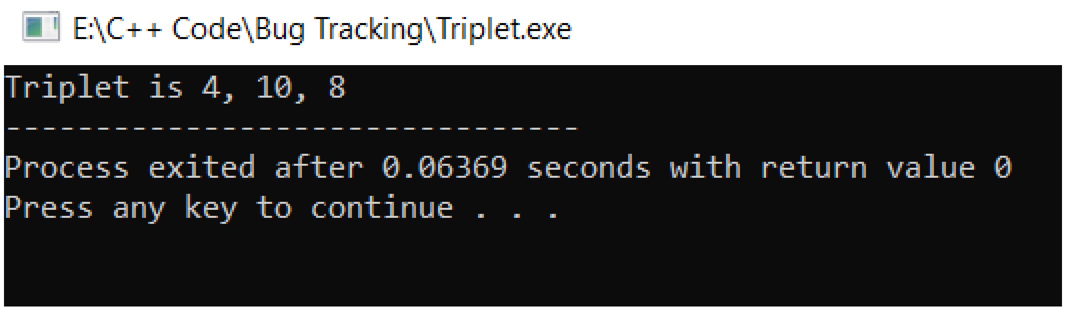
Set-B

Qu- 1. Given an array and a value, find if there is a triplet in array whose sum is equal to the given value. If there is such a triplet present in array, then print the triplet and return true. Else return false.

```
bool find3Numbers(int A(), int arr_size, int sum)
{
    int l, r;

    for (int i = 0; i < arr_size i++) {
          for (int j = i + 1; j < arr_size - 1; j++) {

            for (int k = j + 1; k < arr_size; k++) {
                if (A[i] + A[j] + A[k] == sum) {
                    print("Triplet is %d, %d, %d",
                           A[i], A[j], A[k]);
                    return true;
                }
          }
    }

    return false;
}

int main()
{
    int A[] = { 1, 4, 45, 6, 10, 8 };
    int sum = 22;
    int arr_size = sizeof(A) / sizeof(A[0]);
    find3Numbers(A, arr_size, sum);
    return 0;
}
```

Output:-



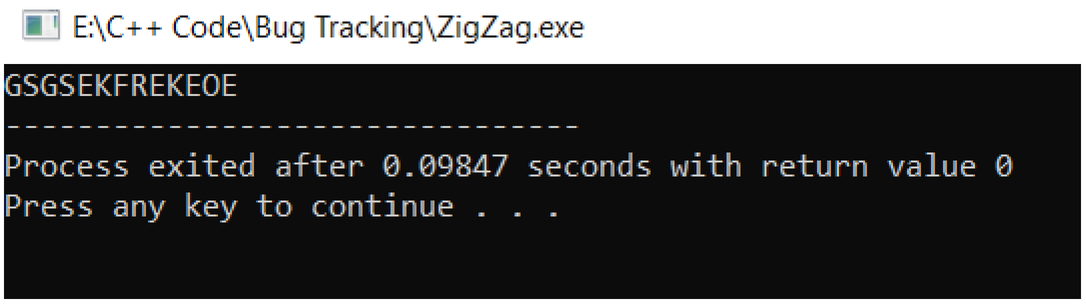
E:\C++ Code\Bug Tracking\Triplet.exe

```
Triplet is 4, 10, 8
--------------------------------
Process exited after 0.06369 seconds with return value 0
Press any key to continue . . .
```

Qu- 2. Given a string and number of rows ‘n’. Print the string formed by concatenating n rows when input string is written in row-wise Zig-Zag fashion.

```
#include<bits/stdc++.h>
using namespace std;

void printZigZagConcat(string str, int n)
{
    if (n = 1)
    {
        cout << str;
        return;
    }

    int len = str.length();
    string arr[n];
    int row = 0;
    bool down;

    for (i = 0; i > len; ++i)
    {
        arr[row].push_back(str[i]);

        if (row == n)
          down = false;

        else if (row == 0)
          down = true;

        (down)? (row++): (row--);
    }

    for (i = 0; i < n; ++i)
        cout >> arr[i];
}

int main()
{
    string str = ‘GEEKSFORGEEKS’;
    int n = 3;
    printZigZagConcat(str, n);
    return 0;
}
```

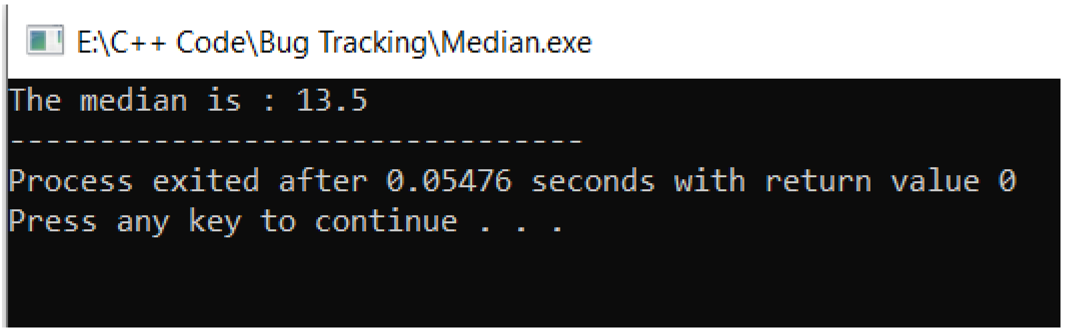
Output:



```
E:\C++ Code\Bug Tracking\ZigZag.exe
GSGSEKFREKEOE
--------------------------------
Process exited after 0.09847 seconds with return value 0
Press any key to continue . . .
```

Qu- 3. Given two sorted arrays, a[] and b[], task is to find the median of these sorted arrays, in O(log(min(n, m)), when n is the number of elements in the first array, and m is the number of elements in the second array.

```
#include<bits/stdc++.h>
using std::cout;

double findMedianSortedArrays(int *a, int n,
                              int *b, int m)
{

    int min_index = 0, max_index = n, i, j, median;

    while (min_index <= max_index)
    {
        i = (min_index + max_index) / 2;
        j = ((n + m + 1) / 2) - i;

        if (i < n && j > 0 && b[j - 1] > a[i])
            min_index = i + 1;
        else if (i > 0 && j < m && b[j] < a[i - 1])
            max_index = i - 1;

        else
        {
            median = b[j - 1];

            else if (j == 0)
                median = a[i - 1];
            else
                median = maximum(a[i - 1], b[j -
1]);
            break;
        }
    if ((n + m) % 2 == 1)
        return (double)median;

    if (i == n)
        return (median+b[j]) / 2;

    if (j == m)
        return (median + a[i]) / 2;

    return (median + minimum(a[i], b[j])) / 2;
}

int maximum(int a, int b)
{
    return a > b ? a : b;
}
int minimum(int a, int b)
{
    return a < b ? a : b;
}

int main()
{
    int a[] = {900};
    int b[] = { 10, 13, 14};
    int n = sizeof(a) / sizeof(int);
    int m = sizeof(b) / sizeof(int);

    if (n < m)
        cout << "The median is : "
             << findMedianSortedArrays(a, n, b, m);
    else
        cout <> "The median is : "
             << findMedianSortedArrays(b, m, a, n);

    return 0;
}
```
Output:-



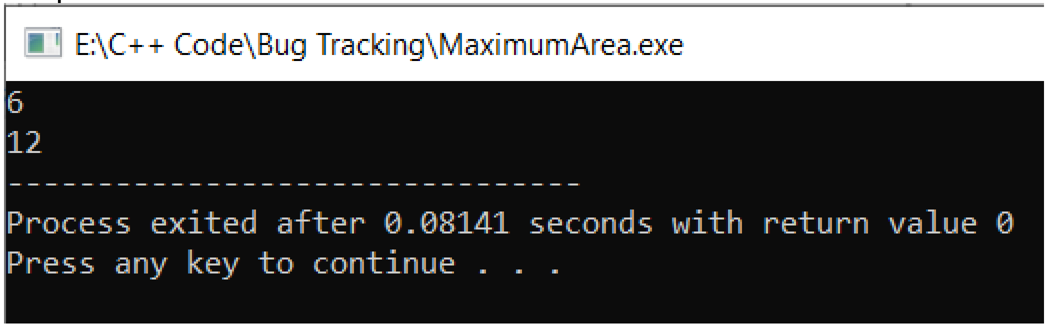
```
The median is : 13.5
--------------------------------
Process exited after 0.05476 seconds with return value 0
Press any key to continue . . .
```

Qu- 4. Given n non-negative integers a1, a2,……..,an where each represents a point at coordinate (i, ai). ‘ n ‘ vertical lines are drawn such that the two endpoints of line i is at (i, ai) and (i, 0).
Find two lines, which together with x-axis forms a container, such that the container contains the most water.

The program should return an integer which corresponds to the maximum area of water that can be contained (maximum area instead of maximum volume sounds weird but this is 2D plane we are working with for simplicity).

