­­

**CODE**

#include <bits/stdc++.h>

using namespace std;

class guess;//forward declaration

class Random // class random

{

private:

    int r;

public:

    Random() // generating a random number

    {

        srand(time(NULL));//seeding null

        r = 1 + (rand() % 1000);//generating between 1-1000

    }

    friend class guess; // friend class

};

class guess // class guess

{

private:

    int input;

    int k;

    Random \*j ; // dynamic initialization

    string playername;

public:

    guess()

    {

        input = 0;

        k = 0;

    }

    friend void operator >>(istream &in,guess &Number\_game)

    {

        details(Number\_game);

        Number\_game.guessing();

    }

    void guessing() // taking input from user

    {

        char ch;

        j=new Random;

        k=0;

        do

        {

            cout << "Guess the number between 1 and 1000!!!" << endl;

            cin >> input;

            k++;

            if (input < j->r)

                cout << "Too low try again!!" << endl;

            else if (input > j->r)

                cout << "Too high try again!!" << endl;

            else

            {

                int cnt = 0;

                cout << "Excellent you guessed the number would you like to play again!!\n" << endl;

            label://to get back to statement if user does not enter y/n

                cout << "press Y to try again and N to exit the game\n";

                cin >> ch;

                if (ch == 'Y' || ch == 'y')

                {

                    guessing();

                }

                else if (ch == 'N' || ch == 'n')

                {

                    break;

                }

                else

                {

                    cout << "enter a valid choice\n";

                    cnt++;

                if (cnt >= 3)

                {

                    cout << "Exiting the game!!" << endl; // win + ;😎

                    exit(0);

                }

                    goto label;

                }

            }

        } while (input != j->r);

        if (k < 10)

        {

            cout << "Either you know the secret or you got lucky!!\n" << endl;

            cout << "You just guessed in " << k << " attempts\n " << endl;

        }

        else if (k == 10)

        {

            cout << "haha you know the secret!!\n" << endl;

            cout << "You just guessed in " << k << " attempts \n" << endl;

        }

        else

        {

            try\_again();

        }

    }

    void try\_again() // to ask if user want to play again

    {

        int count = 0;

    to:

        cout << "you should be able to do better why should it take no more than 10 guesses!!\n" << endl;

        cout << "Do you want to try again ??\nEnter Y/N \n" << endl;

        char choice;

        cin >> choice;

        if (choice == 'Y' || choice == 'y')

        {

            k = 0;

            guessing();

        }

        else if (choice == 'N' || choice == 'n')

        {

            cout << "Do you want the  solution for the game Y/N" << endl;

            cin >> choice;

            if (choice == 'Y' || choice == 'y')

            {

                solution();

            }

            else if (choice == 'N' || choice == 'n')

            {

                cout << "Better luck next time!!" << endl;

                exit(0);

            }

            else{

                cout<<"Wrong input\nExiting game!!\n";

            }

        }

        else

        {

            cout << "You entered a wrong input" << endl;

            count++;

            if (count > 3)

            {

                cout << "Since Player is not entering right input.\n" << endl;

                cout << "We are providing the solution \n" << endl;

                solution();

                exit(0);

            }

            goto to;

        }

    }

    void solution()               // step by step solution to guess the number in 10 guesses

    {

        cout << "Our number was " << j->r;

        cout << " So the following steps should have been followed for the solution" << endl

             << endl;

        int low = 1, mid, high = 1000;

        int counter = 0;

        do

        {

            mid = (low + high) / 2;

            cout << "Take: " << mid << endl;

            if (mid == j->r)

            {

                break;

            }

            else if (mid < j->r)

            {

                cout << counter + 1 << ".The number is too low so take a number between " << mid<< " & " << high << endl;

                low = mid + 1;

            }

            else if (j->r < mid)

            {

                cout << counter + 1 << ".The number is too high so take a number between " << low << " & " << mid << endl;

                high = mid - 1;

            }

            counter++;

        } while (mid != j->r);

        cout << "So we found " << mid << " in " << counter << " attempts :)" << endl;

    }

    ~guess() // destructor

    {

        cout << "Game is Over" << endl;

    }

    friend void details(guess &number);

};

void details(guess &number){

    cout<<"Enter the player name\n";

    string nam;

    getline(cin,nam);

    number.playername=nam;

    cout<<"Welcome "<<number.playername<<endl;

}

int main()

{

    // random guess;

    guess number;

    cin>>number;

    // details(number);

