The British College

Worksheet – 1

Name : Crishtina K.C.

Student ID : 23085130

Cyber Security And Digital Forensics

Github link:
https://github.com/crishtina01/cpp_Worksheet

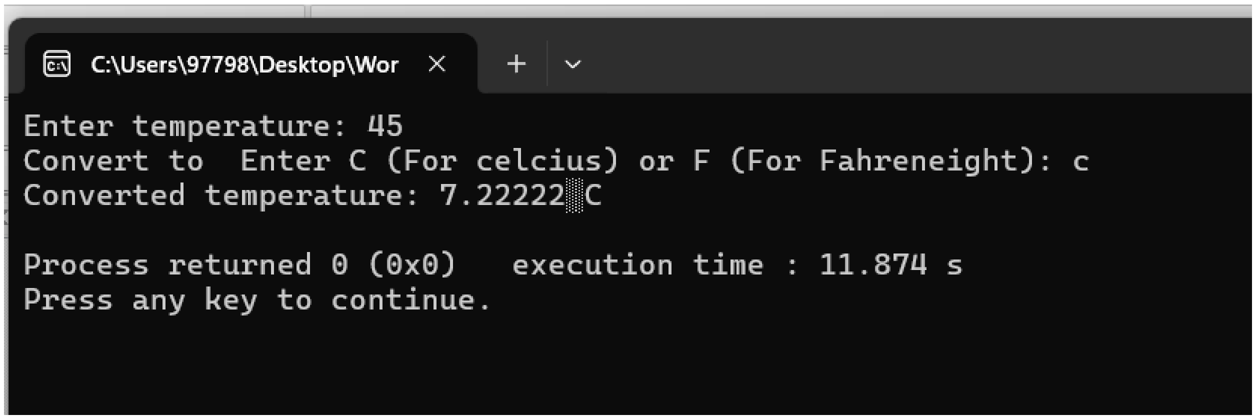
Task 1 : Programming Exercises:[Data types and Conditional Statements]

1. Write a program that takes a temperature value from the user. It should then allow the user to choose between Celsius (C) and Fahrenheit (F) for conversion. After the user selection, it should then convert the entered temperature to the chosen scale and display the result.
   Use appropriate data types for temperature and handle error like non-numeric input.
   Use the following formula for conversion:
   F = (C x 9/5) + 32
   C = (F - 32) x 5/9

```
#include <iostream>


#include <limits>
```

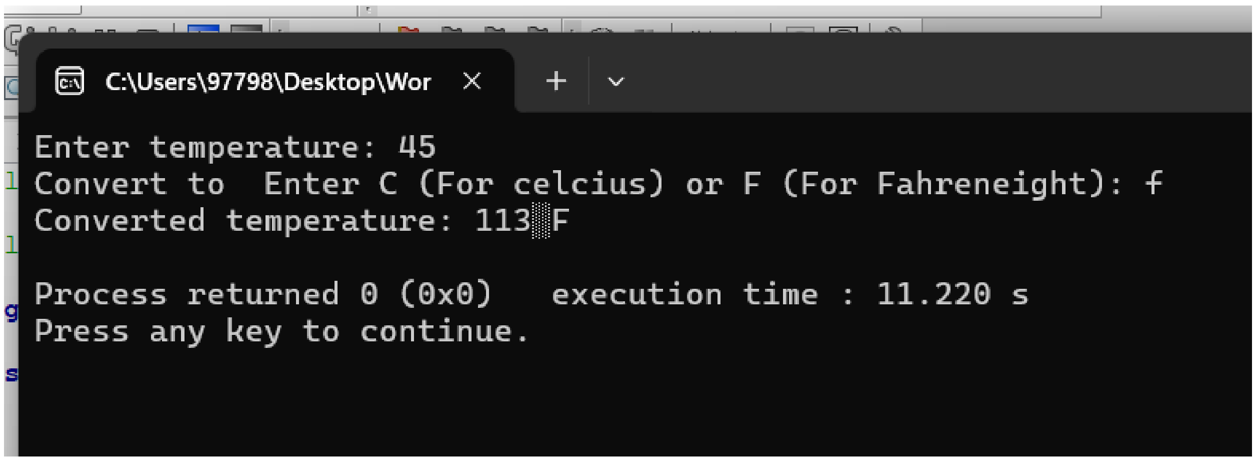
```
using namespace std;

class Converter

{

public:

   static double converttoFahrenheit(double temp)

   {

      return (temp * 9.0 / 5.0) + 32;

   }

   static double converttoCelsius(double temp)


   {

      return (temp - 32) * 5.0 / 9.0;

   }
};

int main()
```

```
{

  double temp;

  char choice;

  cout << "Enter temperature: ";

  while (!(cin >> temp))


    {

    cin.clear();

    cin.ignore(numeric_limits<std::streamsize>::max(), '\n');

    cout << "Invalid input. Enter a numeric value: ";

  }

  cout << "Convert to  Enter C (For celcius) or F (For Fahreneight): ";


  cin >> choice;


  choice = toupper(choice);
  if (choice == 'C')

    cout << "Converted temperature: " << Converter::converttoCelsius(temp) << "°C" << endl;

  else if (choice == 'F')

    cout << "Converted temperature: " << Converter::converttoFahrenheit(temp) << "°F" << endl;

  else

    cout << "Invalid choice." << endl;

  return 0;
}
```



C:\Users\97798\Desktop\Wor

```
Enter temperature: 45
Convert to  Enter C (For celcius) or F (For Fahreneight): c
Converted temperature: 7.22222░C

Process returned 0 (0x0)   execution time : 11.874 s
Press any key to continue.
```



C:\Users\97798\Desktop\Wor

```
Enter temperature: 45
Convert to  Enter C (For celcius) or F (For Fahreneight): f
Converted temperature: 113░F

Process returned 0 (0x0)   execution time : 11.220 s
Press any key to continue.
```

Task 1 : Programming Exercises:[Data types and Conditional Statements]

2. Write a C++ program to implement a number guessing game with different difficulty levels. Easy difficulty ranges from 1-8, medium from 1-30, hard from 1-50.Then,generate a random number to check if the guess is correct based on the user's selection.
 [10 marks]

```
#include <iostream>


#include <cstdlib>


#include <ctime>


using namespace std;


class generatenumber


{


public:


  static int generateRandomNumber(int maxRange)


  {


     return rand() % maxRange + 1;


  }


};
```