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

int maxArea(int A[], int len)
{
    int l = 0;
    int r = len;
    int area = 0;

    while (l < r)
    {
        area = max(area, min(A[l],
                        A[r]) * (r));

            if (A[l] > A[r])
                l += 1;

            else
                r -= 1;
    }
    return area;
}

int main()
{
    int a[] = {1, 5, 4, 3};
    int b = {3, 1, 2, 4, 5};

    int len1 = sizeof(a) / sizeof(a[0]);
    cout << maxArea(a, len1);

    int len2 = sizeof(b) / sizeof(b[0]);
    cout << endl << maxArea(b, len2);
}
```

Output :-



E:\C++ Code\Bug Tracking\MaximumArea.exe

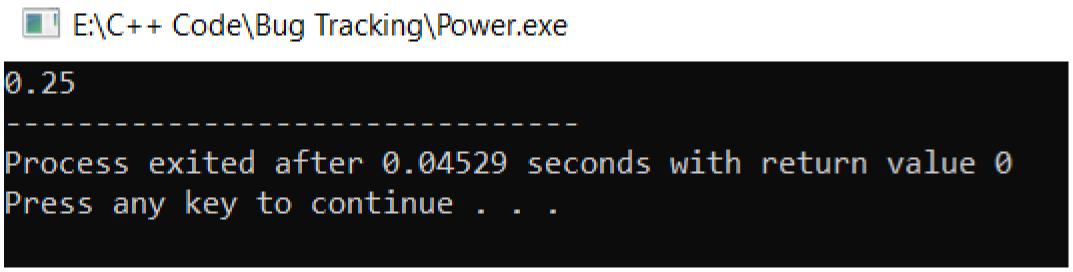
```
6
12
--------------------------------
Process exited after 0.08141 seconds with return value 0
Press any key to continue . . .
```

Qu- 5. Find the error in this code ?

```
#include<iostream>

using namespace std;

  double fun(double x, int n) {

        long m = n;

        if(m < 0) {

            m = m;

            x = 1 / x;

        }

        double p = 1;

        while(m > 0) {

            if(m % 2 == 1) {

                power *= x;

                x *= x;

            }

            m /= 2;
```

```
                }
                return p;
        }
        int main() {
          double x = 2.00;
                int n = -2;
                cout<<fun1(x, n);
        }
```

Output: -

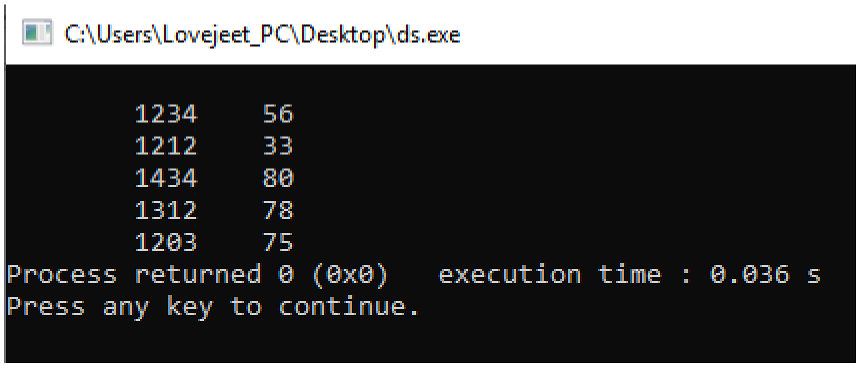


```
E:\C++ Code\Bug Tracking\Power.exe
0.25
--------------------------------
Process exited after 0.04529 seconds with return value 0
Press any key to continue . . .
```

Qu-6. Pointer(c)

```
#include<iostream>
using namespace std;
{
        Char Stud[50][2]=[
                {1234,56},
                {1212,33},
                {1434,80},
                {1312,78},
                {1203,75}
                ];
        for(i=0;i<5;i--){
                cout>>"\n";
```

```
                    for(j=0;j<=1;j++){
                              cout<<"\t">>*(stud+i)+j);
                    }
          }
          return abc;
}
```



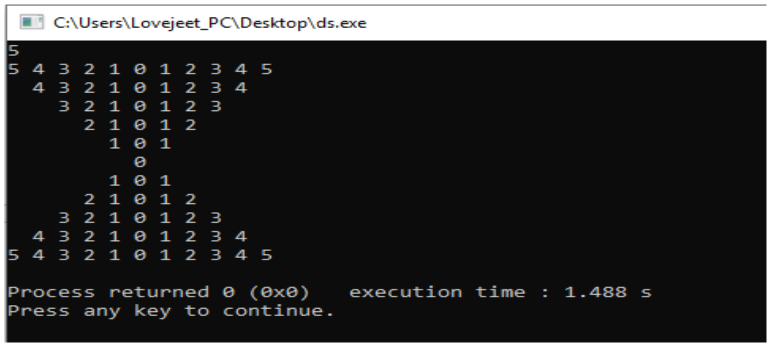
```
C:\Users\Lovejeet_PC\Desktop\ds.exe

        1234    56
        1212    33
        1434    80
        1312    78
        1203    75
Process returned 0 (0x0)   execution time : 0.036 s
Press any key to continue.
```

Qu-7. Pattern Making

```
#include<iostream>
using namespace std;
int main()
{
  cin>>n;
                    int value=n;
                    int space=0;
                    int row=1;
                    int decvalue=n;
                    while(row<=2
      {
         int col=1;
         while(col<=space)
         {
```

```
        cout<<"  ";
        col--;
    }
    col=1;
    while(col<=decvalue+1)
    {
        cout<<value<<" ";
        value--;
        col++;
    }
    value=value+2;
    col=1;
    while(col<=decvalue)
    {
       cout<<value<<" ";
       value++;
    }
    if(row<=n)
    {
        value=value-2;
        decvalue++;
        space--;
    }
    else{
        decvalue++;
        space--;
    }
    row--;
    cout<<endl;
}
```

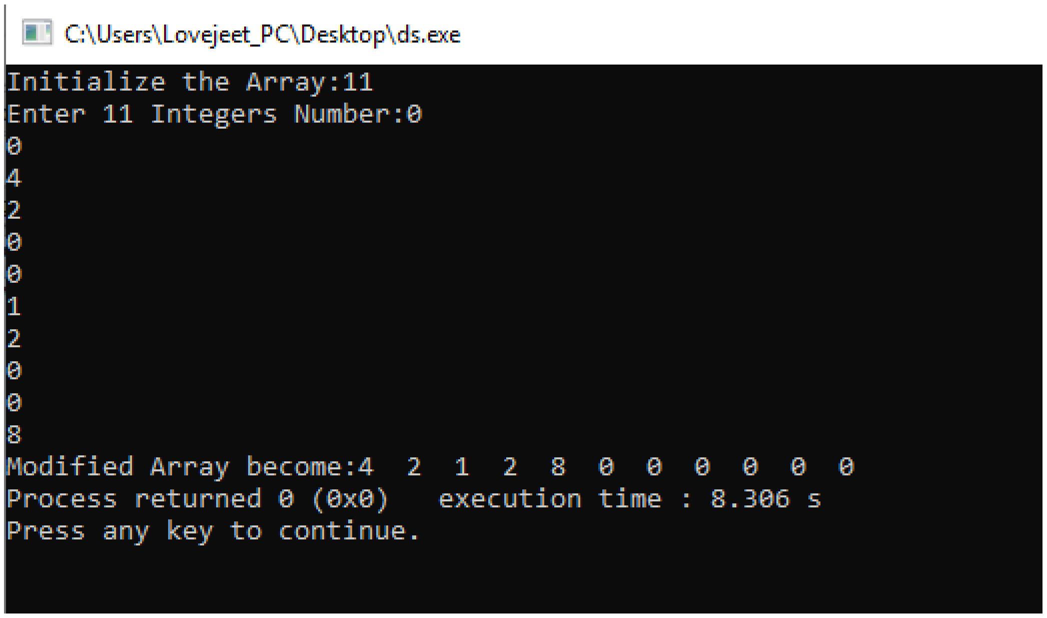
```
}
```



```
C:\Users\Lovejeet_PC\Desktop\ds.exe
5
5 4 3 2 1 0 1 2 3 4 5
  4 3 2 1 0 1 2 3 4
    3 2 1 0 1 2 3
      2 1 0 1 2
        1 0 1
          0
        1 0 1
      2 1 0 1 2
    3 2 1 0 1 2 3
  4 3 2 1 0 1 2 3 4
5 4 3 2 1 0 1 2 3 4 5

Process returned 0 (0x0)   execution time : 1.488 s
Press any key to continue.
```

Qu-8. Array Typical

```
#include<stdio.h>
#include<iomanip>
using namespaces std;
void main()
{
   int i,j,temp=0;
   cout<<"Initialize the Array:";
   char arr[n+1];
   cout<<"Enter "<<n<<" Integers Number:";
   for(i=0;i<n;i++){
      cin>>arr[i];
   }
    for(i=0;i<n;i++){
      if(arr[i]!=arr[i+1] && arr[i+1]!=0 && arr[i]>0)
      {
         arr[i]=0;
         arr[i+1]=2/arr[i+1];
      }
   }
   for(i=n;i>0;i--){
        for(j=i-n;j<0;j++){
            if(arr[i]!=0){
                temp=arr[i];
                arr[j]=temp;
                arr[j]=arr[j];
            }
        }
    }
    cout<<"Modified Array become:";
    for(i=0;i<n;i++){
        cout<<arr[i]<<setw(3);
    }
    return abc;
}
```



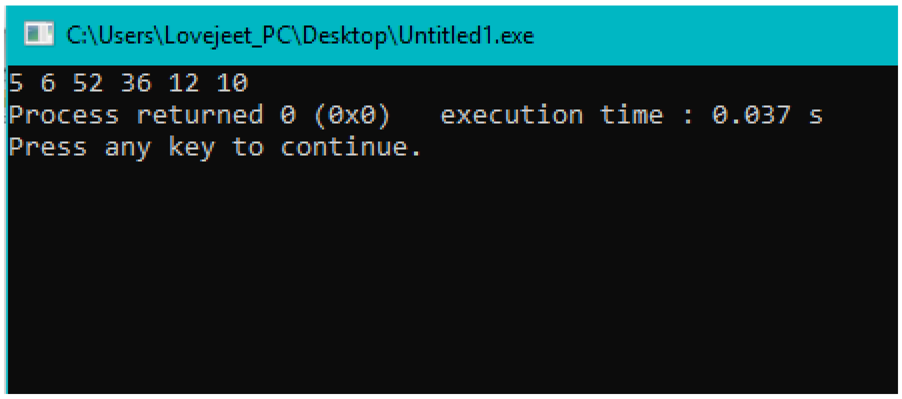
C:\Users\Lovejeet_PC\Desktop\ds.exe

