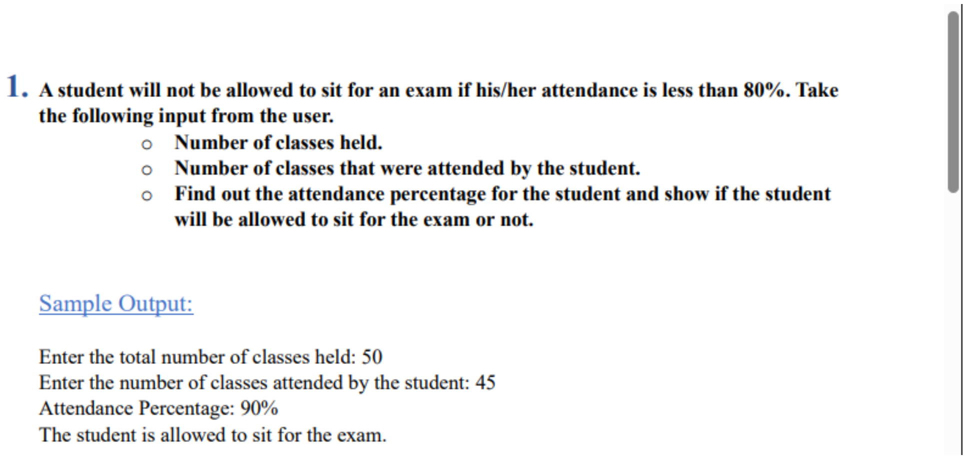
Assignment

Name:MD.Abdur Rafi

ID:24-57874-2 Serial:15

Quition:1

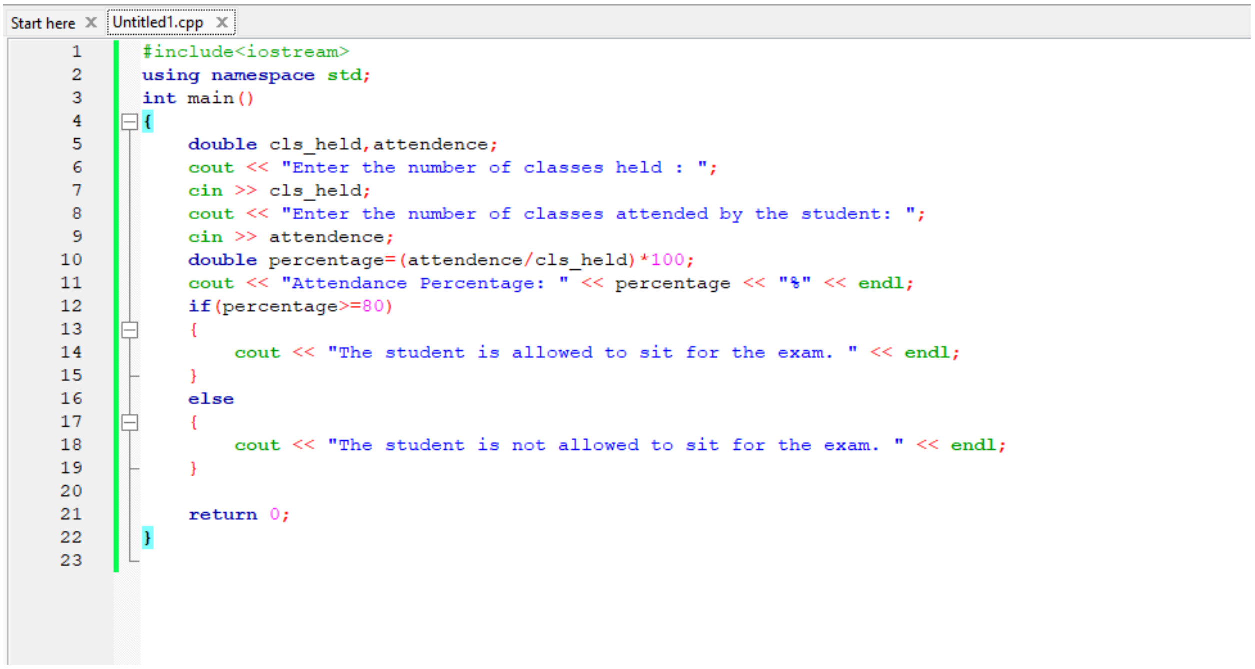


1. **A student will not be allowed to sit for an exam if his/her attendance is less than 80%. Take the following input from the user.**
   - ○ **Number of classes held.**
   - ○ **Number of classes that were attended by the student.**
   - ○ **Find out the attendance percentage for the student and show if the student will be allowed to sit for the exam or not.**

Sample Output:

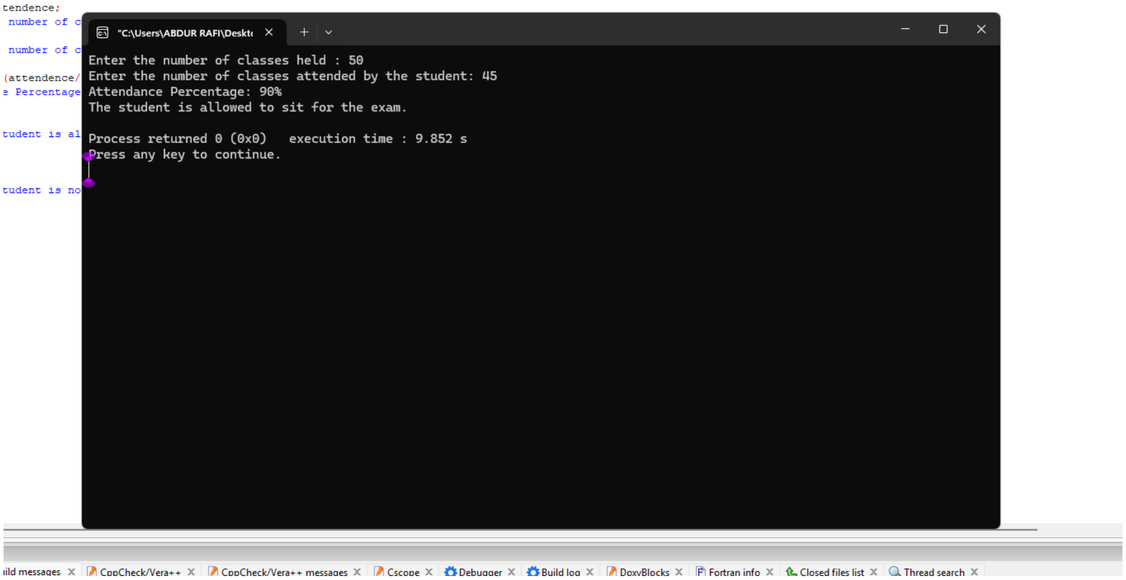
```
Enter the total number of classes held: 50
Enter the number of classes attended by the student: 45
Attendance Percentage: 90%
The student is allowed to sit for the exam.
```

Answer:



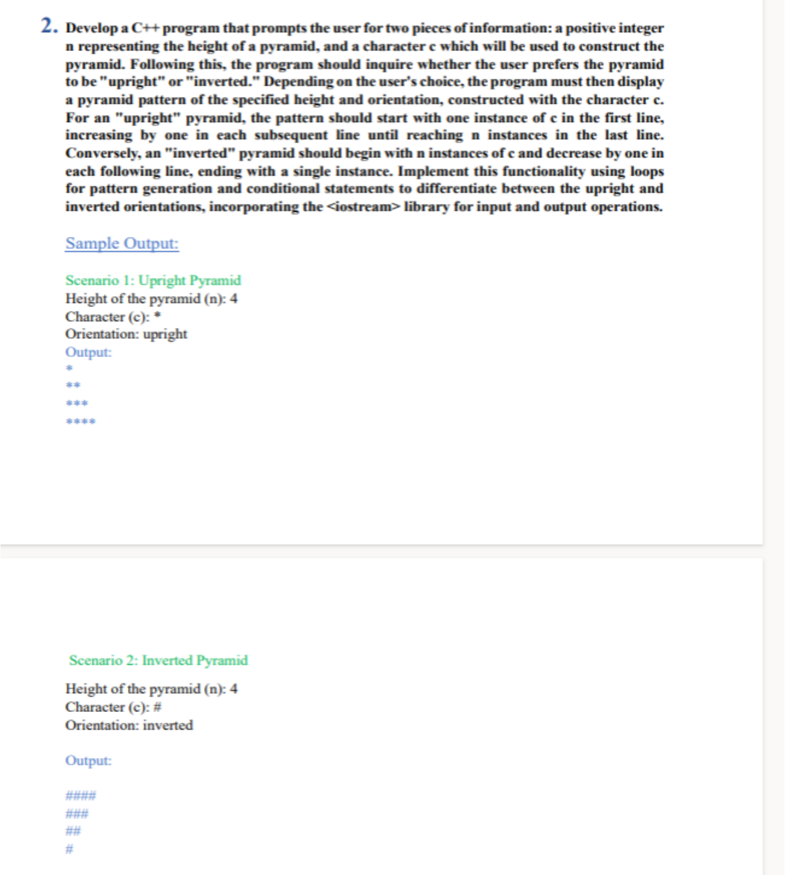
```cpp
#include<iostream>
using namespace std;
int main()
{
    double cls_held,attendence;
    cout << "Enter the number of classes held : ";
    cin >> cls_held;
    cout << "Enter the number of classes attended by the student: ";
    cin >> attendence;
    double percentage=(attendence/cls_held)*100;
    cout << "Attendance Percentage: " << percentage << "%" << endl;
    if(percentage>=80)
    {
        cout << "The student is allowed to sit for the exam. " << endl;
    }
    else
    {
        cout << "The student is not allowed to sit for the exam. " << endl;
    }

    return 0;
}

```

Output:



```
Enter the number of classes held : 50
Enter the number of classes attended by the student: 45
Attendance Percentage: 90%
The student is allowed to sit for the exam.

Process returned 0 (0x0)   execution time : 9.852 s
Press any key to continue.
```

Quition:2



2. Develop a C++ program that prompts the user for two pieces of information: a positive integer n representing the height of a pyramid, and a character c which will be used to construct the pyramid. Following this, the program should inquire whether the user prefers the pyramid to be "upright" or "inverted." Depending on the user's choice, the program must then display a pyramid pattern of the specified height and orientation, constructed with the character c. For an "upright" pyramid, the pattern should start with one instance of c in the first line, increasing by one in each subsequent line until reaching n instances in the last line. Conversely, an "inverted" pyramid should begin with n instances of c and decrease by one in each following line, ending with a single instance. Implement this functionality using loops for pattern generation and conditional statements to differentiate between the upright and inverted orientations, incorporating the <iostream> library for input and output operations.

Sample Output:

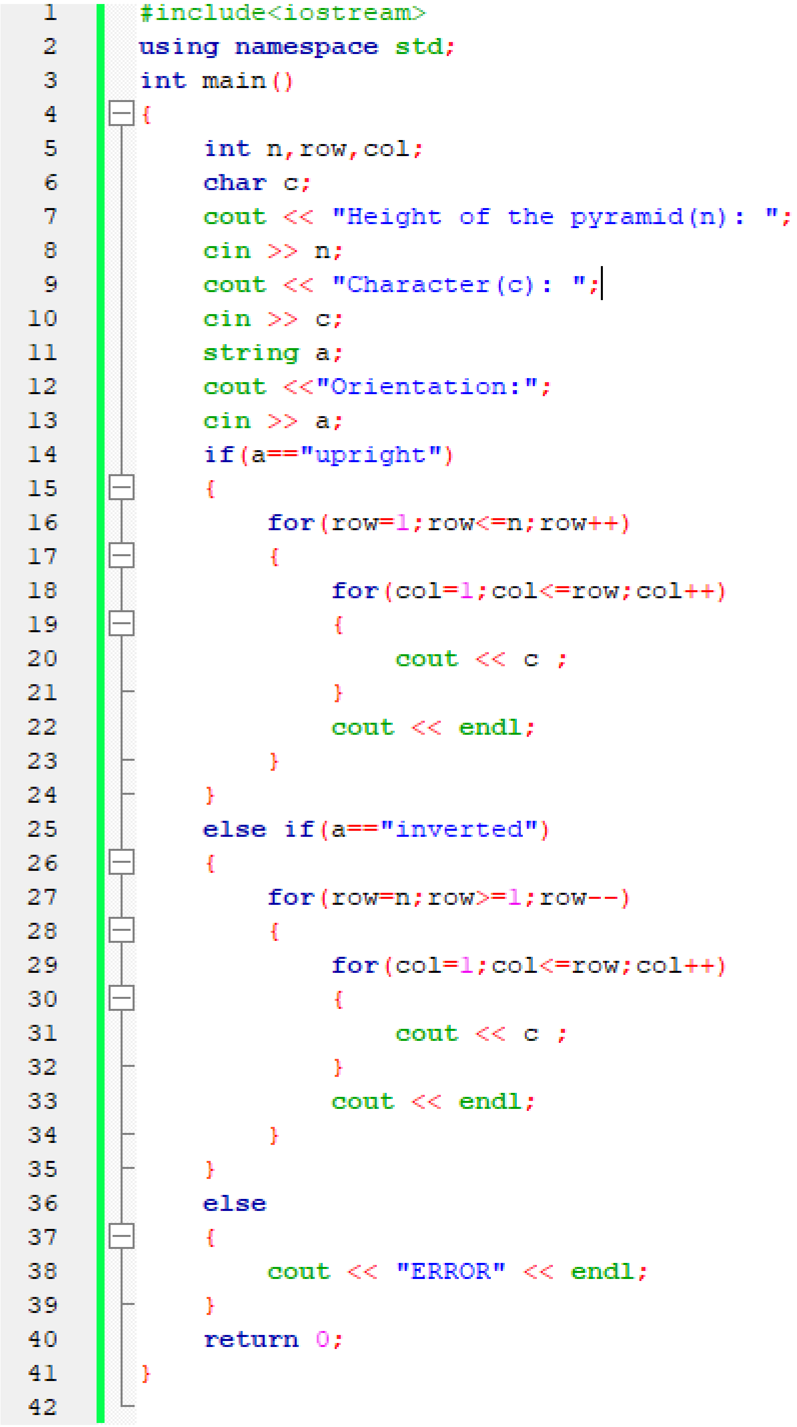
Scenario 1: Upright Pyramid

```
Height of the pyramid (n): 4
Character (c): *
Orientation: upright
Output:
*
**
***
****
```