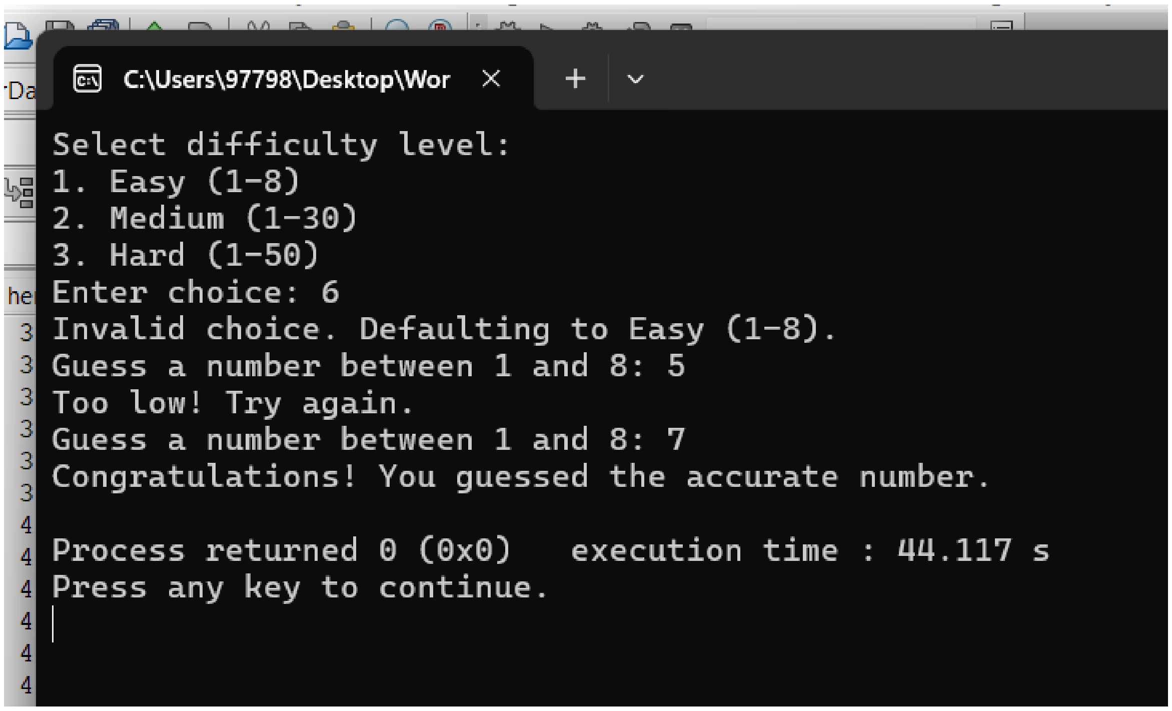
```
class UserData

{

public:

  static int getDifficultyLevel()

  {

    int choice;

    cout << "Select difficulty level:\n";

    cout << "1. Easy (1-8)\n";

    cout << "2. Medium (1-30)\n";

    cout << "3. Hard (1-50)\n";

    cout << "Enter choice: ";

    cin >> choice;

    return choice;

  }

  static int getyourGuess(int maxRange)
```

```
{
  int g;
  while (true)
  {
    cout << "Guess a number between 1 and " << maxRange << ": ";
    cin >> g;
    if (cin.fail() || g < 1 || g > maxRange)
    {
      cin.clear();
      cin.ignore(10000, '\n');
      cout << "Invalid input. Try again." << endl;
    }
    else
    {
      return g;
    }
```

```
        }
    }
};

class theGuessingGame

{

private:

    int maxRange;

    int randomNumber;

public:

    theGuessingGame(int range) : maxRange(range),
randomNumber(generatenumber::generateRandomNumber(range)) {}



    void play()

    {
        while (true)

        {
            int g = UserData::getyourGuess(maxRange);

            if (g == randomNumber)
            {
                cout << "Congratulations! You guessed the accurate number." << endl;
                break;

            }

            else if (g < randomNumber)

            {
                cout << "Too low! Try again." << endl;
            }

            else

            {
                cout << "Too high! Try again." << endl;
            }
        }
    }
};

int main()

{
    srand(time(0));

    int choice = UserData::getDifficultyLevel();
    int range;
```