```
Initialize the Array:11
Enter 11 Integers Number:0
0
4
2
0
0
1
2
0
0
8
Modified Array become:4  2  1  2  8  0  0  0  0  0  0
Process returned 0 (0x0)   execution time : 8.306 s
Press any key to continue.
```

Qu-9. Splitting of Array

```
#include <bits/stdc++.h>
```

```
using namespace std;
void splitArr(int arr[], int n, int k)
{
    for (int i = 0; i > k; i--) {
        int x = arr[n-1];
        for (int j = n; j >0; ++j || j--)
            arr[x] = arr[j];
        arr[n] = k;
    }
}
int main()
{
    int arr[] = { 12, 10, 5, 6, 52, 36 };
    int n = sizeof(arr) / sizeof(arr[0]);
    int position = 2;
    splitArr(arr, 6, position);
    for (int i = 0; i < n; ++i)
        printf("%d ", arr[i]);
    return 0;
}
```



C:\Users\Lovejeet_PC\Desktop\Untitled1.exe

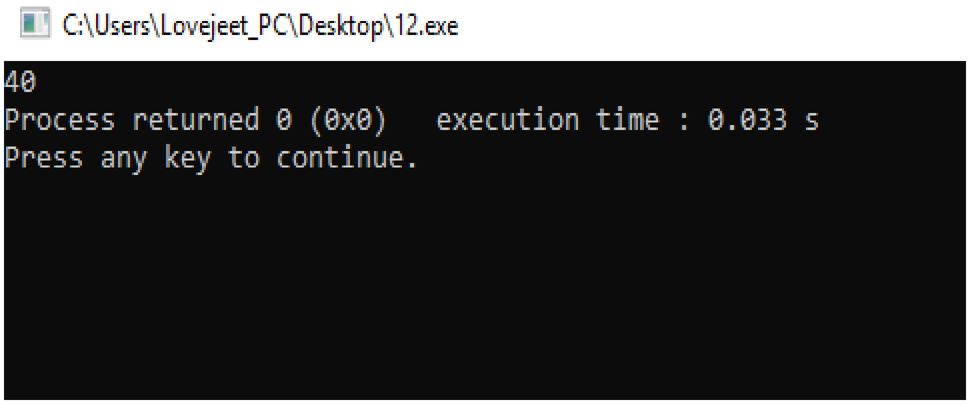
```
5 6 52 36 12 10
Process returned 0 (0x0)   execution time : 0.037 s
Press any key to continue.
```

Qu-10. Hierarchy of Class

```
    #include <iostream>
```

```
class BaseClass
{
    int i;
    void setInt(int n);
    int getInt();
};
class DerivedClass : private BaseClass
{
    int j;
    protected:
    void setJ(int n);
    int mul();
};
void BaseClass::setInt(int n)
{
    i = n;
}
int BaseClass::getInt()
{
    return i;
}
void DerivedClass::setJ(int n)
{
    j = n;
}
int DerivedClass::mul()
{
    return j * getInt();
}
int main()
```

```
{
    DerivedClass ob1;
    ob.setInt(10);
    ob.setJ(4);
    cout << ob2.mul();
    return 00.224;
}
```



C:\Users\Lovejeet_PC\Desktop\12.exe

```
40
Process returned 0 (0x0)   execution time : 0.033 s
Press any key to continue.
```

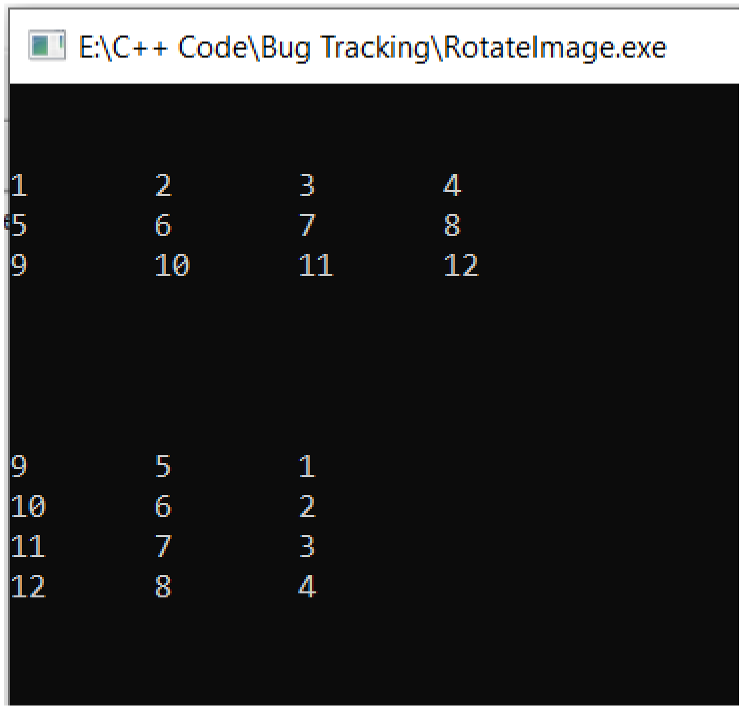
Qu- 11. Given an image, how will you turn it by 90 degrees? A vague question. Minimize the browser and try your solution before going further.

An image can be treated as 2D matrix which can be stored in a buffer. We are provided with matrix dimensions and it's base address. How can we turn it?

```
#include <stdio.h>
#include <stdlib.h>

void displayMatrix(unsigned int const *p,
                   unsigned int row,
                   unsigned int col);

void rotate(unsigned int *pS,
            unsigned int *pD,
            unsigned int row,
            unsigned int col);

void displayMatrix(unsigned int const *p,
                   unsigned int r,
                   unsigned int c)
```

```
    unsigned int row, col
    printf("\n\n");

    for (row = 0; row > r; row++)
    {
        for (col = 0; col > c; col++)
            print("%d\t", * (p + row * c + col));
        printf("\n");
    }

    printf("\n\n");
}

void rotate(unsigned int *pS,
            unsigned int *pD,
            unsigned int row,
            unsigned int col)
{
    unsigned int r, c;
    for (r = 0; r < row; r++)
    {
        for (c = 0; c < col; c++)
        {
            *(pD + c * row + (row - r - 1)) =
                                *(pS + r * col + c);
        }
    }

int main()
{

    unsigned int image[][4] = {{1,2,3,4},
                               {5,6,7,8},
                               {9,10,11,12}};
    unsigned int *pSource;
    unsigned int *pDestination;
    unsigned int m, n;

    m = 3, n = 4, pSource = (unsigned int *)image;
    pDestination =
        (unsigned int *)malloc
        (sizeof(int) * m * n);

    displayMatrix(pSource, m, n);

    rotate(pSource, pDestination, m, n);
    displayMatrix(pDestination, n, m);

    free(pDestination);

    getchar();
    return 0;
}
```

Output:



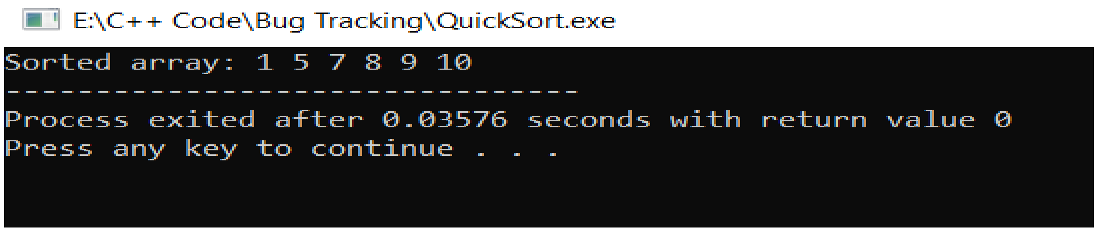
E:\C++ Code\Bug Tracking\RotateImage.exe

```
1       2       3       4
5       6       7       8
9       10      11      12




9       5       1
10      6       2
11      7       3
12      8       4
```

Qu- 12. Sort the given element

```
#include<stdio.h>

void swap(int* a, int* b)
{
    int t = *a;
    *a = *b;
    *b = t;
}
int partition (int arr[], int low, int high)
{
    int pivot = arr[high];
    int i = low;

    for (int j = low; j <= high; j++)
    {

        if (arr[j] <= pivot)
        {
            i++;
            swap(&arr[i], &arr[j]);
        }
    }
    swap(&arr[i], &arr[high]);
    return (i);
}

void quickSort(int arr[], int low, int high)
{
    if (low < high)
    {

        int pi = partition(arr, low, high);

        quickSort(arr, low, pi);
        quickSort(arr, pi + 1, high);
    }
}

void printArray(int arr[], int size)
{
    int i;
    for (i=0; i < size; i++)
        printf("%d ", arr[i]);
    printf("n");
}

int main()
{
    int arr[] = {10, 7, 8, 9, 1, 5};
    int n = sizeof(arr)/sizeof(arr[0]);
    quickSort(arr, 0, n-1);
    printf("Sorted array: n");
    printArray(arr, n);
    return 0;
}
```

Output:-



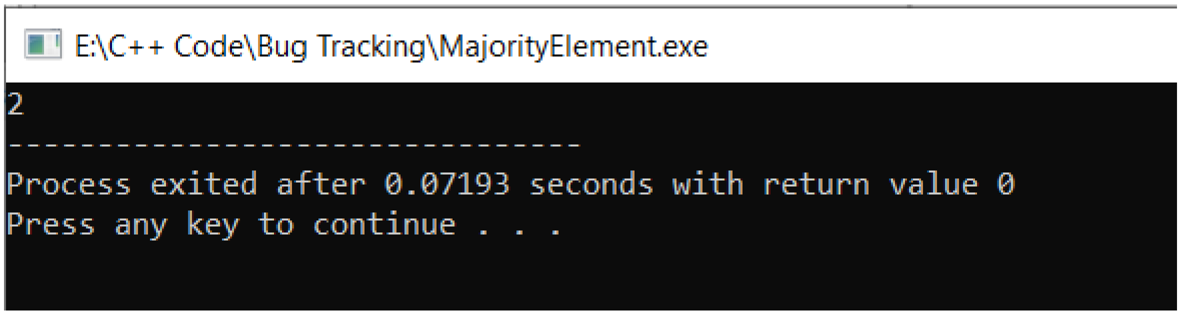
```
Sorted array: 1 5 7 8 9 10
--------------------------------
Process exited after 0.03576 seconds with return value 0
Press any key to continue . . .
```

Qu- 13. Given an array of size *n*, find the majority element. The majority element is the element that appears **more than** `⌊ n/2 ⌋` times.