Scenario 2: Inverted Pyramid

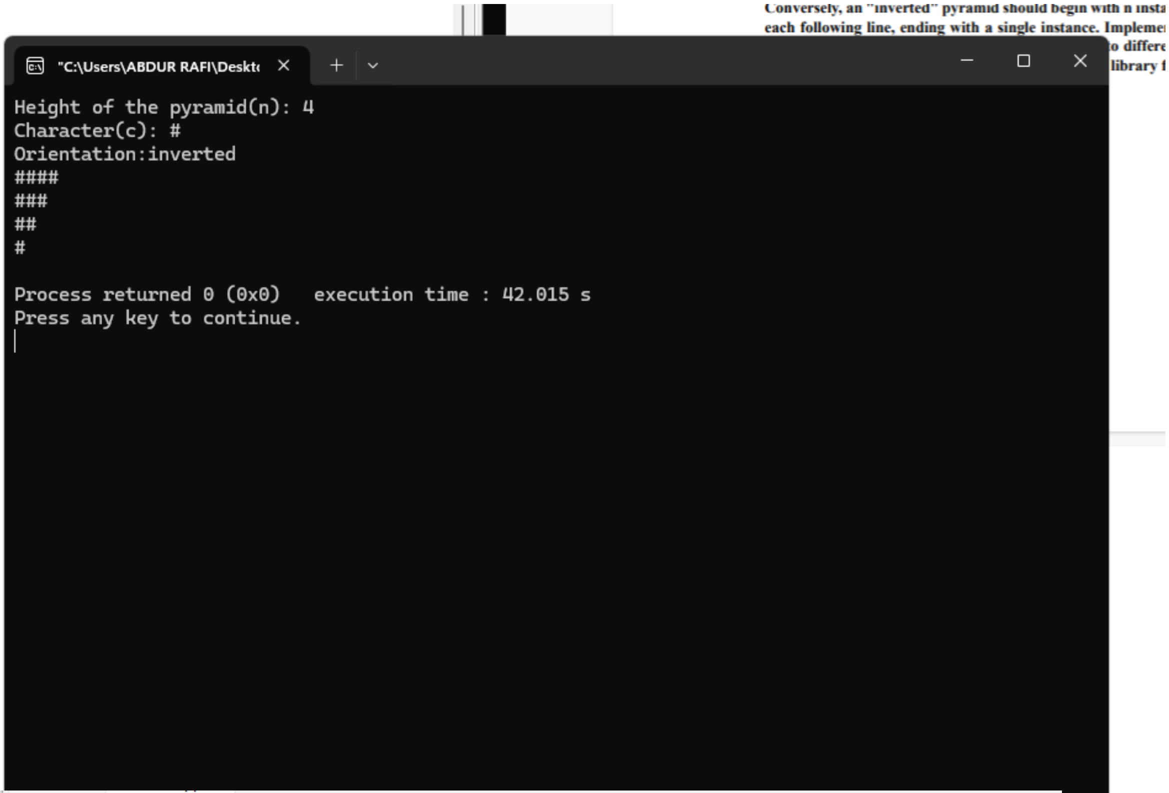
```
Height of the pyramid (n): 4
Character (c): #
Orientation: inverted

Output:

####
###
##
#
```

Answer:



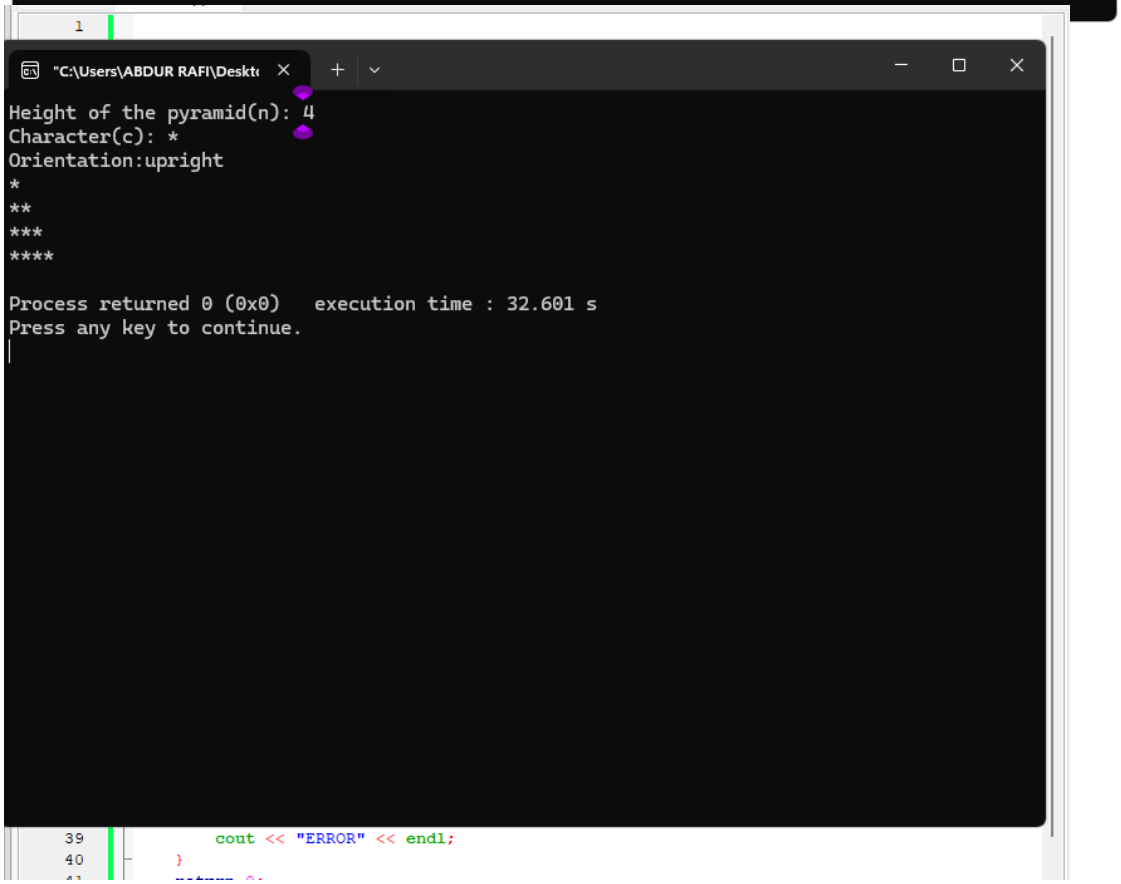
```cpp
#include<iostream>
using namespace std;
int main()
{
    int n,row,col;
    char c;
    cout << "Height of the pyramid(n): ";
    cin >> n;
    cout << "Character(c): ";
    cin >> c;
    string a;
    cout <<"Orientation:";
    cin >> a;
    if(a=="upright")
    {
        for(row=1;row<=n;row++)
        {
            for(col=1;col<=row;col++)
            {
                cout << c ;
            }
            cout << endl;
        }
    }
    else if(a=="inverted")
    {
        for(row=n;row>=1;row--)
        {
            for(col=1;col<=row;col++)
            {
                cout << c ;
            }
            cout << endl;
        }
    }
    else
    {
        cout << "ERROR" << endl;
    }
    return 0;
}
```