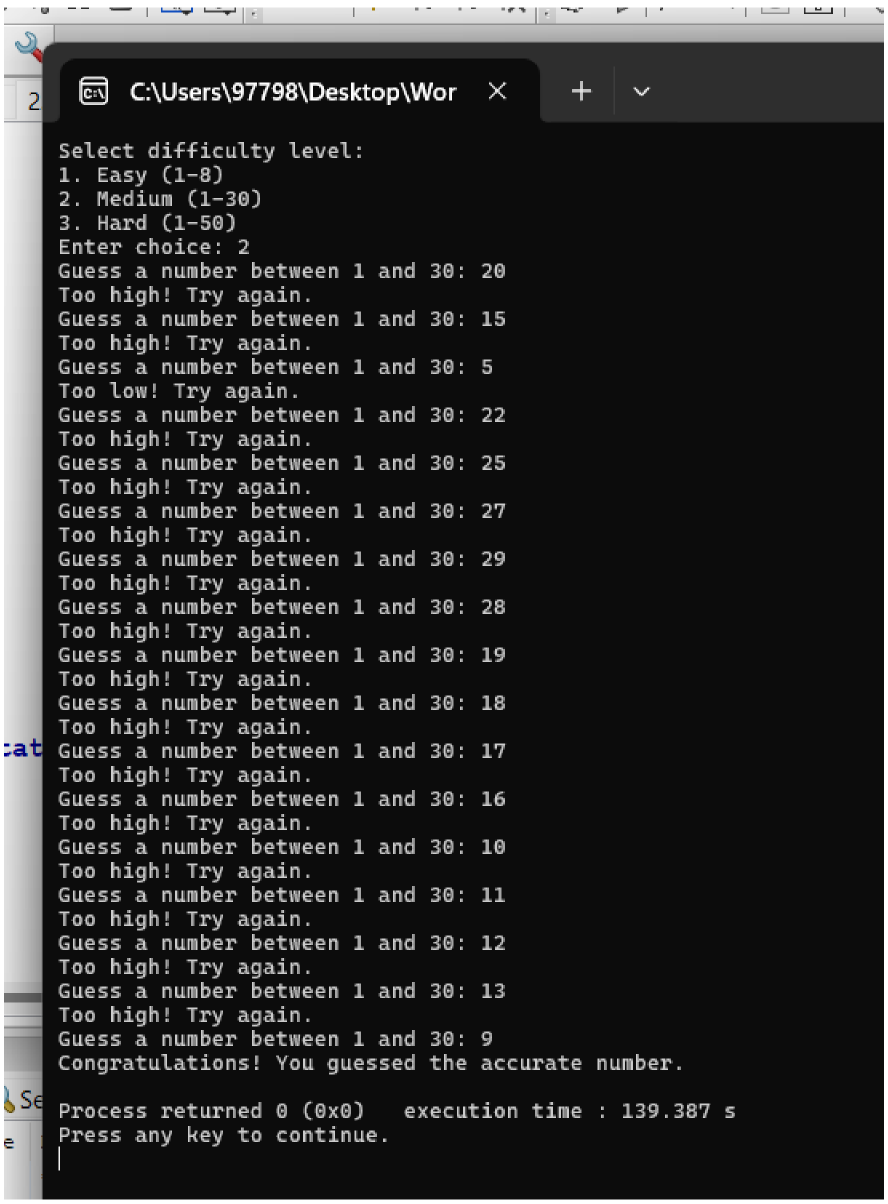
```
switch (choice)

{

case 1:
   range = 8;
   break;

case 2:
   range = 30;

   break;

case 3:
   range = 50;
   break;

default:

   cout << "Invalid choice. Defaulting to Easy (1-8)." << endl;

   range = 8;

}

theGuessingGame game(range);

game.play();
```

```
    return 0;

}
```