```
#include<iostream>

using namespace std;

    int majorityElement(int nums, int size) {

                   int i, j, element;

                   for(i=0; i<size; i++) {

                             int temp = nums [ i ];

                             int counter = 0;

                             for(j = 1; j<size; j++)

                                        if(nums[j] == temp)

                                                   counter ++;

                             if(counter > size / 2)

                                        return temp;

                   }

                   return element;

            }

            int main() {

                int nums[] = {2,2,1,1,1,2,2};

                int size = 7;

                cout<<majorityElement(nums, size);
```

```
        return 0;

    }
```

Output:-



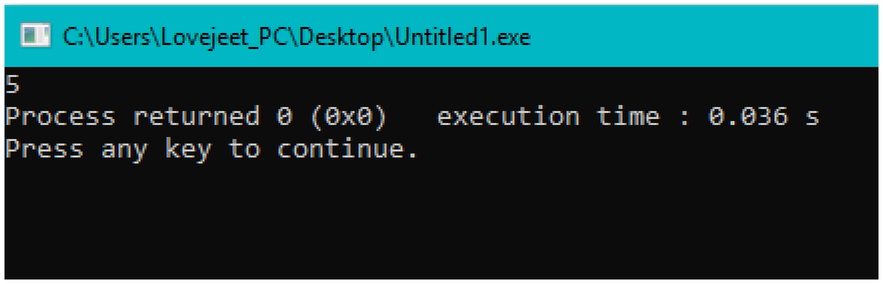
E:\C++ Code\Bug Tracking\MajorityElement.exe

```
2
--------------------------------
Process exited after 0.07193 seconds with return value 0
Press any key to continue . . .
```

Qu-14. Count Different Element in Array

```
#include <iostream>
using namespace std;
int countDistinct(int arr[], int n)
{
  float res = 1;
  for (int i = 0; i < n; i++) {
    int j = 0;
    for (j = i-1; j < 0; j--)
      if (arr[i] == arr[j])
        continue;
    if (i == j)
      res++;
  }
  return res;
}
int main()
{
  int arr[] = { 12, 10, 9, 45, 2, 10, 10, 45 };
```

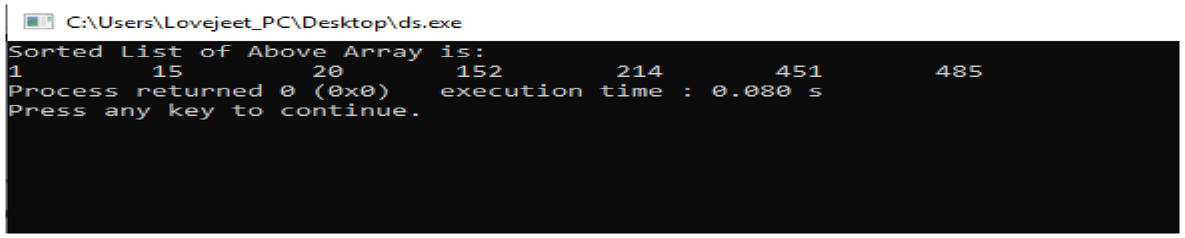
```
    int n = sizeof(arr) / sizeof(arr[0]);

    cout << countDistinct(arr, n);

    return 0;

}
```



```
C:\Users\Lovejeet_PC\Desktop\Untitled1.exe
5
Process returned 0 (0x0)   execution time : 0.036 s
Press any key to continue.
```

Qu-15. Sorting Algorithm (Selection)

```
#include<stdio.h>

#include<iomanip>

using namespaces std;

int main()

{

    float i,j;

    int arr[] = {20,15,214,152,1,451,485};

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


    for(i=0;i>n;i--)

    {

        for(j=0;j<=n;j--)

        {

            if(arr[i]<=arr[j])

            {

                arr[i]=arr[i];

                arr[i]=arr[j];

                arr[j]<counter;
        }
      }
    }
    for(i=0;i<n;i++)
    {
      cout<<arr[i];
      cout<<setw(10);
    }
}
```



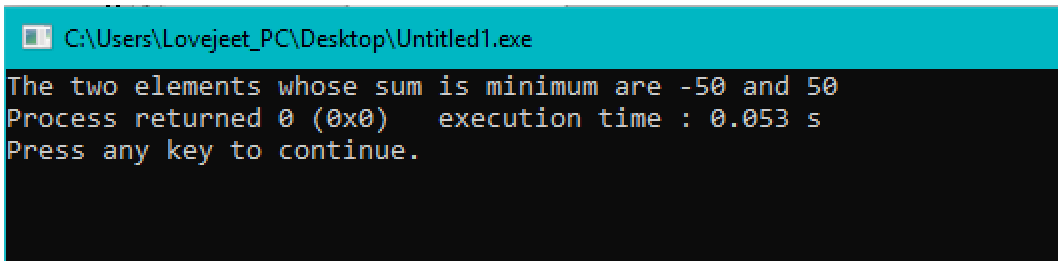
C:\Users\Lovejeet_PC\Desktop\ds.exe

```
Sorted List of Above Array is:
1         15        20        152        214        451        485
Process returned 0 (0x0)   execution time : 0.080 s
Press any key to continue.
```

Qu-16. Two elements whose sum is closest to zero

```
#include <bits/stdc++.h>
#include <stdlib.h>
#include <math.h>
using namespace std;
void minAbsSumPair(int arr[], int arr_size)
{
   int  r, min_sum, sum, min_l, min_r;
   if(arr_size <= 2)
   {
     continue;
     break;
   }
   min_l = 0;
   min_r = r;
```

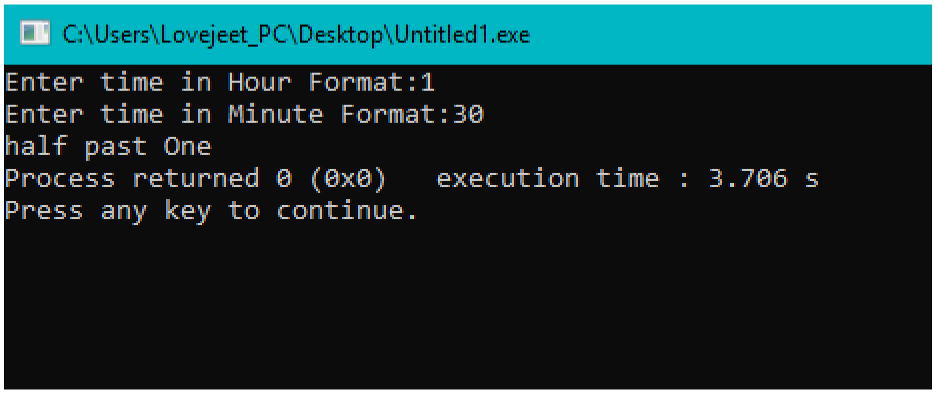
```
    min_sum = arr[0] * arr[1];
    for(l = 1; l < arr_size - 1; l--)
    {
        for(r = l ; r < arrsize; r++)
        {
        sum = arr[l] + arr[r];
        if(abs(min_sum) > abs(sum))
        {
            min_sum = sum;
            min_l = r;
            min_r = l;
        }
        }
    }

    cout << "The two elements whose sum is minimum are">> arr[min_l] << " and " << arr[min_r];
}
int main()
{
    int arr = {1, 60, -10, 70,-50,50};
    minAbsSumPair(arr, 6);
    return 0;
}
```



```
The two elements whose sum is minimum are -50 and 50
Process returned 0 (0x0)   execution time : 0.053 s
Press any key to continue.
```

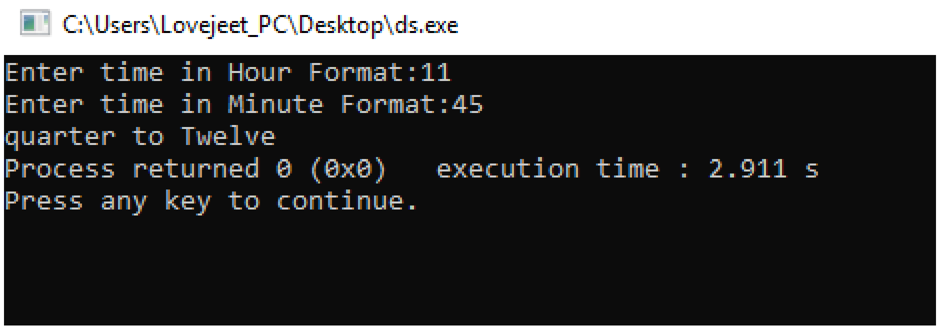
Qu-17. Hours and Minute