Output:



Conversely, an "inverted" pyramid should begin with n insta
each following line, ending with a single instance. Impleme
o differe
library f

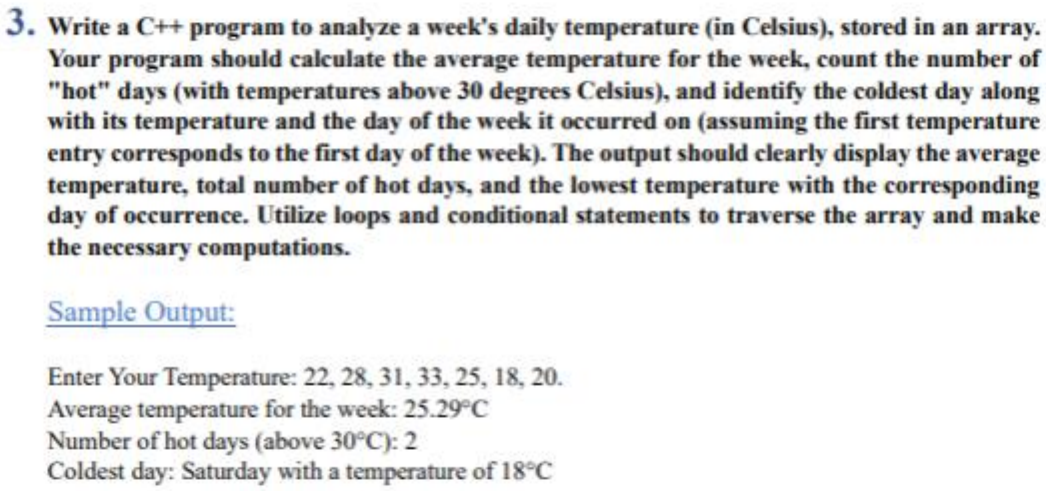
```
Height of the pyramid(n): 4
Character(c): #
Orientation:inverted
####
###
##
#

Process returned 0 (0x0)   execution time : 42.015 s
Press any key to continue.
```



```
Height of the pyramid(n): 4
Character(c): *
Orientation:upright
*
**
***
****

Process returned 0 (0x0)   execution time : 32.601 s
Press any key to continue.
```

```
        cout << "ERROR" << endl;
    }
```

Quition:3

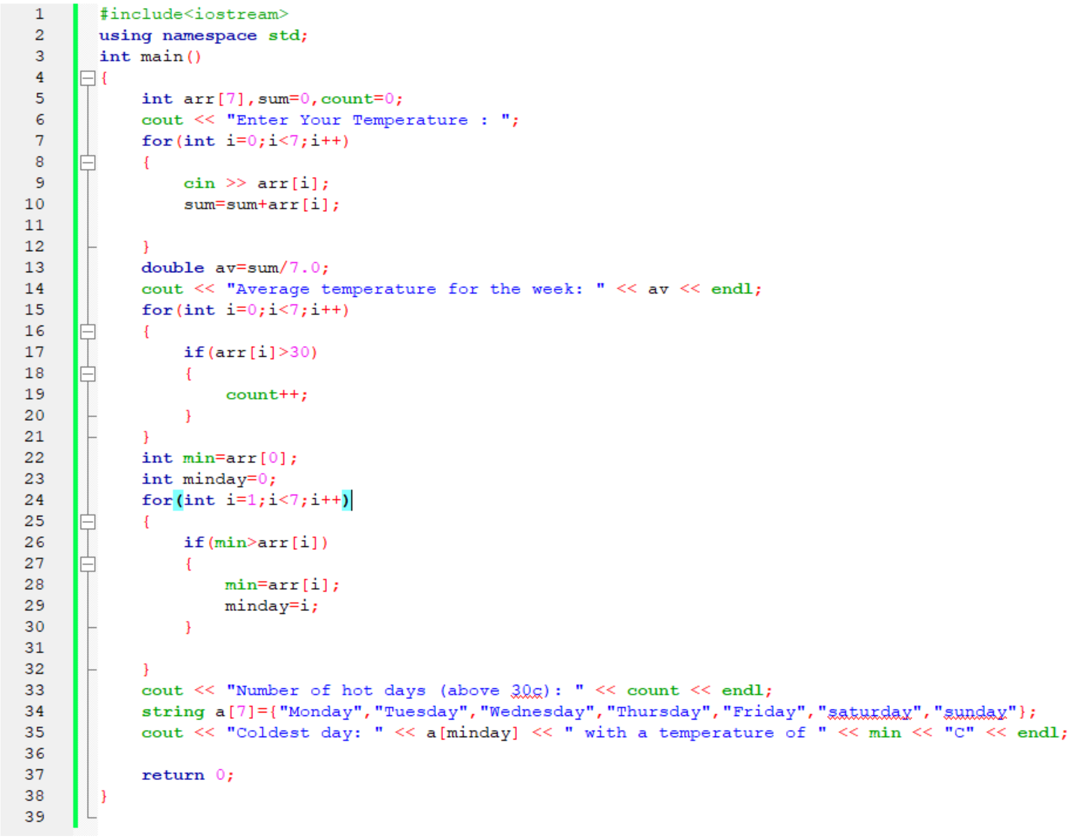


3. **Write a C++ program to analyze a week's daily temperature (in Celsius), stored in an array. Your program should calculate the average temperature for the week, count the number of "hot" days (with temperatures above 30 degrees Celsius), and identify the coldest day along with its temperature and the day of the week it occurred on (assuming the first temperature entry corresponds to the first day of the week). The output should clearly display the average temperature, total number of hot days, and the lowest temperature with the corresponding day of occurrence. Utilize loops and conditional statements to traverse the array and make the necessary computations.**

Sample Output:

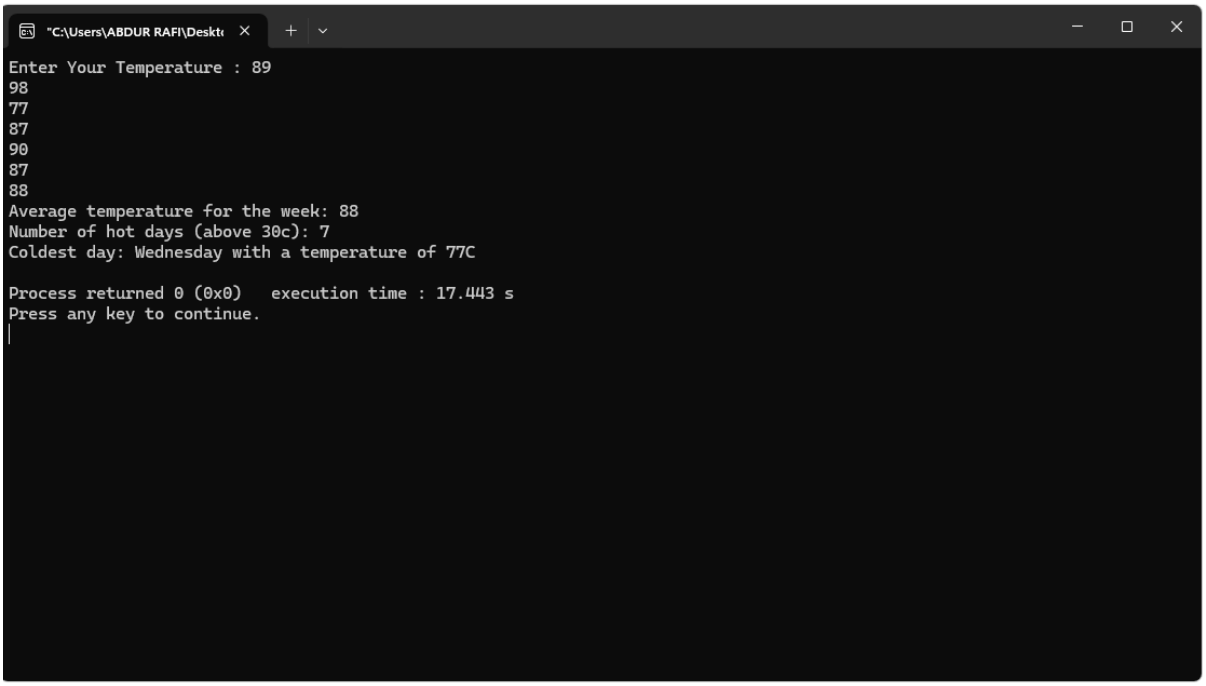
```
Enter Your Temperature: 22, 28, 31, 33, 25, 18, 20.
Average temperature for the week: 25.29°C
Number of hot days (above 30°C): 2
Coldest day: Saturday with a temperature of 18°C
```

Answer:



```cpp
#include<iostream>
using namespace std;
int main()
{
    int arr[7],sum=0,count=0;
    cout << "Enter Your Temperature : ";
    for(int i=0;i<7;i++)
    {
        cin >> arr[i];
        sum=sum+arr[i];

    }
    double av=sum/7.0;
    cout << "Average temperature for the week: " << av << endl;
    for(int i=0;i<7;i++)
    {
        if(arr[i]>30)
        {
            count++;
        }
    }
    int min=arr[0];
    int minday=0;
    for(int i=1;i<7;i++)
    {
        if(min>arr[i])
        {
            min=arr[i];
            minday=i;
        }

    }
    cout << "Number of hot days (above 30c): " << count << endl;
    string a[7]={"Monday","Tuesday","Wednesday","Thursday","Friday","saturday","sunday"};
    cout << "Coldest day: " << a[minday] << " with a temperature of " << min << "C" << endl;

    return 0;
}
```

Output:



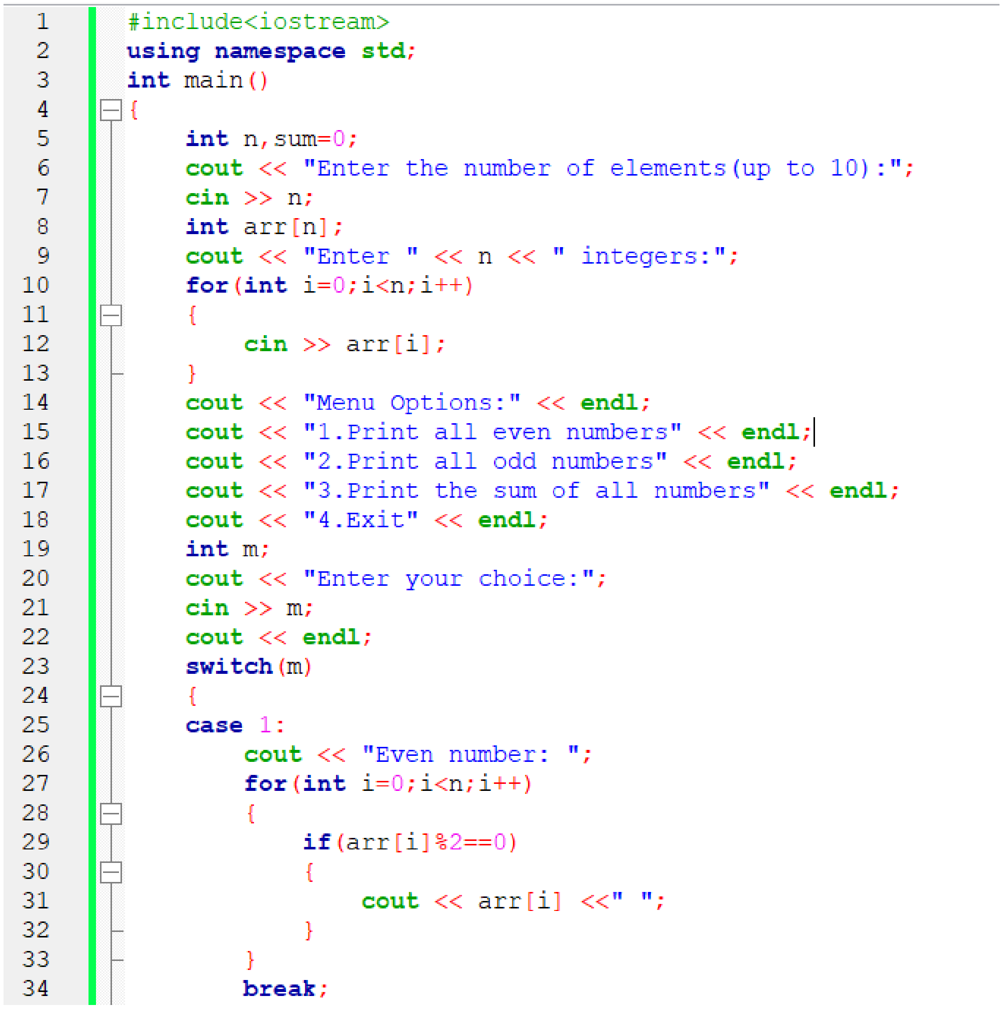
```
Enter Your Temperature : 89
98
77
87
90
87
88
Average temperature for the week: 88
Number of hot days (above 30c): 7
Coldest day: Wednesday with a temperature of 77C

Process returned 0 (0x0)   execution time : 17.443 s
Press any key to continue.
```