    // number.guessing();

    return 0;

}

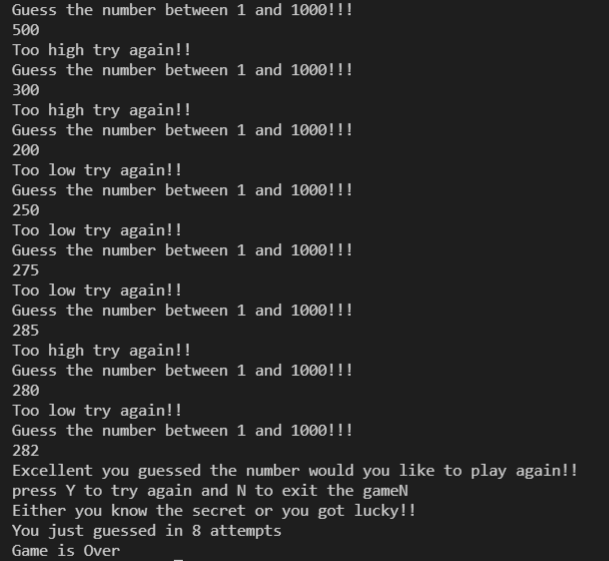
**CLASS RANDOM**

* **Creates a random number between 1 to 1000 using rand() [inbuilt in C++]**
* **srand time is set to NULL to get different number in every game**
* **Has a friend class “guess“**

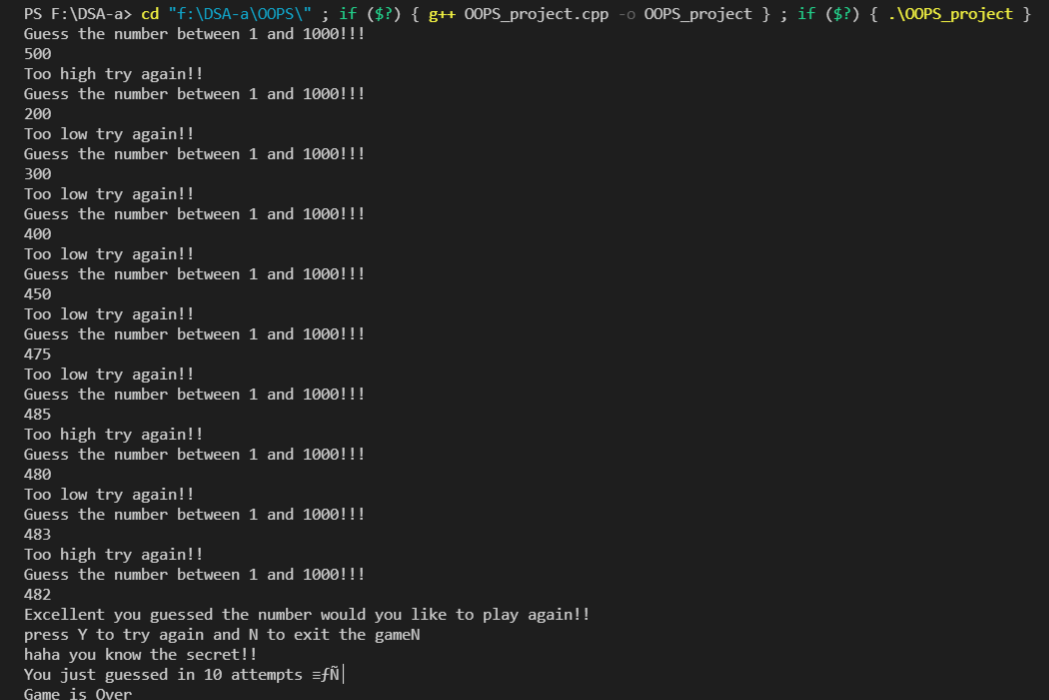
**CLASS GUESS**

* **Dynamically initializes the class random ,hence creating a random number. Also initializes the count to 0.**
* **Member function “guessing()”**
  + - **Takes the input from user along with taking the account of number of inputs while providing the required hints to guess the number**
    - **The program continues to take the input till the user gets the right number.**
    - **If the number of inputs>10 the program provide the choice of playing the game again or getting the solution .**
* **Member function “try\_again()”**
  + - **Provides a replay functionality to the user.**
    - **Asks the user for an efficient solution of the game**
* **Member function “solution()”**
  + - **Provides an efficient solution of the game with concept of binary search within an input span of 10**
* **Friend function “details()”**
* **Takes user’s details as input**
* **Provides friend functionality**