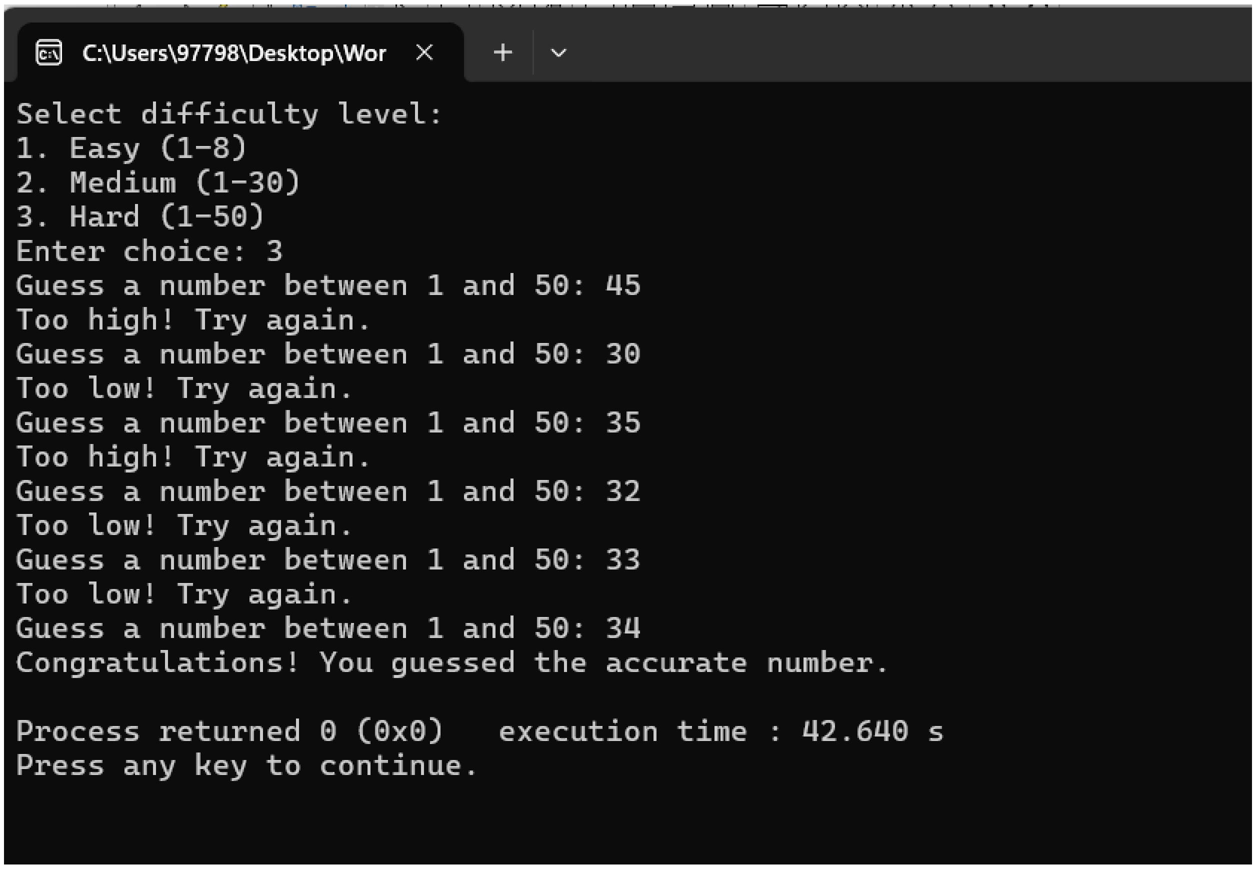


C:\Users\97798\Desktop\Wor

```
Select difficulty level:
1. Easy (1-8)
2. Medium (1-30)
3. Hard (1-50)
Enter choice: 6
Invalid choice. Defaulting to Easy (1-8).
Guess a number between 1 and 8: 5
Too low! Try again.
Guess a number between 1 and 8: 7
Congratulations! You guessed the accurate number.

Process returned 0 (0x0)   execution time : 44.117 s
Press any key to continue.
```



```
Select difficulty level:
1. Easy (1-8)
2. Medium (1-30)
3. Hard (1-50)
Enter choice: 2
Guess a number between 1 and 30: 20
Too high! Try again.
Guess a number between 1 and 30: 15
Too high! Try again.
Guess a number between 1 and 30: 5
Too low! Try again.
Guess a number between 1 and 30: 22
Too high! Try again.
Guess a number between 1 and 30: 25
Too high! Try again.
Guess a number between 1 and 30: 27
Too high! Try again.
Guess a number between 1 and 30: 29
Too high! Try again.
Guess a number between 1 and 30: 28
Too high! Try again.
Guess a number between 1 and 30: 19
Too high! Try again.
Guess a number between 1 and 30: 18
Too high! Try again.
Guess a number between 1 and 30: 17
Too high! Try again.
Guess a number between 1 and 30: 16
Too high! Try again.
Guess a number between 1 and 30: 10
Too high! Try again.
Guess a number between 1 and 30: 11
Too high! Try again.
Guess a number between 1 and 30: 12
Too high! Try again.
Guess a number between 1 and 30: 13
Too high! Try again.
Guess a number between 1 and 30: 9
Congratulations! You guessed the accurate number.

Process returned 0 (0x0)   execution time : 139.387 s
Press any key to continue.
```

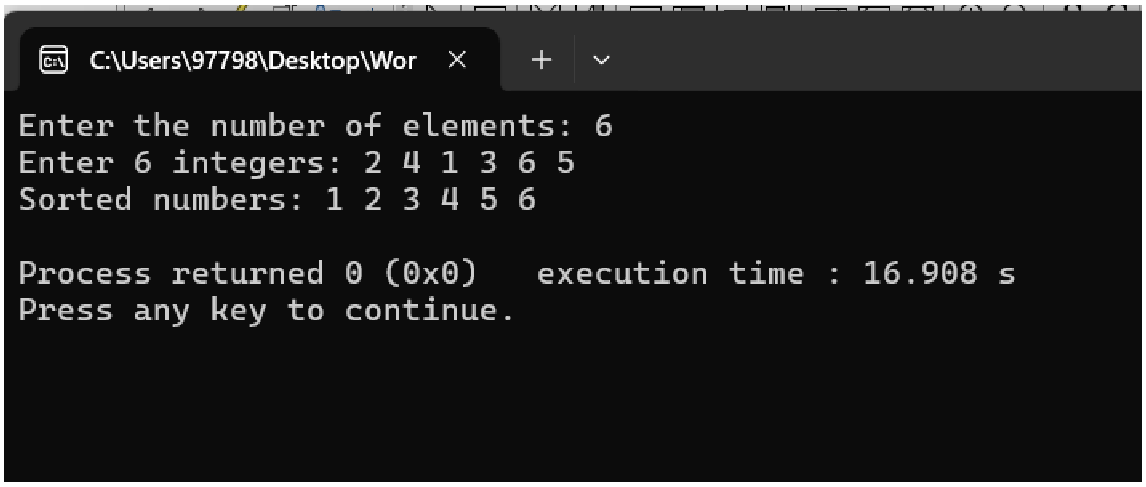


```
Select difficulty level:
1. Easy (1-8)
2. Medium (1-30)
3. Hard (1-50)
Enter choice: 3
Guess a number between 1 and 50: 45
Too high! Try again.
Guess a number between 1 and 50: 30
Too low! Try again.
Guess a number between 1 and 50: 35
Too high! Try again.
Guess a number between 1 and 50: 32
Too low! Try again.
Guess a number between 1 and 50: 33
Too low! Try again.
Guess a number between 1 and 50: 34
Congratulations! You guessed the accurate number.

Process returned 0 (0x0)   execution time : 42.640 s
Press any key to continue.
```

3. Write a program that reads an array of integer numbers from the user and sorts the numbers in the ascending order.
   [10 marks]

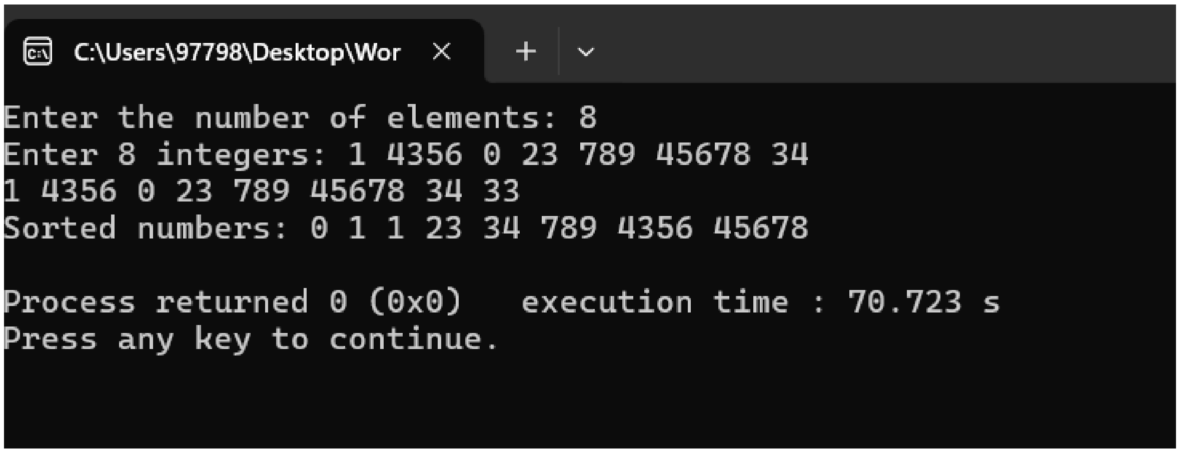
```
#include <iostream>

#include <vector>

#include <algorithm>

using namespace std;

class Sorter_Number

{

public:

  static void sortNumbers(vector<int>& numbers)
  {
      sort(numbers.begin(), numbers.end());
  }
};
class UserInput
{
public:
  static vector<int> getNumbers()
  {
      int num;
      cout << "Enter the number of elements: ";
      cin >> num;
      vector<int> numbers(num);
      cout << "Enter " << num << " integers: ";
      for (int i = 0; i < num; i++)
      {
          cin >> numbers[i];
      }
      return numbers;
  }
};
class NumberSorting_game
```

```
{

public:

   static void run()

   {
      vector<int> numbers = UserInput::getNumbers();

      Sorter_Number::sortNumbers(numbers);

      cout << "Sorted numbers: ";

      for (int num : numbers)

      {

         cout << num << " ";

      }

      cout << endl;
   }
};

int main()

{
   NumberSorting_game::run();

   return 0;

}
```