Quition:4

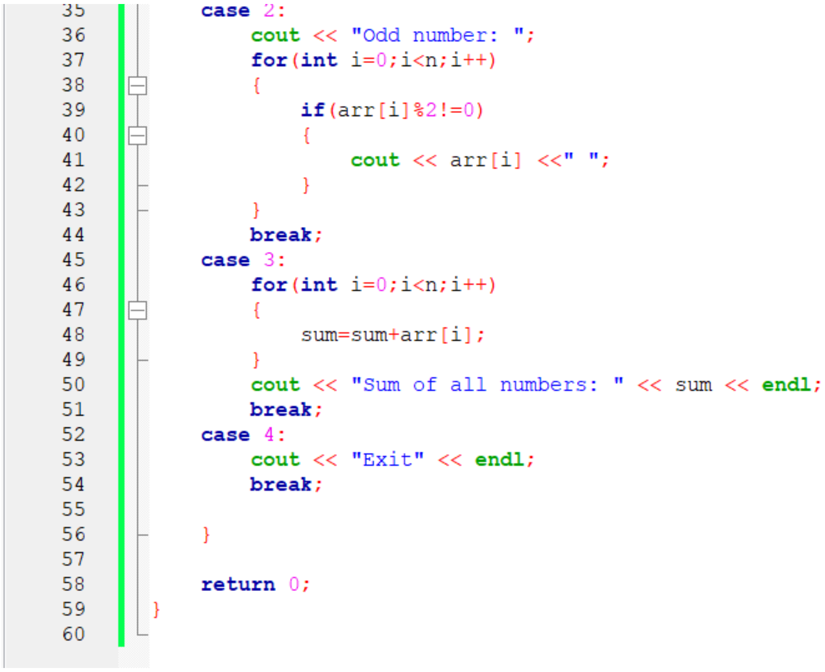
4. Write a program that asks the user to input a list of integers (up to 10) into a 1D array. The program should then:

1. Check if each number in the array is even or odd using a for loop and an if/else statement.
2. Allow the user to choose an option using a menu implemented with a switch-case statement:
    - **Option 1**: Print all even numbers in the array.
    - **Option 2**: Print all odd numbers in the array.
    - **Option 3**: Print the sum of all numbers in the array.
    - **Option 4**: Exit the program.

Answer:

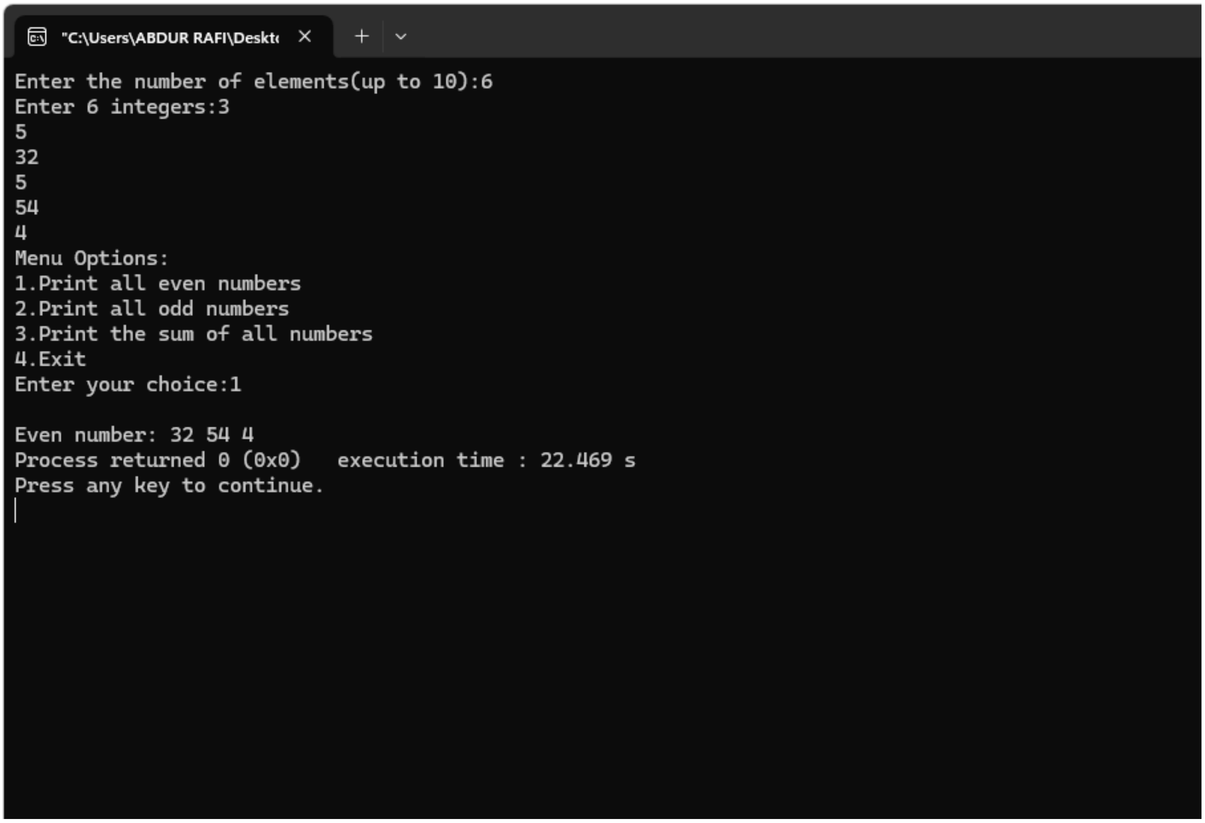


```cpp
#include<iostream>
using namespace std;
int main()
{
    int n,sum=0;
    cout << "Enter the number of elements(up to 10):";
    cin >> n;
    int arr[n];
    cout << "Enter " << n << " integers:";
    for(int i=0;i<n;i++)
    {
        cin >> arr[i];
    }
    cout << "Menu Options:" << endl;
    cout << "1.Print all even numbers" << endl;
    cout << "2.Print all odd numbers" << endl;
    cout << "3.Print the sum of all numbers" << endl;
    cout << "4.Exit" << endl;
    int m;
    cout << "Enter your choice:";
    cin >> m;
    cout << endl;
    switch(m)
    {
    case 1:
        cout << "Even number: ";
        for(int i=0;i<n;i++)
        {
            if(arr[i]%2==0)
            {
                cout << arr[i] <<" ";
            }
        }
        break;
```

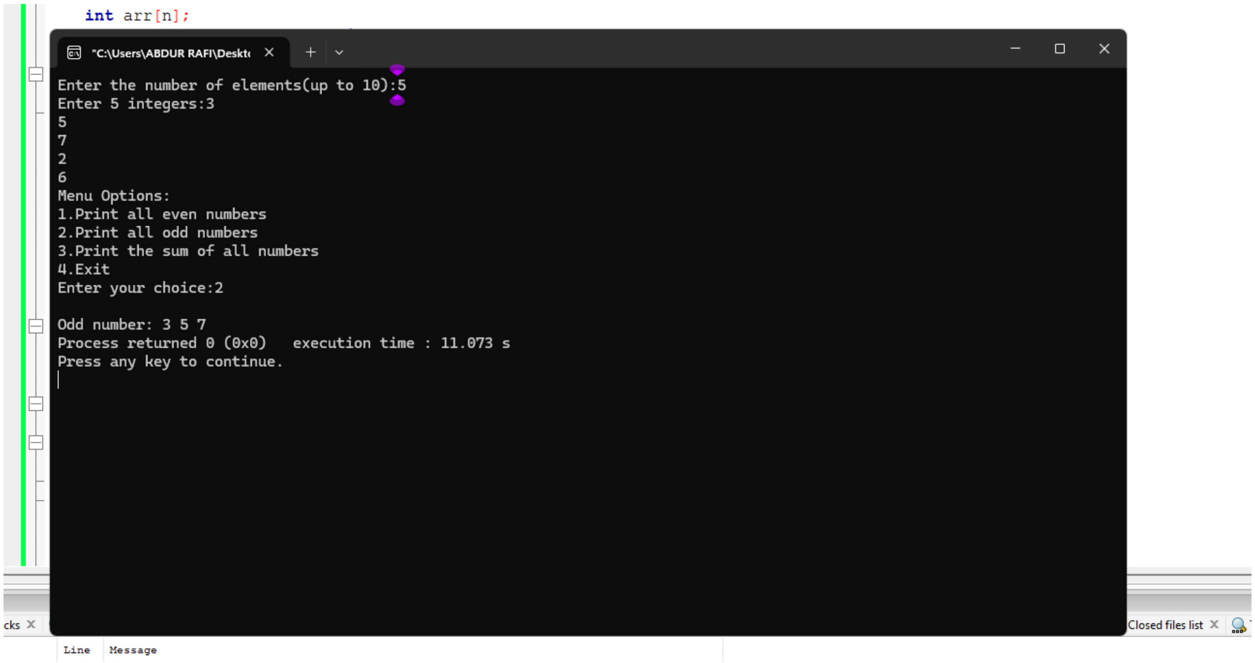


```cpp
    case 2:
        cout << "Odd number: ";
        for(int i=0;i<n;i++)
        {
            if(arr[i]%2!=0)
            {
                cout << arr[i] <<" ";
            }
        }
        break;
    case 3:
        for(int i=0;i<n;i++)
        {
            sum=sum+arr[i];
        }
        cout << "Sum of all numbers: " << sum << endl;
        break;
    case 4:
        cout << "Exit" << endl;
        break;

    }

    return 0;
}
```