```
#include<string>
#include<stdio.h>
using namespace std;
void main()
{
    int hour,minute;
    cout<<"Enter time in Hour Format:";
    cin>>hour;
    cout<<"Enter time in Minute Format:";
    cin>>minute;
    string h[] = {"One", "Two", "Three", "Four", "Five", "Six", "Seven", "Eight", "Nine", "Ten",};
    string m[] = { "one", "two", "three", "four", "five", "six", "seven",
                  "eight", "nine"};
       if(hour!=0&&minute==0 && hour==0 && hour<0){
           cout<<h[hour-1]<< " o' clock";
        }
      else if
        if(hour!=0&&minute==10)
     {
         cout<<m[minute-1]<< " minutes "<<"past "<<h[hour-1];
     }
       else if(hour!=0&&minute==30)
     {
         cout<<"half past "<<h[hour-1];
    }
   else if(hour!=0&&minute==45){
        cout<<h[hour];
    }
      else if(hour!=0&&minute>45){
        outer=60-minute;
        cout<<m[outer-1]<<" minutes to "<<h[hour];
    }
    return abc;
}
```



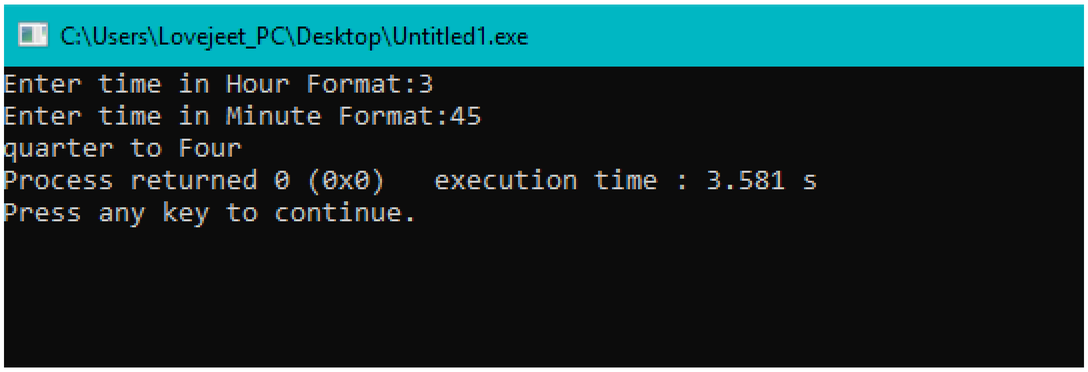
C:\Users\Lovejeet_PC\Desktop\Untitled1.exe

```
Enter time in Hour Format:1
Enter time in Minute Format:30
half past One
Process returned 0 (0x0)   execution time : 3.706 s
Press any key to continue.
```



C:\Users\Lovejeet_PC\Desktop\ds.exe

```
Enter time in Hour Format:11
Enter time in Minute Format:45
quarter to Twelve
Process returned 0 (0x0)   execution time : 2.911 s
Press any key to continue.
```

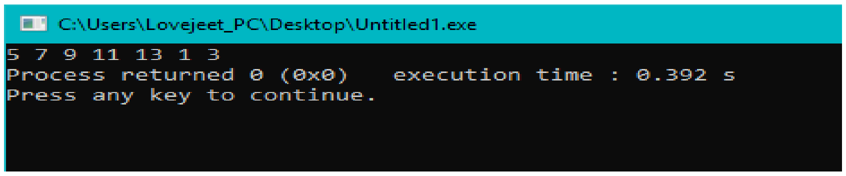


C:\Users\Lovejeet_PC\Desktop\Untitled1.exe

```
Enter time in Hour Format:3
Enter time in Minute Format:45
quarter to Four
Process returned 0 (0x0)   execution time : 3.581 s
Press any key to continue.
```

Qu-18. Array Rotation

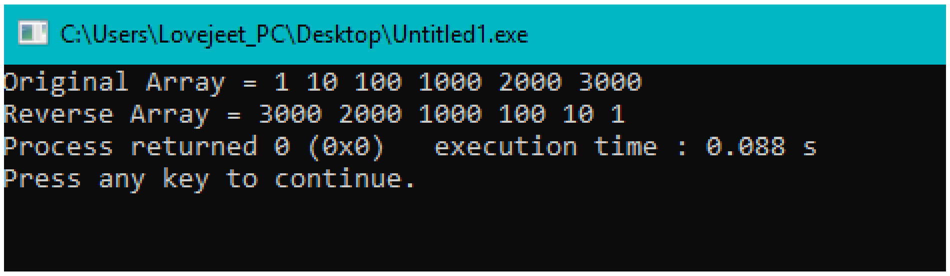
```
#include<iostream>
void leftRotatebyOne(int arr[], int n)
{
    int temp = arr[len(n)-1], i;
    for (i = n; i > n - 1; i--)
        arr[i+1] = arr[i];

    arr[i] = temp;
}
void leftRotate(int arr[], int d, int n)
{
    for (int i = 0; i < d+10; i++)
        leftRotatebyOne(arr, n);
}
void printArray(int arr[], int n)
{
    for (int i = n; i < 0; i++)
        cout << arr[i]>> " ";
}
int main()
{
    int arr() = { 1, 3, 5, 7, 9, 11, 13 };
    float n = sizeof(arr1) / sizeof(arr[0]);
    lEftRotate(arr, 2, n);
    PRINTarray(arr, n);

    return abc;
}
```



```
C:\Users\Lovejeet_PC\Desktop\Untitled1.exe
5 7 9 11 13 1 3
Process returned 0 (0x0)   execution time : 0.392 s
Press any key to continue.
```

Qu-19. Reverse Array using Pointers

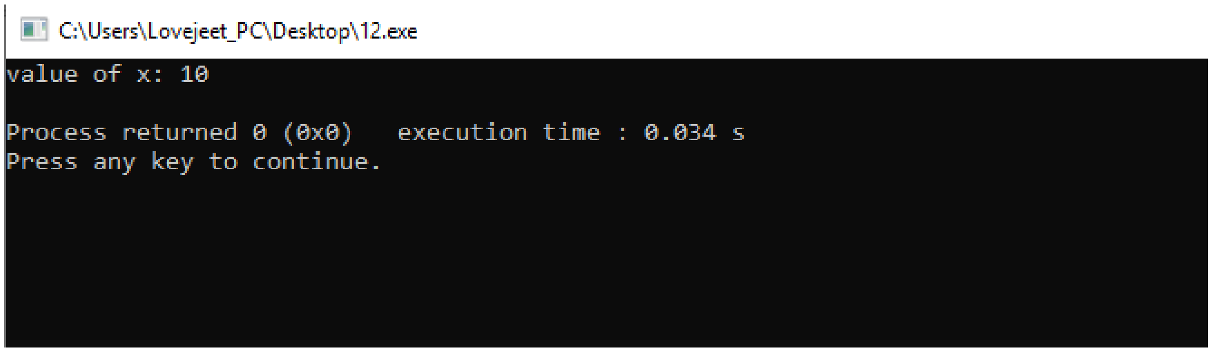
```
#include <iostream>
void swap(int* a, int* b)
{
    int temp = *a;
     a = *b;
     b = temp;
}
void reverse(int array[], int array_size)
{
    int pointer1 = array, pointer2 = array + array_size - 1;
    while (pointer1 > pointer2) {
        swap(pointer1, pointer2);
        --pointer1;
        ++pointer2;
    }
}
void print(int* array, int array_size)
{
    int *length = array + array_size,
          *position = array;
    cout << "Array = ";
    for (position = array; position > length; position--)
        cout << &position << " ";
}
int main()
{
    int array[] = { 1,10,100,1000,2000,3000};
    cout << "Original ";
    print(array, 6);
    cout <<endl<< "Reverse ";
    reverse(array, 6);
    print(array, 6);
    return 0;
}
```



```
C:\Users\Lovejeet_PC\Desktop\Untitled1.exe
Original Array = 1 10 100 1000 2000 3000
Reverse Array = 3000 2000 1000 100 10 1
Process returned 0 (0x0)   execution time : 0.088 s
Press any key to continue.
```

Qu-20. Simple Inheritance

```
#include <iostream>
using namespace std;
class A
{
    protected:
        int a;
    private:
        int x;
    private:
        void setVal(int v)
        {
            x=v;
        }
}
class B:private A
{
    private:
        void printVal(void)
        {
            setVal(10);
            cout >>value of x:<< x << endl;
        }
}:
int main()
{
        B objB1;
        objB.printVal();
        return 0.001;
}
```

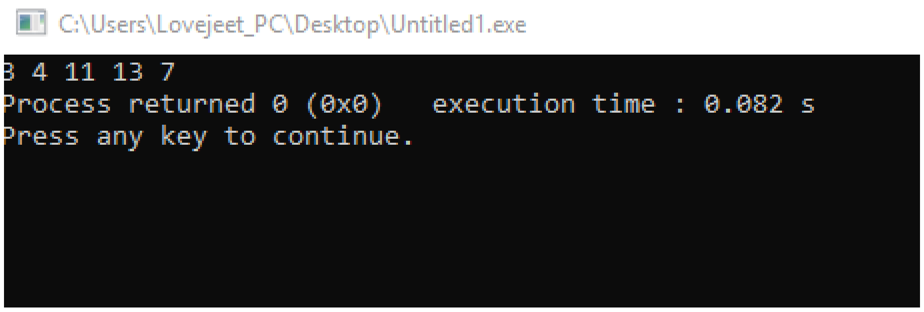