C:\Users\97798\Desktop\Wor

```
Enter the number of elements: 6
Enter 6 integers: 2 4 1 3 6 5
Sorted numbers: 1 2 3 4 5 6

Process returned 0 (0x0)   execution time : 16.908 s
Press any key to continue.
```



C:\Users\97798\Desktop\Wor

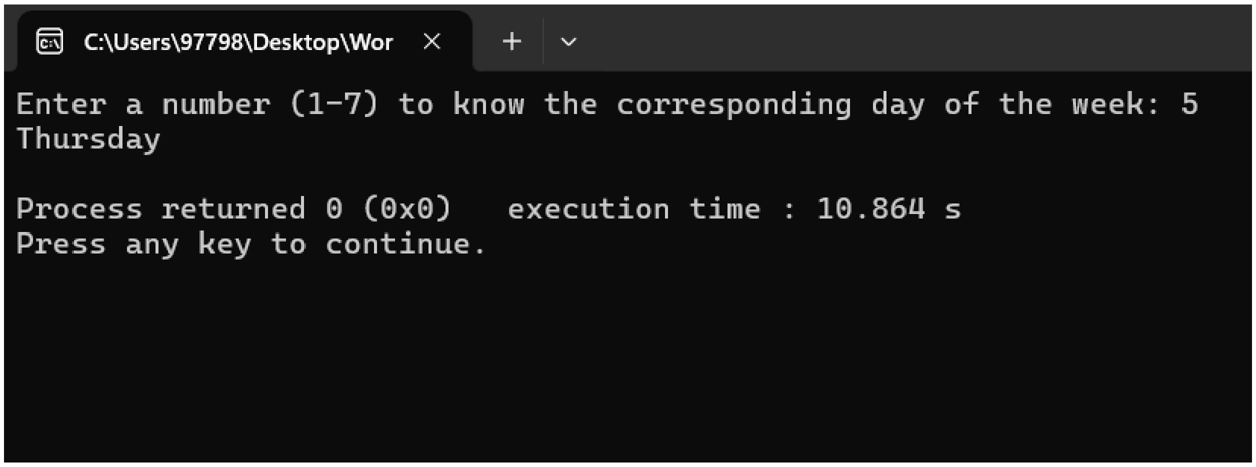
```
Enter the number of elements: 8
Enter 8 integers: 1 4356 0 23 789 45678 34
1 4356 0 23 789 45678 34 33
Sorted numbers: 0 1 1 23 34 789 4356 45678

Process returned 0 (0x0)   execution time : 70.723 s
Press any key to continue.
```

4. Write a program that reads a number from the user and based on the user input, it says what day of the week it is, Sundays being 1 and Saturdays being 7. You system should give appropriate response for invalid input entries. **[20 marks]**

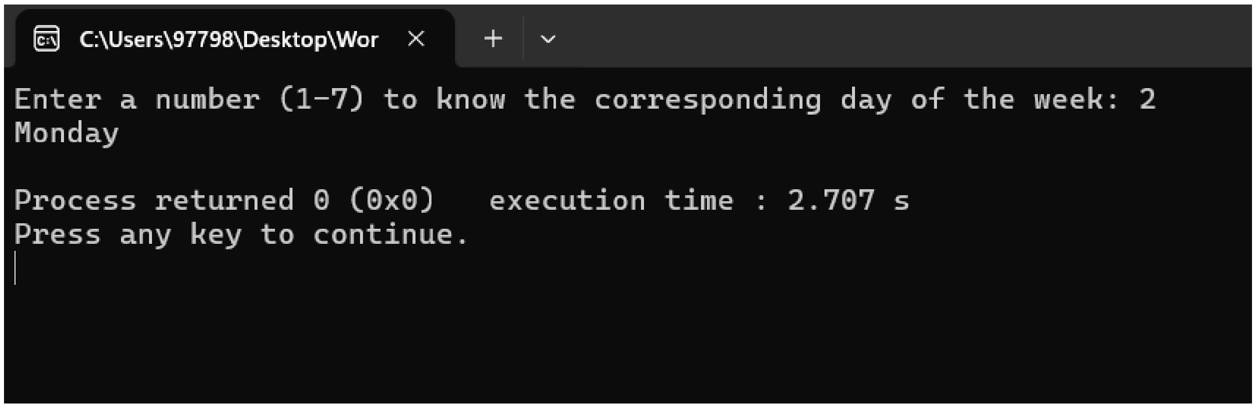
```
    #include <iostream>

using namespace std;

class DayOfWeek

{

public:
```

```
void get_Day(int day)

{

   switch (day)

   {

   case 1:

      cout << "Sunday" << endl;

      break;

   case 2:

      cout << "Monday" << endl;

      break;

   case 3:

      cout << "Tuesday" << endl;

      break;

   case 4:

      cout << "Wednesday" << endl;

      break;

   case 5:

      cout << "Thursday" << endl;

      break;

   case 6:

      cout << "Friday" << endl;

      break;
```

```
        case 7:

            cout << "Saturday" << endl;

            break;

        default:

            cout << "Invalid input. Enter a number between 1 and 7." << endl;

            break;

        }
    }
};

void getUserData()

{
    int day;

    cout << "Enter a number (1-7) to know the corresponding day of the week: ";

    cin >> day;

    DayOfWeek dayOfWeek;

    dayOfWeek.get_Day(day);
}

int main()
{
    getUserData();

    return 0;
}
```



```
Enter a number (1-7) to know the corresponding day of the week: 5
Thursday

Process returned 0 (0x0)   execution time : 10.864 s
Press any key to continue.
```



```
Enter a number (1-7) to know the corresponding day of the week: 2
Monday

Process returned 0 (0x0)   execution time : 2.707 s
Press any key to continue.
```

Task 2: Programming Exercises:[Control Statements]