Output:



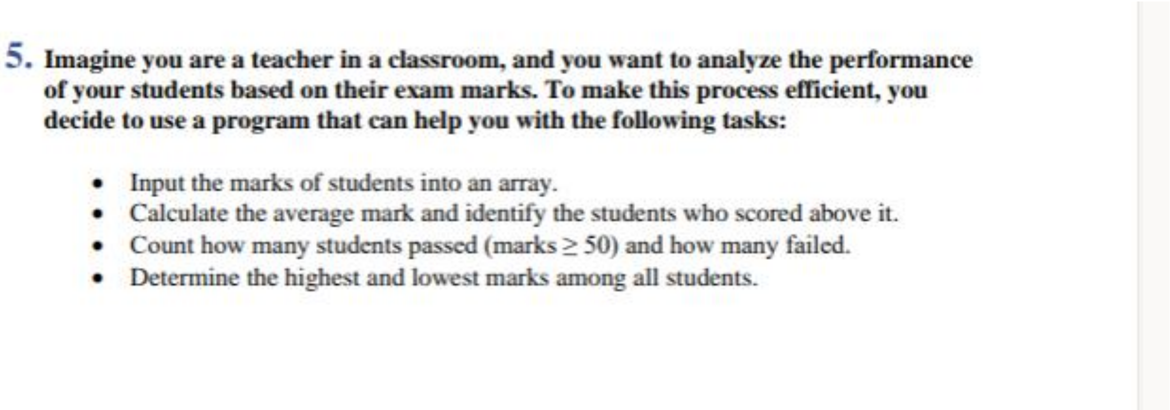
```
Enter the number of elements(up to 10):6
Enter 6 integers:3
5
32
5
54
4
Menu Options:
1.Print all even numbers
2.Print all odd numbers
3.Print the sum of all numbers
4.Exit
Enter your choice:1

Even number: 32 54 4
Process returned 0 (0x0)   execution time : 22.469 s
Press any key to continue.
```



```
    int arr[n];
```

```
Enter the number of elements(up to 10):5
Enter 5 integers:3
5
7
2
6
Menu Options:
1.Print all even numbers
2.Print all odd numbers
3.Print the sum of all numbers
4.Exit
Enter your choice:2

Odd number: 3 5 7
Process returned 0 (0x0)   execution time : 11.073 s
Press any key to continue.
```