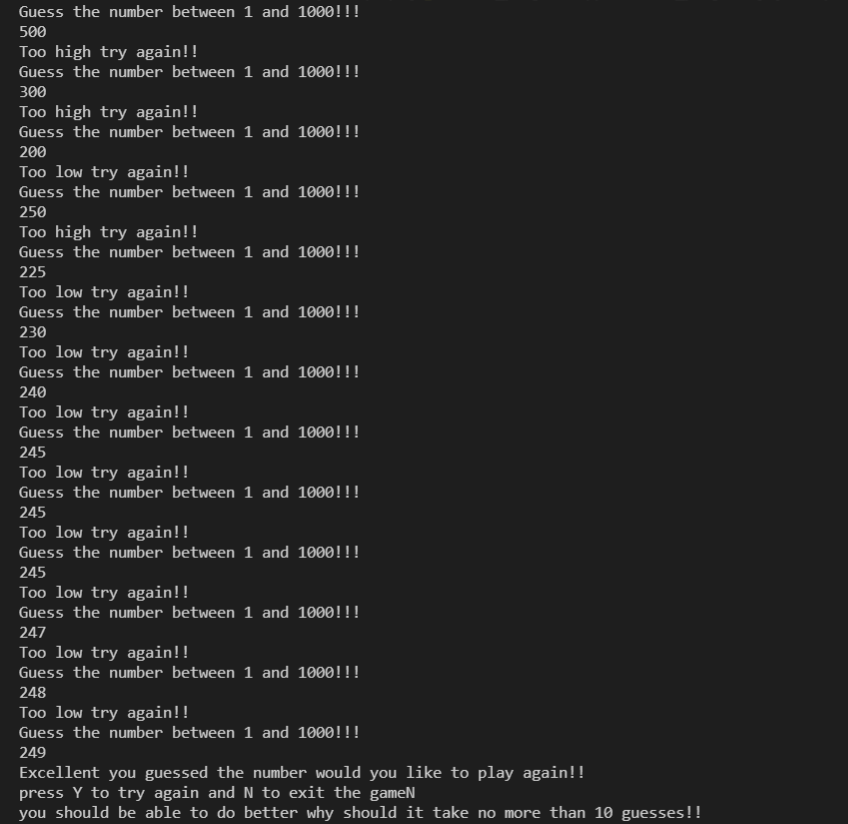
**OUTPUTS:**

**1.) **

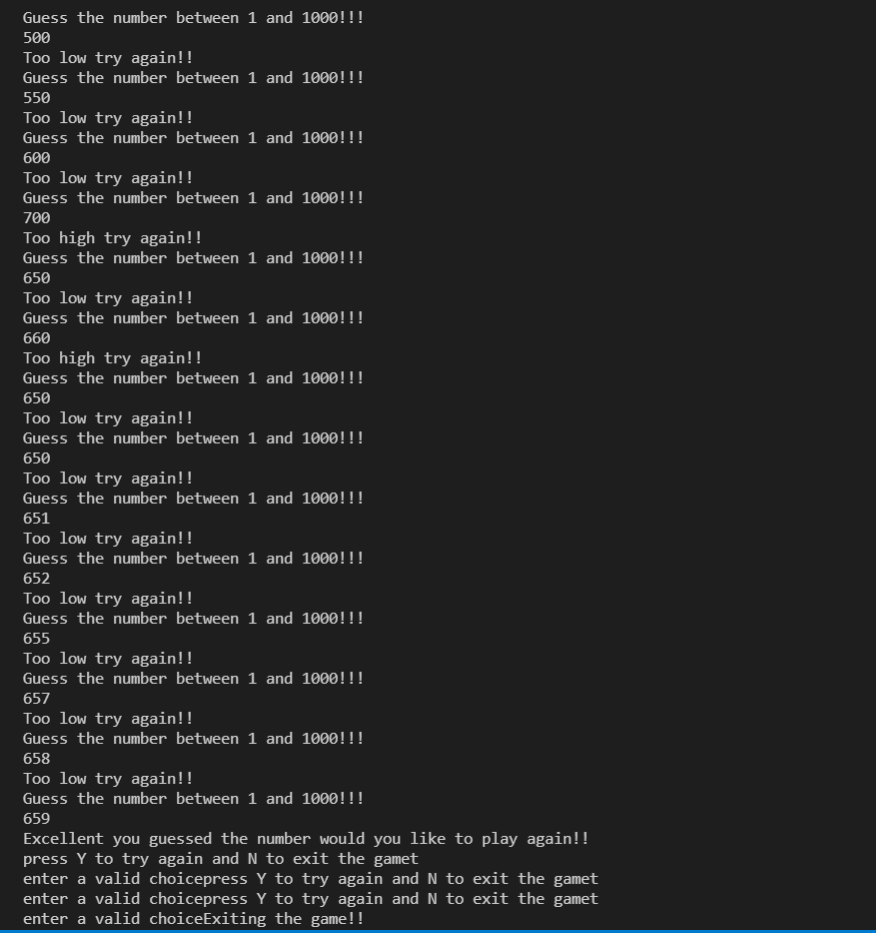