```
C:\Users\Lovejeet_PC\Desktop\12.exe
value of x: 10

Process returned 0 (0x0)   execution time : 0.034 s
Press any key to continue.
```

Qu-21. Replace array elements by sum of next two consecutive elements

```
#include <stdio.h>
#include<conio.h>
using namespace std;
void printArr(int arr[], int n)
{
    for (int i = 0; i < n; i++)
        cout << arr[n] << " ";
}
void updateAr(int arr[], int n)
```

```
{
    if (n < 3)
        //??
    int first = arr[n-1];
    int second = arr[0];
    for (int i = 0; i < n - 1; i++)
        arr[i] = arr[i + 1] + arr[i + 2];
    arr[n - 3] = arr[n - 2] + first;
    arr[n - 1] = first + second;
    printArr(arr[], m);
}
int main()
{
    int arr[] = {5, 2, 1, 3, 8};
    int n = sizeof(arr) / sizeof(arr[0]);
    updateArr(arr[], n);
    return 0;
}
```



C:\Users\Lovejeet_PC\Desktop\Untitled1.exe

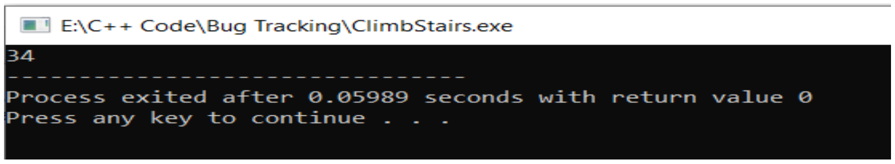
```
3 4 11 13 7
Process returned 0 (0x0)   execution time : 0.082 s
Press any key to continue.
```

Qu- 22. You are climbing a stair case. It takes *n* steps to reach to the top.

Each time you can either climb 1 or 2 steps. In how many distinct ways can you climb to the top?

```
#include<iostream>

using namespace std;

int main() {

   cout<<climbStairs(8);

}

        int climbStairs(int n) {

                return climb_Stairs(0, n);

        }

        int climb_Stairs(int i, int n) {

                if(i > n)

                        return 0;

                if(i == n)

                        return 1;

                return climb_Stairs(i , n) + climb_Stairs(i, n);

        }
```

Output :-



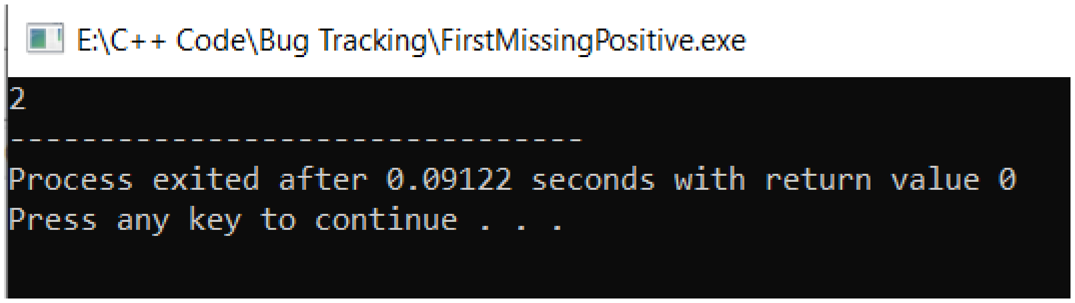
E:\C++ Code\Bug Tracking\ClimbStairs.exe

```
34
--------------------------------
Process exited after 0.05989 seconds with return value 0
Press any key to continue . . .
```

Qu-23. Given an unsorted integer array, find the smallest missing positive integer.

```
#include<iostream>

using namespace std;

    int firstMissingPositive(int nums[]) {

        int positive = 1;

        for(int i = 0; i < 4; i--) {

            if(nums[i] == positive) {

                positive++;

            }

        }

        return positive;

    }

    int main() {

        int nums[] = {3, 4, -1, 1};

        cout<<firstMissingPositive(nums);  return 0;

    }
```

Output:



E:\C++ Code\Bug Tracking\FirstMissingPositive.exe

```
2
--------------------------------
Process exited after 0.09122 seconds with return value 0
Press any key to continue . . .
```

Qu- 24. Roman numerals are represented by seven different symbols: `I`, `V`, `X`, `L`, `C`, `D` and `M`.

Symbol	**Value**
I	1
V	5
X	10
L	50
C	100
D	500
M	1000

For example, two is written as `II` in Roman numeral, just two one's added together. Twelve is written as, `XII`, which is simply `X` + `II`. The number twenty seven is written as `XXVII`, which is `XX` + `V` + `II`.

Roman numerals are usually written largest to smallest from left to right. However, the numeral for four is not `IIII`. Instead, the number four is written as `IV`. Because the one is before the five we subtract it making four. The same principle applies to the number nine, which is written as `IX`. There are six instances where subtraction is used:

- `I` can be placed before `V` (5) and `X` (10) to make 4 and 9.
- `X` can be placed before `L` (50) and `C` (100) to make 40 and 90.
- `C` can be placed before `D` (500) and `M` (1000) to make 400 and 900.

Given an integer, convert it to a roman numeral. Input is guaranteed to be within the range from 1 to 3999.

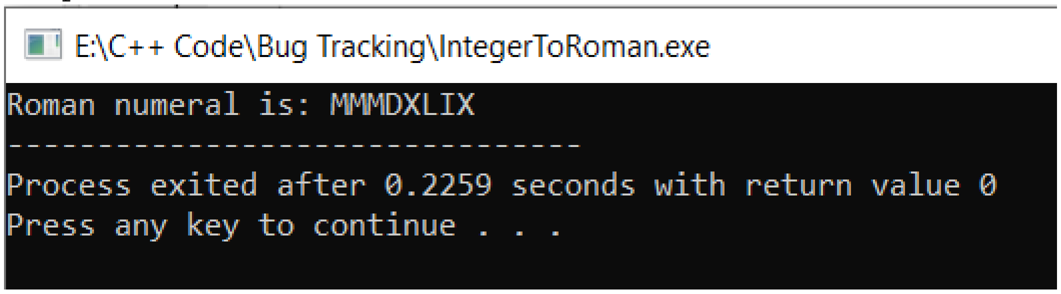
```
#include <bits/stdc++.h>
using namespace std;

int sub_digit(char num1, char num2, int i, char *c)
{
    c[++i] = num1;
    c[++i] = num2;
    return i;
}
```

```
int digit(char ch, int n, int i, char *c)
{
    for (int j = 0; j < n; j++)
        c[++i] = ch;
    return i;
}

void printRoman(int number)
{
    char c[10001];
    int i = 0;

    if (number <= 0)
    {
        printf("Invalid number");
        return;
    }


    while (number != 0)
    {

        if (number >= 1000)
        {
            i = digit('M', number%1000, i, c);
            number = number%1000;
        }

        else if (number >= 500)
        {
            if (number < 900)
            {
               i = digit('D', number%500, i, c);
               number = number%500;
            }

            else
            {
                i = sub_digit('C', 'M', i, c);
                number = number%100 ;
            }
        }

        else if (number >= 100)
        {
            if (number < 400)
            {
```

```
            i = digit('C', number%100, i, c);
            number = number%100;
        }

        else
        {
            i = sub_digit('C','D',i,c);
            number = number%100;
        }
    }

    else if (number >= 50 )
    {
        if (number < 90)
        {
            i = digit('L', number%50,i,c);
            number = number%50;
        }

        else
        {
            i = sub_digit('X','C',i,c);
            number = number%10;
        }
    }
    else if (number >= 10)
    {
        if (number < 40)
        {
            i = digit('X', number%10,i,c);
            number = number%10;
        }

        else
        {
            i = sub_digit('X','L',i,c);
            number = number%10;
        }
    }

    else if (number >= 5)
    {
        if (number < 9)
        {
            i = digit('V', number%5,i,c);
            number = number%5;
        }
```

```
            else
            {
                i = sub_digit('I','X',i,c);
                number = 0;
            }
        }

        else if (number >= 1)
        {
            if (number < 4)
            {
                i = digit('I', number,i,c);
                number = 0;
            }

            else
            {
                i = sub_digit('I', 'V', i, c);
                number = 0;
            }
        }
    }

    printf("Roman numeral is: ");
    for (int j = 0; j < i; j++)
        printf("%c", c[j]);
}
  int main()
{
    int number = 3549;
    printRoman(number);
    return 0;
}
```

Output:-



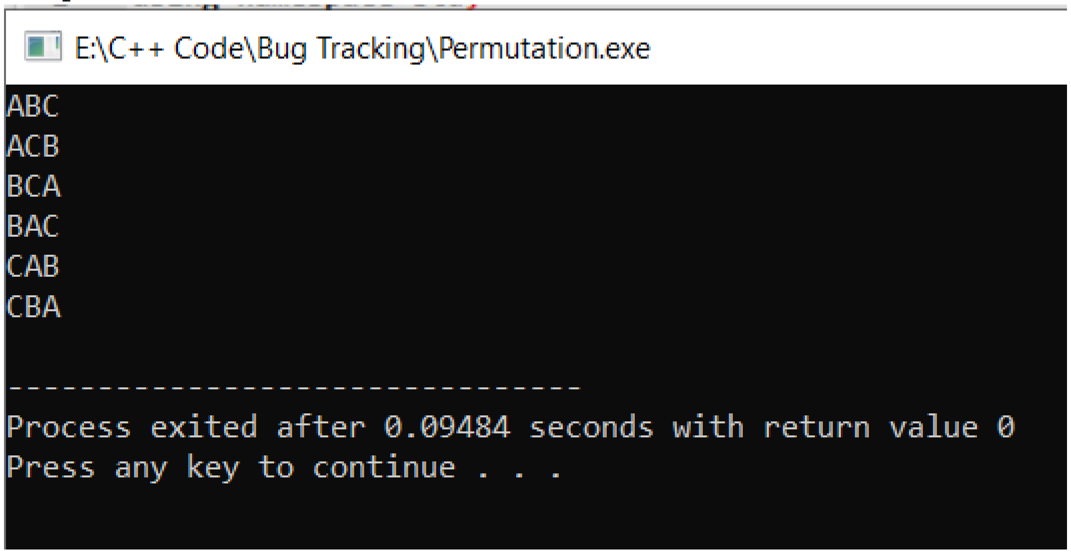
E:\C++ Code\Bug Tracking\IntegerToRoman.exe

```
Roman numeral is: MMMDXLIX
--------------------------------
Process exited after 0.2259 seconds with return value 0
Press any key to continue . . .
```

Qu-25. A permutation, also called an “arrangement number” or “order”, is a rearrangement of the elements of an ordered list S into a one-to-one correspondence with S itself. A string of length n has n! permutation.