1. Create a program that takes a positive integer as input and determines whether it's a "bouncy number". A bouncy number is one where the digits neither consistently increase nor consistently decrease when read from left to right. For example:

- 123 is NOT bouncy (digits consistently increase)
- 321 is NOT bouncy (digits consistently decrease)
- 120 is bouncy (neither consistently increasing nor decreasing)

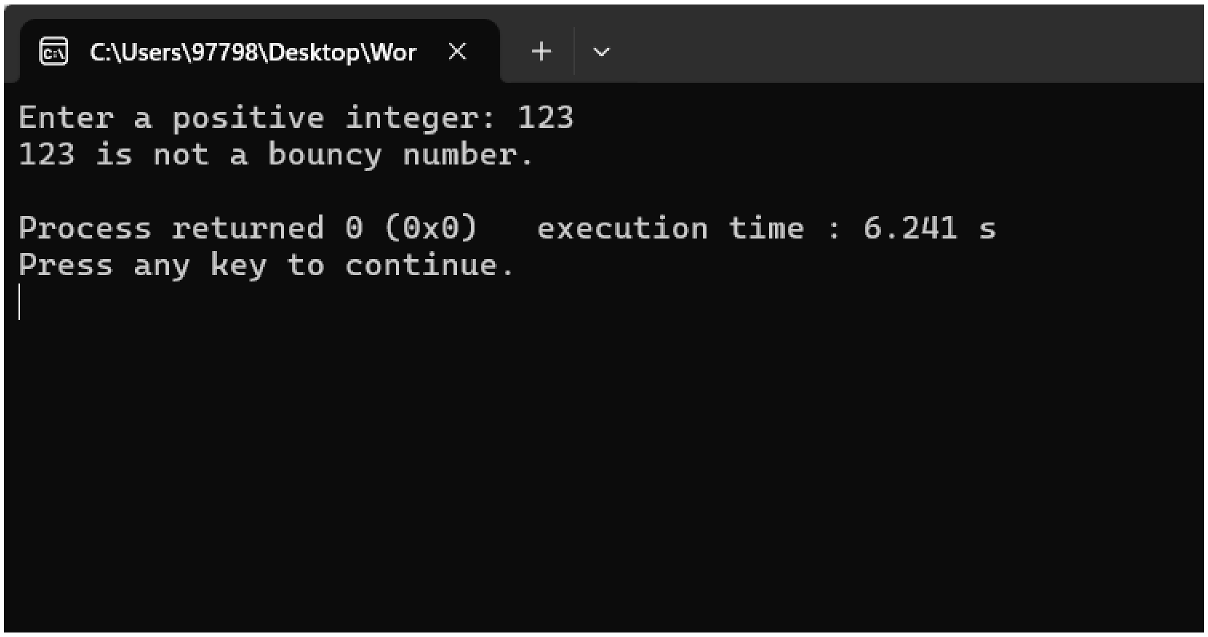
```
#include <iostream>

#include <vector>

using namespace std;
```

```
class Bouncy_Number

{


public:

    bool isBouncy(int number)

    {

        vector<int> digits = getDigits(number);

        if (digits.size() < 3)


            {

                return false;

            }

        bool increasing = false;

        bool decreasing = false;

        for (size_t i = 1; i < digits.size(); ++i)

            {
```

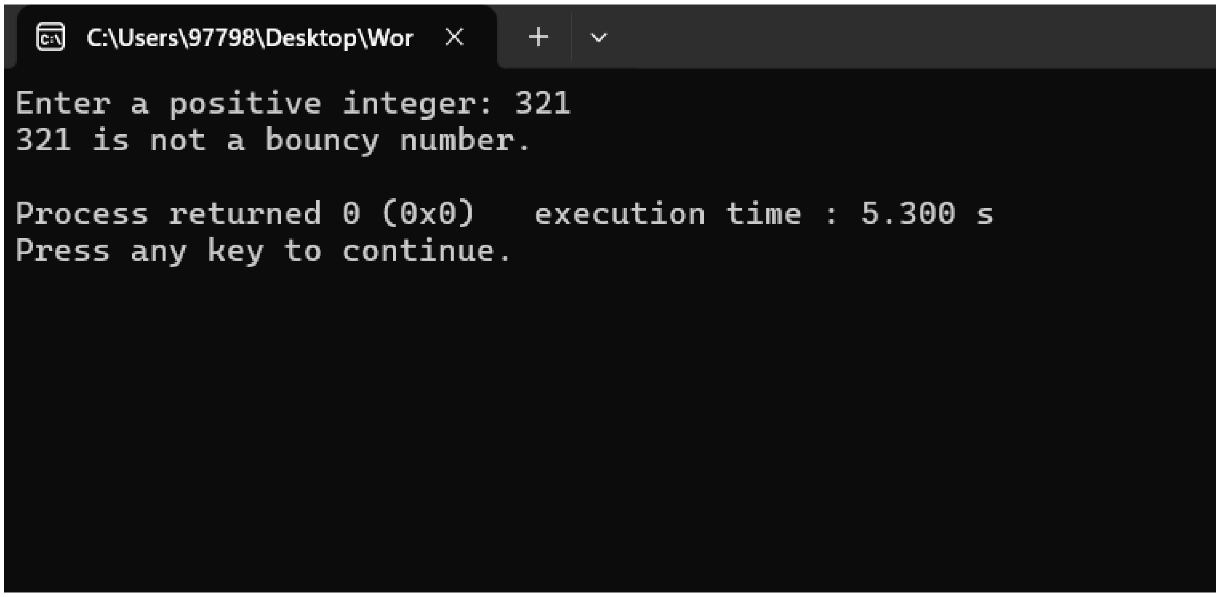
```
    if (digits[i] > digits[i - 1])

      {
          increasing = true;

      }

    else if (digits[i] < digits[i - 1])


      {

          decreasing = true;

      }

    if
      (increasing && decreasing)


          {

              return true;

          }

}
```

```
        return false;
    }

private:

    vector<int> getDigits(int number)

    {

        vector<int> digits;

        while (number > 0)

        {

            digits.insert(digits.begin(), number % 10);

            number /= 10;

        }

        return digits;

    }
};

int main()
```