**2.)**

****

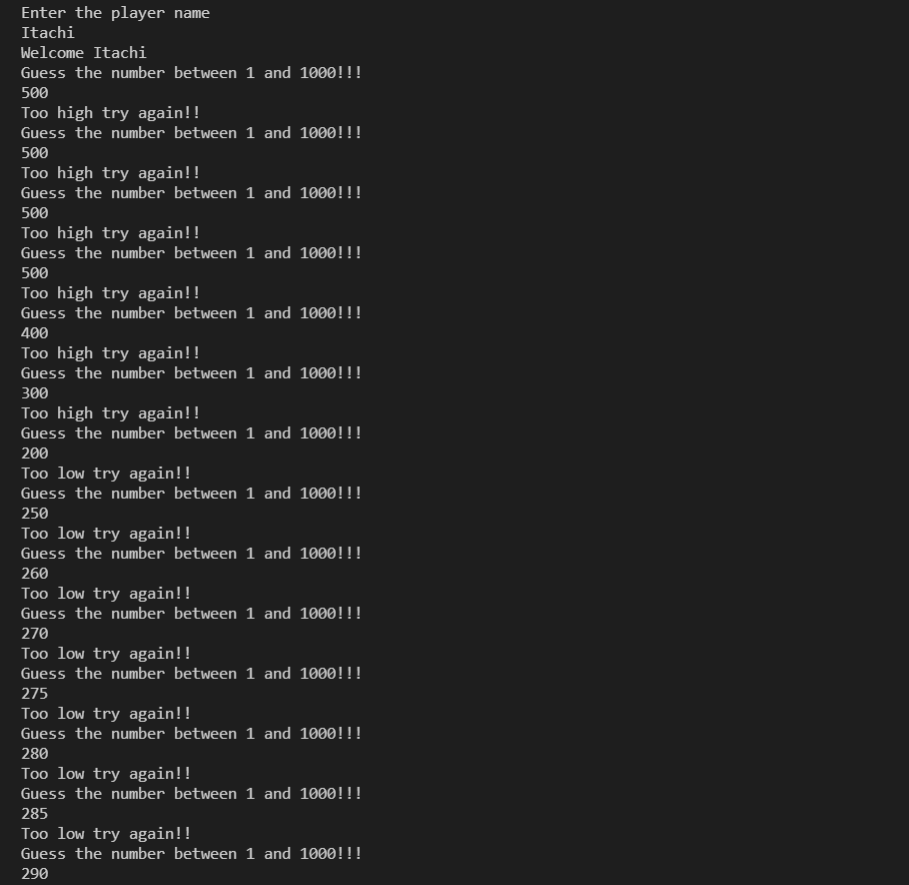
**3.)**

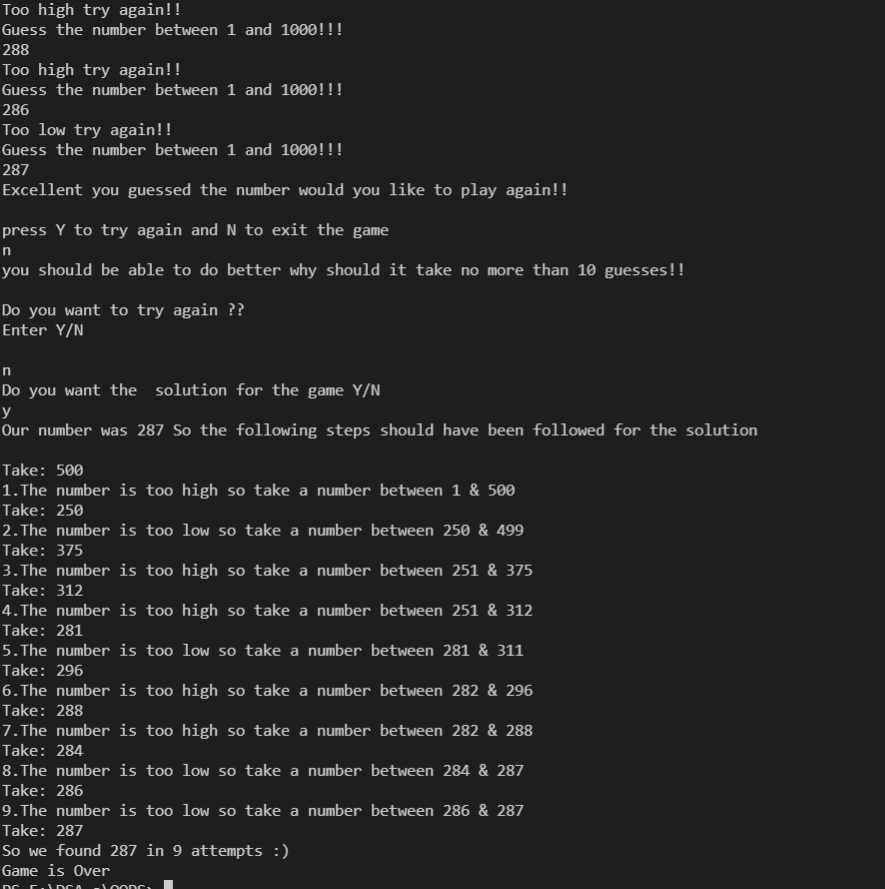
