=0;i<n3;i++)

    {

        cout<<\*(merged\_arr+i)<<" ";

    }

    cout<<endl;

//union operation

    int n4=n1+n2;

    int arr4[n3];

    cout<<"Union of the two arrays is\n";

    union\_operation(arr1,arr2,arr4,n1,n2,n4);

//intersection operation

    int n5=n1+n2;

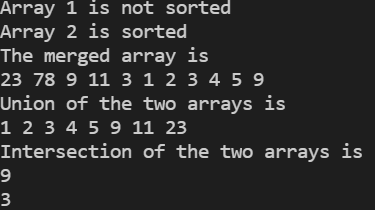
    int arr5[n5];

    cout<<"\nIntersection of the two arrays is\n";

    intersection(arr1, arr2, n1, n2);

    return 0;

}

****

**Q3. For a given array, write functions to perform the following:**

**i. Finding single element in an array**

**ii. Finding multiple elements in an array**

**iii. Finding duplicates in a sorted array;**

**(Here, need to first insert sorted array and then perform operation)**

**iv. Finding duplicates in an unsorted array;**

**(Here, the array need not to be sorted)**

**v. Finding a pair of elements with sum k**

**vi. Finding max and min in a single scan; here you should use only single loop to**

perform both the operations

PROGRAM

#include<iostream>

using namespace std;

void Display(int arr[],int n)

{

    for(int i=0;i<n;i++)

    {

        cout<<arr[i]<<" ";

    }

}

int linearSearch(int x[],int n,int n1)

{

    for(int i=0;i<n;i++)

    {

        if(n1==x[i])

        {

            cout<<"Element found and is at index no. "<<i;

            return 1;

            break;

        }

    }

    return 0;

}

int remove\_duplicate(int arr[],int n)

{

    if(n==1||n==0)

    {return n;}

    int temp[n];

    int j=0;

    for(int i=0;i<n-1;i++)

    {

        if(arr[i]!=arr[i+1])

        {temp[j++]=arr[i];}

    }

        temp[j++]=arr[n-1];// this step is done to add the last element of the array to temp array

        //we feed the elements stored in the temp array back to the original one

        for(int i=0;i<j;i++)

        {

            arr[i]=temp[i];

        }

    return j;// this returns the value of the number of terms in the array without duplicate elements

}

int\* sort(int x[],int n)

{

    for(int i=0;i<n-1;i++)

    {

        for(int j=0;j<n;j++)

        {

            if(x[j]>x[j+1])

            {

                int temp;

                temp=x[j];

                x[j]=x[j+1];

                x[j+1]=temp;

            }

        }

    }

    return x;

}

void find\_max\_min(int arr[],int n)

{

    int max = arr[0], min=arr[0];

    int i,j;

    for(i=1;i<n;i++)

    {

        max = arr[i]>max?arr[i]:max;

        min = arr[i]>min?min:arr[i];

    }

    cout<<"\nThe maximum value in the array is "<<max<<endl;

    cout<<"The minimum value in the array is "<<min<<endl;

}

void pair\_sum(int arr[],int n, int sum)

{

    for(int i =0;i<n-1;i++)

    {

        for(int j=i+1;j<n;j++)

        {

            if(arr[i]+arr[j]==sum)

            {

                cout<<"Sum of "<<arr[i]<<" and "<<arr[j]<<" is equal to "<<sum<<endl;

                return ;

            }

        }

    }

    cout<<"No pair exits to give the value that you want\n";

}

int unsorted\_duplicate(int arr[],int n)

{

    int i,j,k,key,exists;

    int arr\_2[100]={0};

        for(i =0,j=0;i<n;i++)

        {

            key = arr[i];

            exists=0;

            for(int k =0;k<i;k++)

            {

                if(arr\_2[k]==key)

                {

                    exists=1;

                }

            }

            if(!exists)

            {

                arr\_2[j]=arr[i];

                j++;

            }

        }

    n=j;

    for(int z=0;z<n;z++)

    {

        arr[z]=arr\_2[z];

    }

    return n;

}

int main()

{

    int arr[] = {11,8,8,2,3,12,1,1,8,8,8,8,8,1,1,19,21,2,2,11,11,3,3,19,21};

    int n = sizeof(arr)/sizeof(arr[0]);

    //finding a single element

    cout<<"Enter the element you want to find \n";

    int e;

    cin>>e;

    int check = linearSearch(arr,n,e);

    if(!check)

    cout<<"Element not found\n";

    cout<<endl;

    for(int i=0;i<n;i++)

    {

        cout<<arr[i]<<" ";

    }

    cout<<endl;

    //Removing elements from the sorted array

    int\* sorted\_arr = sort(arr,n);

    n = remove\_duplicate(sorted\_arr,n);

    for(int i=0;i<n;i++)

    {

        cout<<\*(sorted\_arr+i)<<" ";

    }

    cout<<endl;

    //remove duplicate elements from unsorted array

    cout<<"\nRemoving duplicates from unsorted array\n";

    //Since the above array is already sorted, take a new array

    int arr1[] = {11,8,8,2,3,12,1,1,8,8,8,8,8,1,1,19,21,2,2,11,11,3,3,19,21};

    int n1 = sizeof(arr)/sizeof(arr[0]);

    n1 = unsorted\_duplicate(arr1,n1);

    Display(arr1,n1);

    cout<<endl;

    //findind the maximum and minimum

    find\_max\_min(arr,n);

    int sum;

    cout<<"enter the value that you would like to see as the sum of the elements of the array\n";

    cin>>sum;

    pair\_sum(arr,n,sum);

    return 0;

}