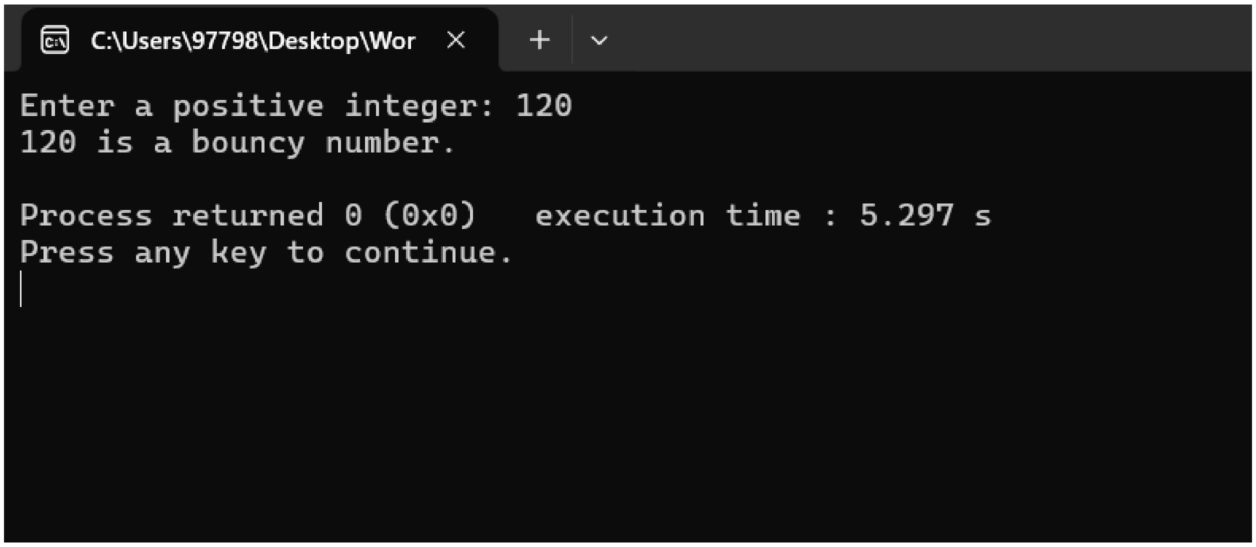
```
{
   int number;

   cout << "Enter a positive integer: ";

   cin >> number;

   Bouncy_Number bouncyChecker;

   if (bouncyChecker.isBouncy(number))

      {

         cout << number << " is a bouncy number." << endl;

      }

   else


      {

         cout << number << " is not a bouncy number." << endl;

      }

   return 0;
}
```


C:\Users\97798\Desktop\Wor

```
Enter a positive integer: 123
123 is not a bouncy number.

Process returned 0 (0x0)   execution time : 6.241 s
Press any key to continue.
```


C:\Users\97798\Desktop\Wor

```
Enter a positive integer: 321
321 is not a bouncy number.

Process returned 0 (0x0)   execution time : 5.300 s
Press any key to continue.
```



```
Enter a positive integer: 120
120 is a bouncy number.

Process returned 0 (0x0)   execution time : 5.297 s
Press any key to continue.
```

Task 3: Programming Exercises on Arrays

1. Write a program that manages a cinema ticket booking system. The program should display a 5x5 seating arrangement where: **[25 marks]**

    1. Available seats are marked with 'O'
    2. Booked seats are marked with 'X'

Program should:

1. Display the current seating arrangement
2. Ask user for row and column number (1-5) for booking
3. Mark that seat as booked ('X')
4. Show updated seating after each booking
5. Display error if user selects already booked seat
6. Display error if user enters invalid row/column numbers

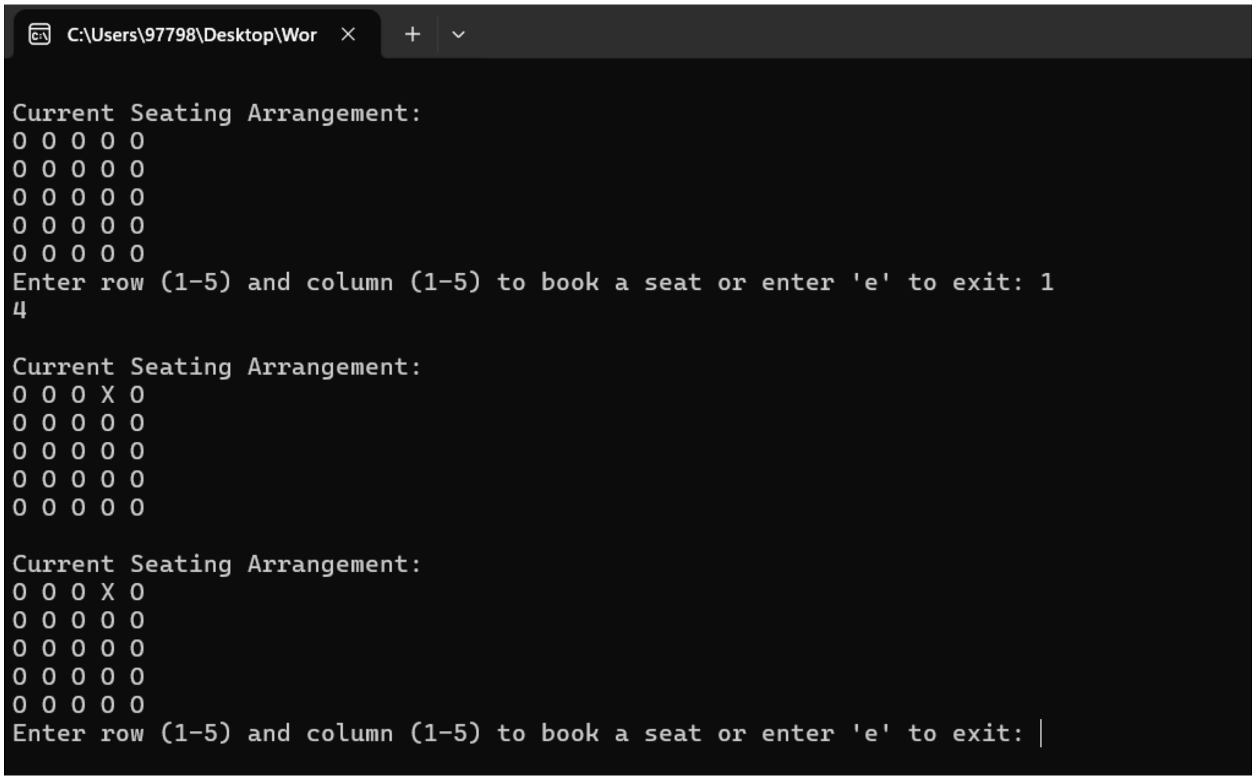
```
#include <iostream>


#include <limits>


using namespace std;
```

```
class Cinema

{

private:

    char seats[5][5];



public:

    Cinema()



    {

        for (int i = 0; i < 5; ++i)



            {

            for (int j = 0; j < 5; ++j)

            {

                seats[i][j] = 'O';
    }



  }
}

void displaySeating()

{
    cout << "\nCurrent Seating Arrangement:\n";

  for (int i = 0; i < 5; ++i)

      {

    for (int j = 0; j < 5; ++j)

      {

        cout << seats[i][j] << " ";

      }

        cout << endl;

  }

}
```