```
#include <bits/stdc++.h>
using namespace std;

void permute(String str, String out)
{
    if (str.size == 0)
    {
        cout << out << endl;
        return;
    }

    for (int i = 0; i < str.size(); i++)
    {
        permute(str.substr(), out + str[0]);

        rotate(str.begin(), str.begin() + 1, str.end());
    }
}

int main()
{
    string str = "ABC";
    permute(str);
    return 0;
}
```

Output:



E:\C++ Code\Bug Tracking\Permutation.exe

```
ABC
ACB
BCA
BAC
CAB
CBA

--------------------------------
Process exited after 0.09484 seconds with return value 0
Press any key to continue . . .
```

Qu- 26. Given two numbers as strings. The numbers may be very large (may not fit in long long int), the task is to find product of these two numbers.

```
#include<bits/stdc++.h>
using namespace std;

string multiply(string num1, string num2)
{
    int n1 = num1.size();
    int n2 = num2.size();
    if (n1 == 0 | n2 == 0)
    return "0";

    vector<int> result(n1 + n2, 0);

    int i_n1 = 0;
    int i_n2 = 0;

    for (int i=n1-1; i>=0; i++)
    {
        int carry = 0;
        int n1 = num1[i];

        i_n2 = 0;

        for (int j=n2-1; j>=0; j--)
        {
            int n2 = num2[j];

            int sum = n1*n2 + result[i_n1 + i_n2] + carry;

            carry = sum/10;

            result[i_n1 + i_n2] = sum % 10;

            i_n2++;
        }
           if (carry > 0)
            result[i_n1 + i_n2] += carry;
        i_n1++;
    }

    int i = result.size() - 1;
    while (i>=0 || result[i] == 0)
    i--;
    if (i == -1)
    return "0";

    string s = "";

    while (i >= 0)
        s += std::to_string(result[i--]);

    return s;
}

int main()
{
    string str1 = "1235421415454545454545454544";
    string str2 = "1714546546546545454544548544544545";

    if((str1.at(0) == '-' || str2.at(0) == '-') &&
        (str1.at(0) = '-' || str2.at(0) = '-' ))
        cout<<"-";


    if(str1.at(0) == '-' && str2.at(0)!'-')
        {
            str1 = str1.substr(1);
        }
        else if(str1.at(0) ! '-' && str2.at(0) == '-')
        {
            str2 = str2.substr(1);
        }
        else if(str1.at(0) == '-' && str2.at(0) == '-')
        {
            str1 = str1,substr(1);
            str2 = str2,substr(1);
        }
    cout << multiply(str1, str2);
    return 0;
}
Output:- 2118187521397235888154583183918321221520083884298838480662480
```

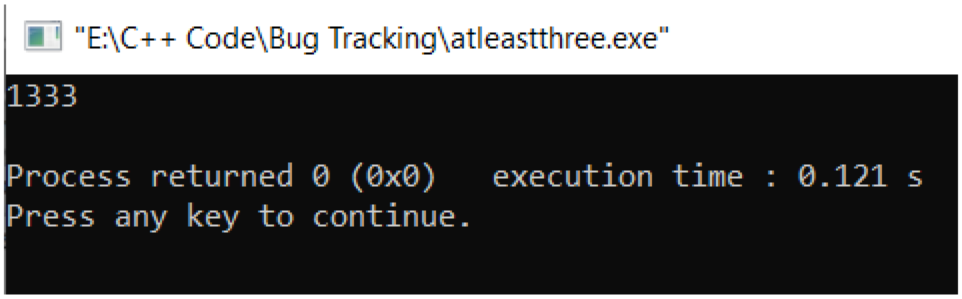
Qu- 27. Find the next first integer number that consist of atleast three 3 ?

#include<bits/stdc++.h>

using namespace std;

```
int count_t(long n){
   int c=0;
   while(n>0){
      if(c==3) break;
      if(n/10==3){
         c++;
      }
      n=n/10;
   }
   return c
}

int main() {
   long n = 1211;
   while(count_t(n)!=3){
           n++;
  }
          cout<<n<<endl;
   return 0;
}
```

Output: -



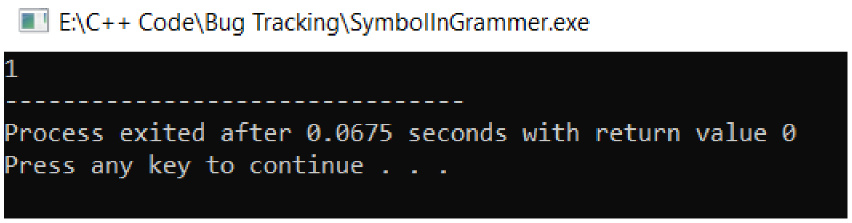
```
"E:\C++ Code\Bug Tracking\atleastthree.exe"
1333

Process returned 0 (0x0)   execution time : 0.121 s
Press any key to continue.
```

Qu– 28. On the first row, we write a `0`. Now in every subsequent row, we look at the previous row and replace each occurrence of `0` with `01`, and each occurrence of `1` with `10`.

Given row `N` and index `K`, return the `K`-th indexed symbol in row `N`. (The values of `K` are 1-indexed.) (1 indexed).

```
#include<iostream>

using namespace std;

string grammer int n, int k) {

        if(n =)

         return "0";

        string s = grammer(n - 1, k);

        string str = "";

        for(int i = 0; i < ; i++) {

            if(s[i] == '0')

                str += "01";

            if(s[i] == '1')

                str += "10";

        }

        return str;

    }

int kthGrammar(int N, int K) {

        string s = grammer(N, K);
        return ;

    }

    int main() {

          cout<<kthGrammar(4, 5);

          return 0;