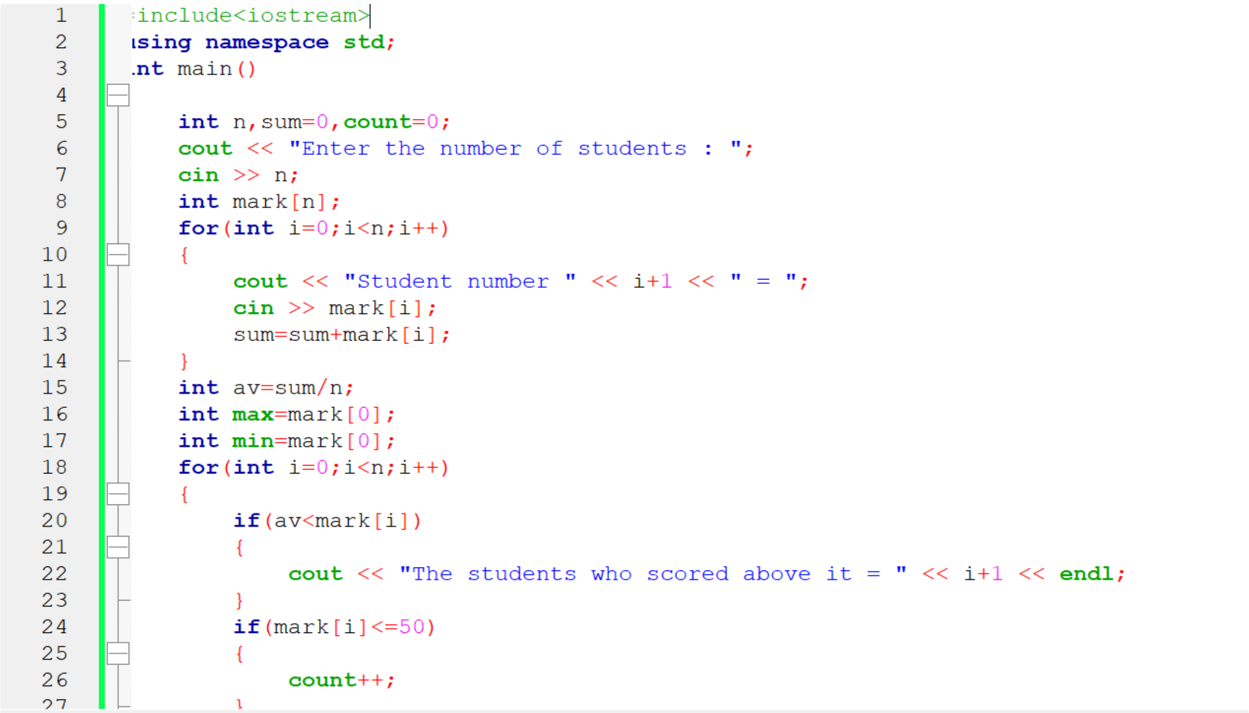
Quition:5



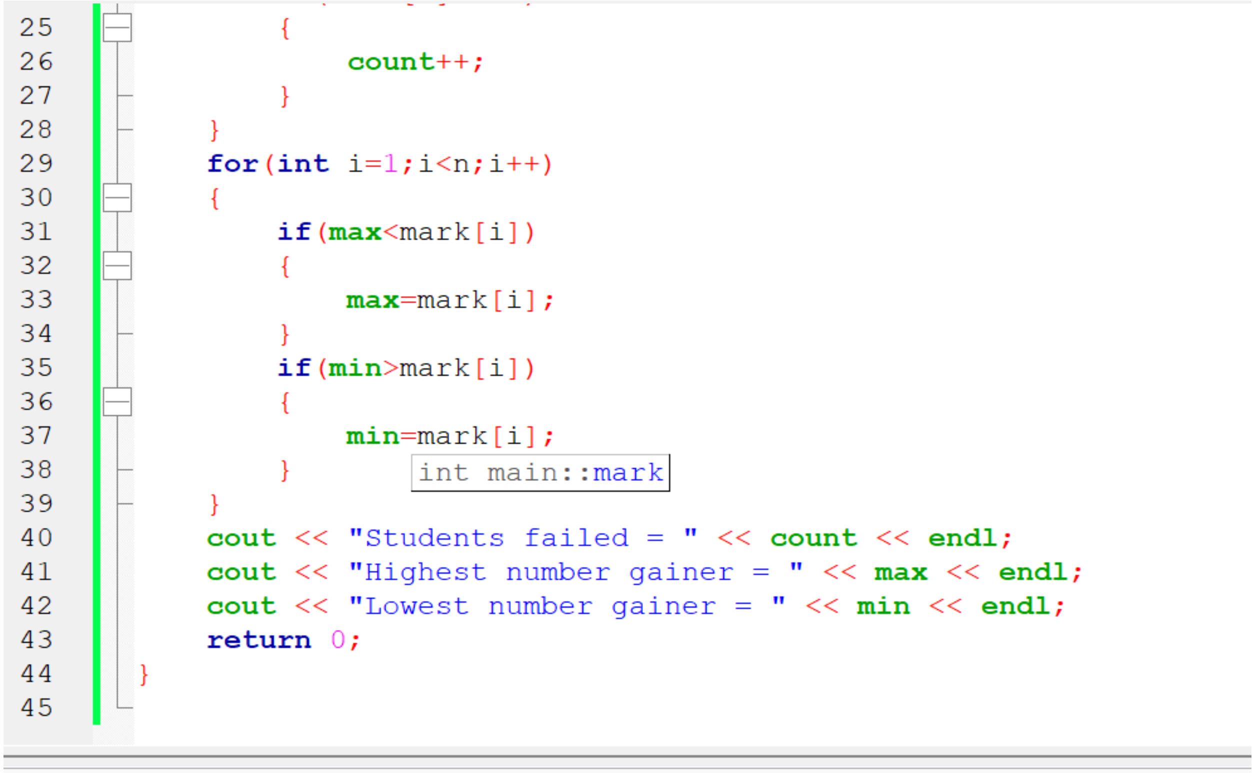
5. Imagine you are a teacher in a classroom, and you want to analyze the performance of your students based on their exam marks. To make this process efficient, you decide to use a program that can help you with the following tasks:

- Input the marks of students into an array.
- Calculate the average mark and identify the students who scored above it.
- Count how many students passed (marks ≥ 50) and how many failed.
- Determine the highest and lowest marks among all students.

Answer:

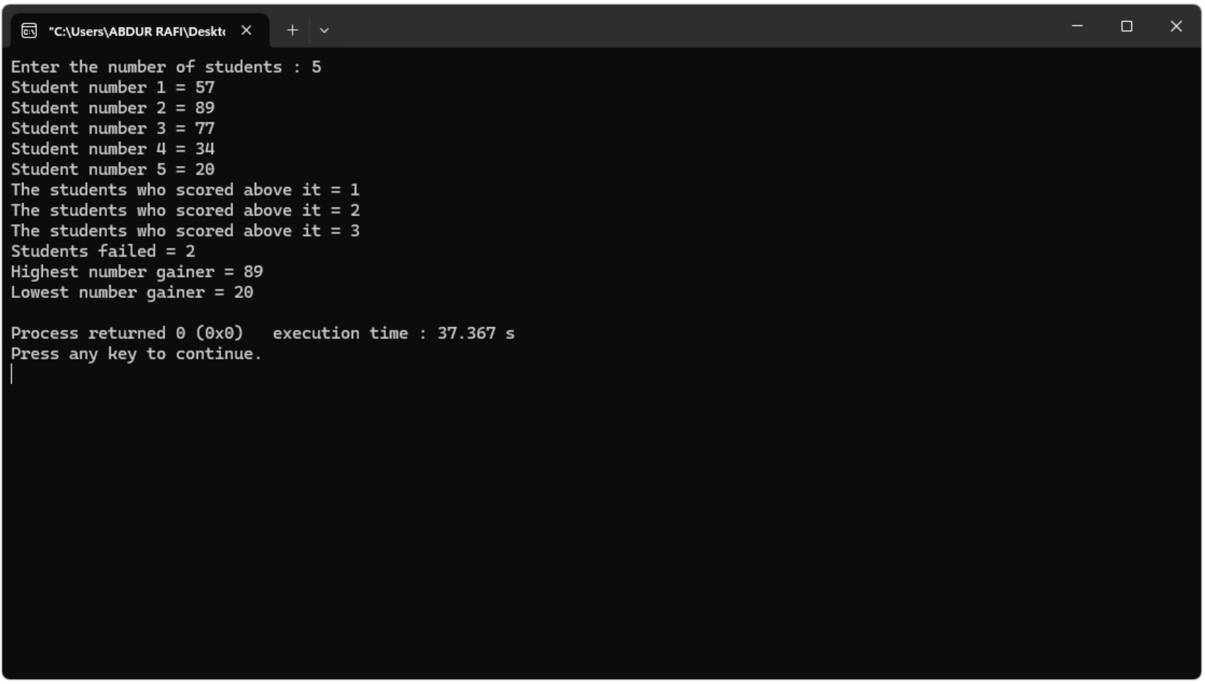


```
:include<iostream>
ısing namespace std;
.nt main()

    int n,sum=0,count=0;
    cout << "Enter the number of students : ";
    cin >> n;
    int mark[n];
    for(int i=0;i<n;i++)
    {
        cout << "Student number " << i+1 << " = ";
        cin >> mark[i];
        sum=sum+mark[i];
    }
    int av=sum/n;
    int max=mark[0];
    int min=mark[0];
    for(int i=0;i<n;i++)
    {
        if(av<mark[i])
        {
            cout << "The students who scored above it = " << i+1 << endl;
        }
        if(mark[i]<=50)
        {
            count++;
```



```
        {
            count++;
        }
    }
    for(int i=1;i<n;i++)
    {
        if(max<mark[i])
        {
            max=mark[i];
        }
        if(min>mark[i])
        {
            min=mark[i];
        }
    }
    cout << "Students failed = " << count << endl;
    cout << "Highest number gainer = " << max << endl;
    cout << "Lowest number gainer = " << min << endl;
    return 0;
}
```

Output:



```
Enter the number of students : 5
Student number 1 = 57
Student number 2 = 89
Student number 3 = 77
Student number 4 = 34
Student number 5 = 20
The students who scored above it = 1
The students who scored above it = 2
The students who scored above it = 3
Students failed = 2
Highest number gainer = 89
Lowest number gainer = 20

Process returned 0 (0x0)   execution time : 37.367 s
Press any key to continue.
```