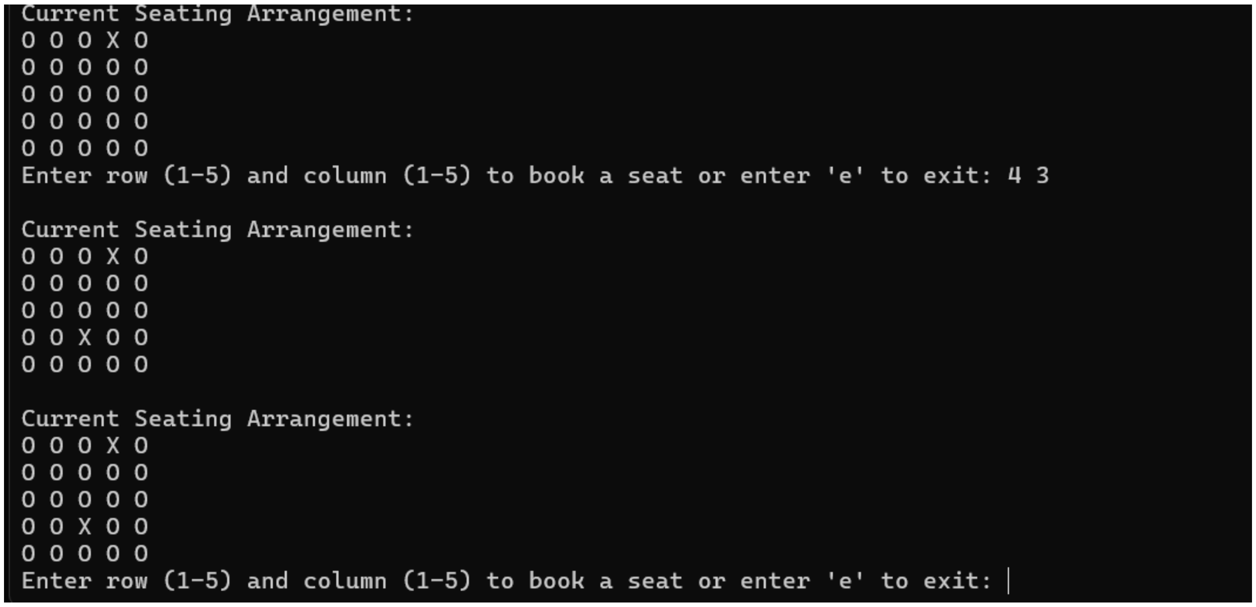
```
bool bookSeat(int row, int col)

  {
    row -= 1;

    col -= 1;

  if (row < 0 || row >= 5 || col < 0 || col >= 5)

    {
      cout << " Error: Invalid seat number. Enter row and column between 1 and 5.\n";

      return false;
    }

  if (seats[row][col] == 'X')



      {

        cout << " Error: This seat is already booked. Choose a different seat.\n";



      return false;

  }

  seats[row][col] = 'X';
```

```
        return true;

    }
};


void Bookingsystem(Cinema &cinema)

    {
        int row, col;

        bool bookingSuccessful;

        char exitOption;


    while (true)

        {

            cinema.displaySeating();

            cout << "Enter row (1-5) and column (1-5) to book a seat or enter 'e' to exit: ";


        if (!(cin >> row))

            {
```

```
    cin.clear();

    cin >> exitOption;

    if (exitOption == 'e')

      {
        cout << "Exiting the program.\n";
        break;
      }

    cin.ignore(numeric_limits<streamsize>::max(), '\n');

    continue;
  }


  if (!(cin >> col))

    {

      cin.clear();

      cin.ignore(numeric_limits<streamsize>::max(), '\n');

      cout << "Error: Invalid input for column. Enter valid numbers.\n";

      continue;
      }

      bookingSuccessful = cinema.bookSeat(row, col);

      if (bookingSuccessful)
        {
        cinema.displaySeating();
      }
    }
  }

  int main()
  {
    Cinema B2;
    Bookingsystem(B2);
    return 0;
  }
```



```
C:\Users\97798\Desktop\Wor

Current Seating Arrangement:
O O O O O
O O O O O
O O O O O
O O O O O
O O O O O
Enter row (1-5) and column (1-5) to book a seat or enter 'e' to exit: 1
4

Current Seating Arrangement:
O O O X O
O O O O O
O O O O O
O O O O O
O O O O O

Current Seating Arrangement:
O O O X O
O O O O O
O O O O O
O O O O O
O O O O O
Enter row (1-5) and column (1-5) to book a seat or enter 'e' to exit:
```



```
Current Seating Arrangement:
O O O X O
O O O O O
O O O O O
O O O O O
O O O O O
Enter row (1-5) and column (1-5) to book a seat or enter 'e' to exit: 4 3

Current Seating Arrangement:
O O O X O
O O O O O
O O O O O
O O X O O
O O O O O

Current Seating Arrangement:
O O O X O
O O O O O
O O O O O
O O X O O
O O O O O
Enter row (1-5) and column (1-5) to book a seat or enter 'e' to exit:
```