**A PROJECT REPORT**

**ON**

**NUMBER GUESSING GAME USING C++**

Submitted in partial fulfillment for the requirement of the award of

INTERNSHIP

IN

COMPETITIVE PROGRAMMING USING C++



Submitted By

Madhav Gaur (LNCT group of Colleges, Bhopal)

Under the guidance of

Mr. Akhilesh Sirohi

**PROJECT IDEA:**

It is a simple game that is developed using C++.

In this game, the computer generates a secret number in the range of 1 to 100, and the player has to guess it.

The game has three difficulty levels. A player’s chances of guessing are limited by the level they choose. The easy level gives the player 10 chances to guess the secret number, the medium level 7 chances, whereas the difficult level only offers 5 chances.

During the game, a player tells the computer his assumption about a number, and the computer tells if the player is correct. If his number is less or more than the secret number, the computer informs the player, and the player tries again.

The player can also end the game at any time.

**APPROACH:**

Step 1: Generate a Random Secret Number Between 1 & 100

No function in C++ generates a random function in a given range. Therefore, we will use the rand() function. We will need the cstdlib library to use rand(). The formula to generate a random function within a range is

random Number = 1 + (rand() %100)

Now, every time we run the program, the computer generates the same random number as the secret number, making it quite repetitive, and boring and this one-time game loses its essence.

To generate different random numbers each time the program runs, we will use the srand(time(0)) function. This function changes the seed each time the program runs. time(0) returns the number of seconds that the system clock shows. Since we will be using the time function, we will have to include the ctime library.

Step 2: Ask the User to Select the Level of Difficulty

While loops let us implement a menu-driven program in which the player can select the degree of difficulty. The user can press 1 to select the easy level, 2 for medium, and 3 for the difficulty level.

Step 3: Determine the Number of Chances the Player has Based on the Level of Difficulty

When the player selects the easy level, he gets 10 chances to guess the secret number. With medium, he has 7 chances while with hard, he has 5.

Step 4: Check Whether the Entered Number is Equal to the Secret Number

Using the if-else construct, we will check if the entered number matches the secret number.

As the player is granted 10 chances at the easy level, we will iterate from 1 to 10 to determine if the entered number matches the actual secret number. Since the player has only seven choices in level medium, we iterate from 1 to 7 to check if the number matches the secret number. We will iterate from 1 to 5 if the player selects hard level since there are only 5 choices.

We will display that the secret number is smaller than the chosen number if the entered number is smaller than the secret number.

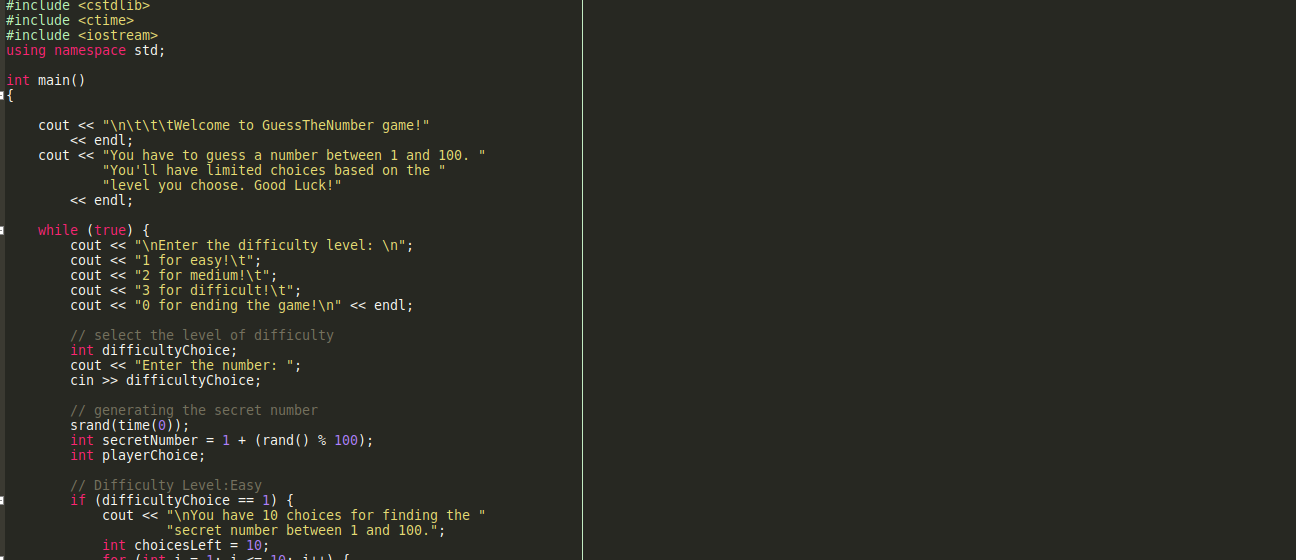
Whenever the entered number exceeds the secret number, we will display that the secret number is greater, which serves as a hint for the player.

We will also display the number of choices left.

Once the number entered matches the secret number, the player will be notified that he has won.

The game ends if the player cannot guess the number and he or she has no more choices left. This terminates the game.

# Code :



## Output :