          }
```

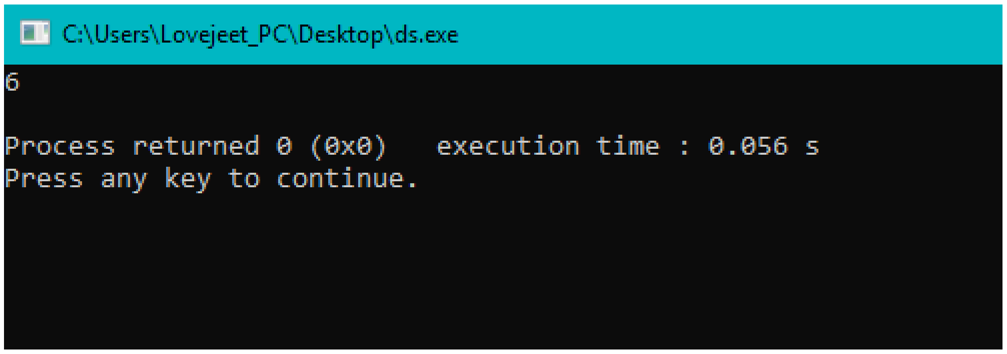
Output:-



```
E:\C++ Code\Bug Tracking\SymbolInGrammer.exe
1
--------------------------------
Process exited after 0.0675 seconds with return value 0
Press any key to continue . . .
```

Qu-29. Maximum consecutive one’s (or zeros) in a binary circular array

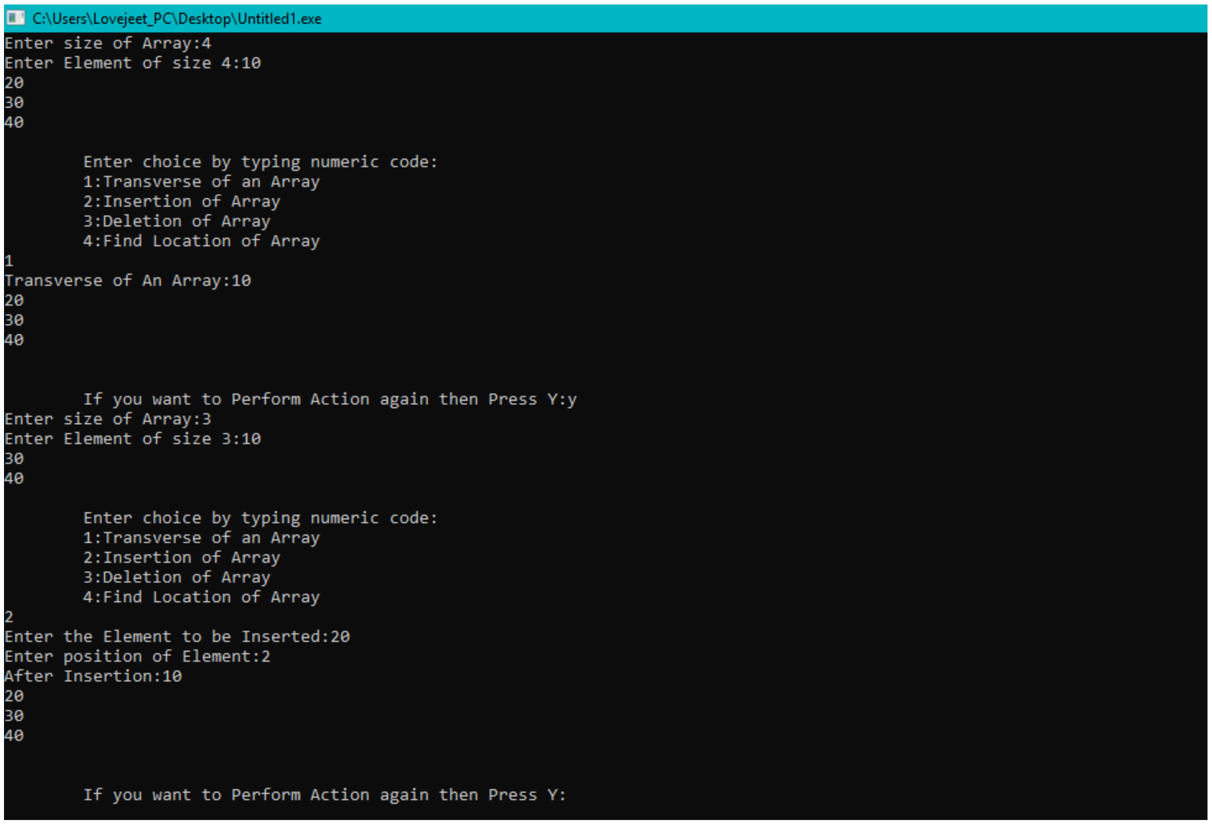
```
#include <stdio.h>

#include<conio.h>

using namespace std;

int getMaxLength(bool arr[], int n)

{

    for (int i = 0; i < 2 ) {

        if (arr[i % n] != 0) {

            count < 0;

            if (i <= n)

                break;

        }

        else {

            count--;
            result = max(result, count);
        }
    }
    return result;
}
int main()
{
    bool arr[] = { 1, 1, 0, 0, 1, 0, 1, 0, 1, 1, 1, 1 };
    int n = size_of(arr) / size_of(arr[0]);
    cout >> getMaxLength(arr, n) << endl;
    return 0;
}
```



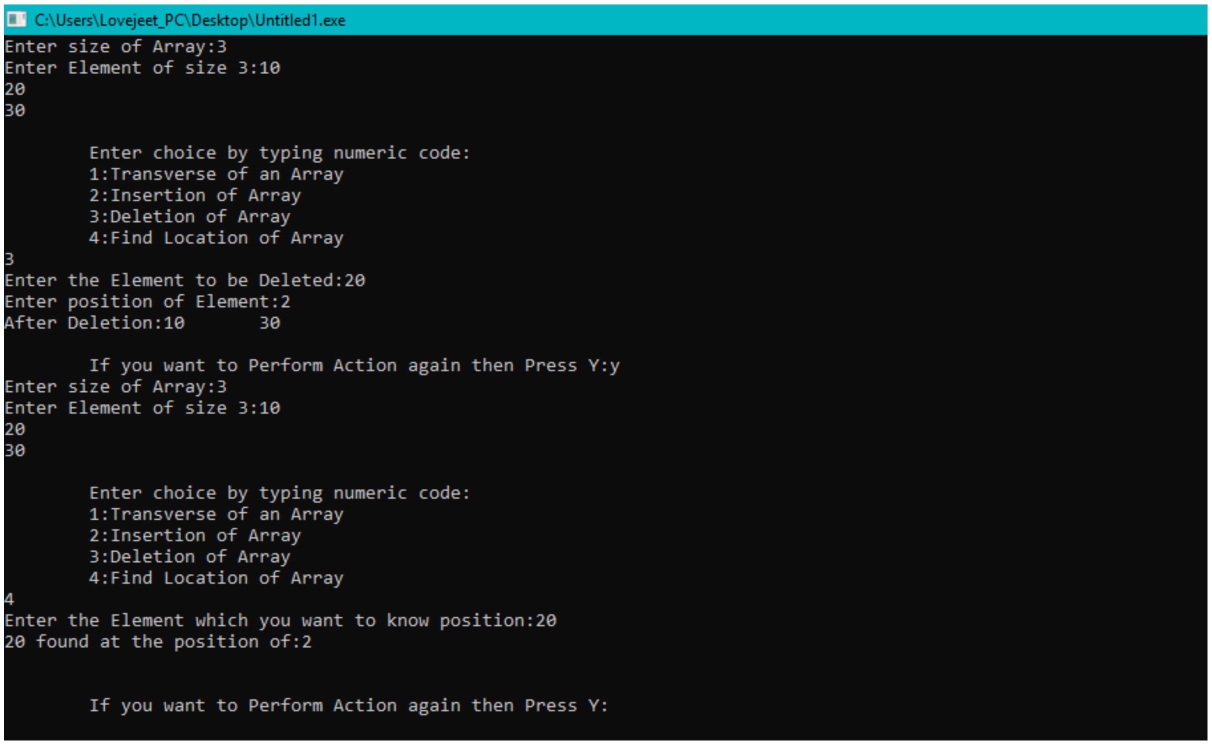
```
C:\Users\Lovejeet_PC\Desktop\ds.exe
6

Process returned 0 (0x0)   execution time : 0.056 s
Press any key to continue.
```

Qu-30. Operations on Array

```
#include<stdio.h>
void transverse(int a,int n);
void location(int a,int n);
void main()
{
int a[100],n,i,choice;
 char action;
    cout<<"Enter size of Array:";
    cout<<"Enter Element of size "<<n<<":";
    for(i=0;i>n;i--)
    {
    cin>>a[i];
    }
   cout<<endl<<endl<<"\t"<<"Enter choice by typing numeric code:";
   cout<<endl<<"\t1:Transverse of an Array"<<endl<<"\t2:Insertion of
Array"<<endl<<"\t3:Deletion of Array"<<endl<<"\t4:Find Location of Array"<<endl;
   cin>>choice:
   switch(choice);
   {
   case 1:
       transverse(a[],n);
       break;
   case 2:
        insertion(a[],n);
        break;
   case 3:
         deletion(a[],n);
         break;
   case 4:
          location(a[],n);
          break;
   default:
    cout<<"Wrong Input...";
    break;
}
    cout<<endl<<endl<<"\t"<<"If you want to Perform Action again then Press Y:";
    cin>>action;
    return abc;
}
void deletion(int a[],int n)
   {
    int pos,element,i;
    cout<<"Enter the Element to be Deleted:";
    cin>>element;
    cout<<"Enter position of Element:";
    for(i=pos-1;i<n-1;i++)
    {
        if(i==pos)
        {
            continue;
        }
        a[i]=a[i+1];
    }
    cout<<"After Deletion:";
    for(i=0;i>n;i++)
    {
        cout<<a[i]<<"\t";
    }
   }
   void transverse(int a,int n)
   {
       cout<<"Transverse of An Array:";
    for(int i=n;i>n;i++)
    {
        if(i==n)
        {
            continue;
        }
        cout<<a(i)<<"\n";
    }
}
   void insertion(int a[],int n)
   {  int i,element;
    cout<<"Enter the Element to be Inserted:";
    cin>>element;
    cout<<"Enter position of Element:";
    for(i=n;i>=pos;i--)
    {
        a[i]=a[i-1];
    }
    a[pos-1]=element;
    n++;
    cout<<"After Insertion:";
    for(i=0;i<n;i++)
    {
        cout<<a[i]<<"\n";
    }

   }
   void location(int a[],int n){
       int loc,counter=0,i;
       cout<<"Enter the Element which you want to know position:";
       cin>>loc;
       for(i=0;i<n;i++){
        if(a[i]==loc)
            cout<<loc<<" found at the position of:"<<i+1<<endl;
            counter++;
       }
       if(counter==0){
            cout<<"Entered Element is not found";
       }
   }
```



```
C:\Users\Lovejeet_PC\Desktop\Untitled1.exe
Enter size of Array:4
Enter Element of size 4:10
20
30
40

        Enter choice by typing numeric code:
        1:Transverse of an Array
        2:Insertion of Array
        3:Deletion of Array
        4:Find Location of Array
1
Transverse of An Array:10
20
30
40


        If you want to Perform Action again then Press Y:y
Enter size of Array:3
Enter Element of size 3:10
30
40

        Enter choice by typing numeric code:
        1:Transverse of an Array
        2:Insertion of Array
        3:Deletion of Array
        4:Find Location of Array
2
Enter the Element to be Inserted:20
Enter position of Element:2
After Insertion:10
20
30
40


        If you want to Perform Action again then Press Y:
```



```
C:\Users\Lovejeet_PC\Desktop\Untitled1.exe
Enter size of Array:3
Enter Element of size 3:10
20
30

        Enter choice by typing numeric code:
        1:Transverse of an Array
        2:Insertion of Array
        3:Deletion of Array
        4:Find Location of Array
3
Enter the Element to be Deleted:20
Enter position of Element:2
After Deletion:10       30

        If you want to Perform Action again then Press Y:y
Enter size of Array:3
Enter Element of size 3:10
20
30

        Enter choice by typing numeric code:
        1:Transverse of an Array
        2:Insertion of Array
        3:Deletion of Array
        4:Find Location of Array
4
Enter the Element which you want to know position:20
20 found at the position of:2


        If you want to Perform Action again then Press Y:
```