****

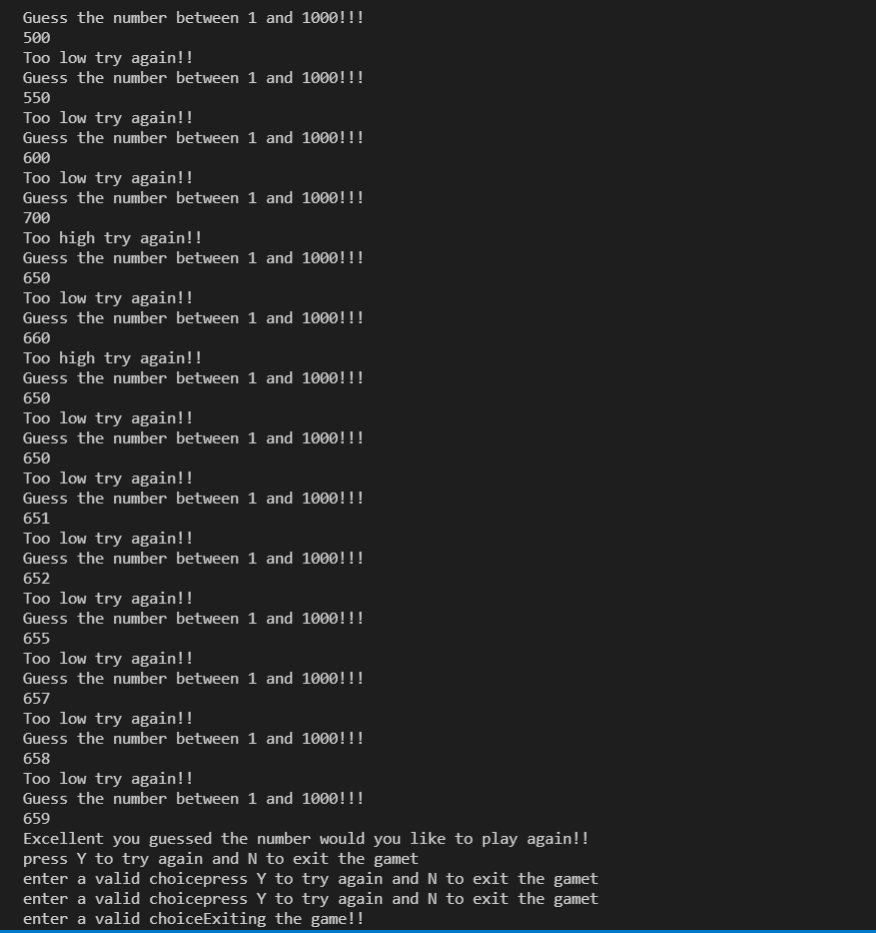
**4.)**

****

**5.)**

****

****

****