PROJECT REPORT

Name: Hardik Bansal

SAP ID: 590022662

Batch: 20

Course: Programming in C

Course code: CSEG1041

Project Title: Random Number Guessing – Two-Player Scoring Game

Submission Date: 25 November 2025

1. Abstract

- This project implements a simple two-player number-guessing game coded in C. A random number between 1 and 100 is generated using the rand() function for which I had used stdlib.h , and both players attempt to guess it .

- The player whose guess is nearer to the random number generated gets higher number of points.Points are awarded based on how close each player’s guess is to the generated number.

- The program makes use of arrays, conditional statements, loops, functions, and string handling. After three rounds, the score of both the players is calculated and the player with higher score is declared winner of the game . The project demonstrates basic programming concepts, user input handling, and logical decision-making in C.

2. Problem Definition

The objective is to design and implement a C program that:

- Generate a random number between 1 and 100.
- Asks the users to input their guess for three rounds
- Distributes points depending on how close is the number guessed by the player to the random number generated
- Stores points of each round for both players using arrays.
- Calculates total scores of both players by finding the total sum of all the elements of the array .
- Displays the name of the winner and also congratulates the winner
- Major concepts used in the project :
    - Functions
    - Loops
    - Conditional Statements
    - Arrays
    - Strings
    - Random number generation

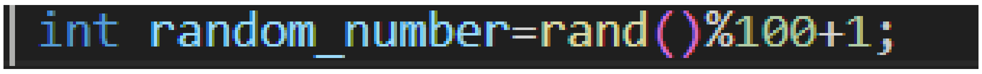
3. System Design (Algorithm)

Step 1: Start
Step 2: Generate random number using rand()%100 + 1
Step 3: Take player names as input using scanf
Step 4: Repeat for 3 rounds :

- Ask Player 1 for a number
- Ask Player 2 for a number
- Call pointcalculate() function for both inputs
- Store points in arrays named input1 for player1 and input2 for player2

Step 5: After all rounds reveal the random number
Step 6: Display round-wise score table
Step 7: Calculate total score of both players and store them in sum1 for player1 and sum2 for player2
Step 8: Compare totals and declare winner
Step 9: End

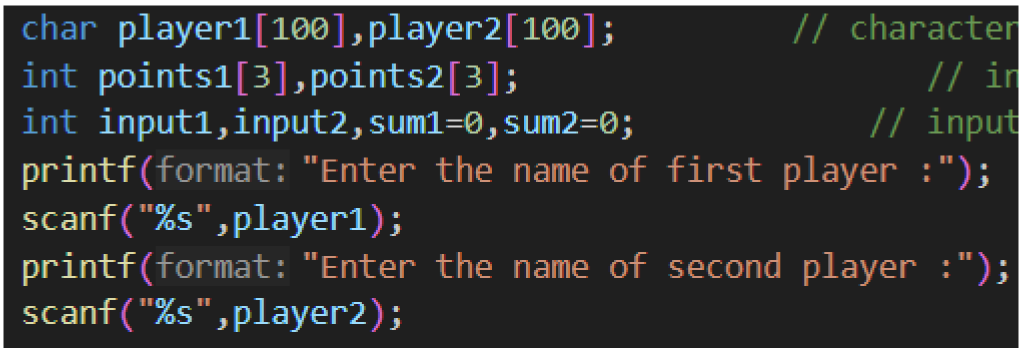
4. Implementation Details (With Snippets)



```
int random_number=rand()%100+1;
```

The game begins by generating a random number between 1 and 100 using the rand() function from the <stdlib.h> library. This number remains constant throughout all three rounds.

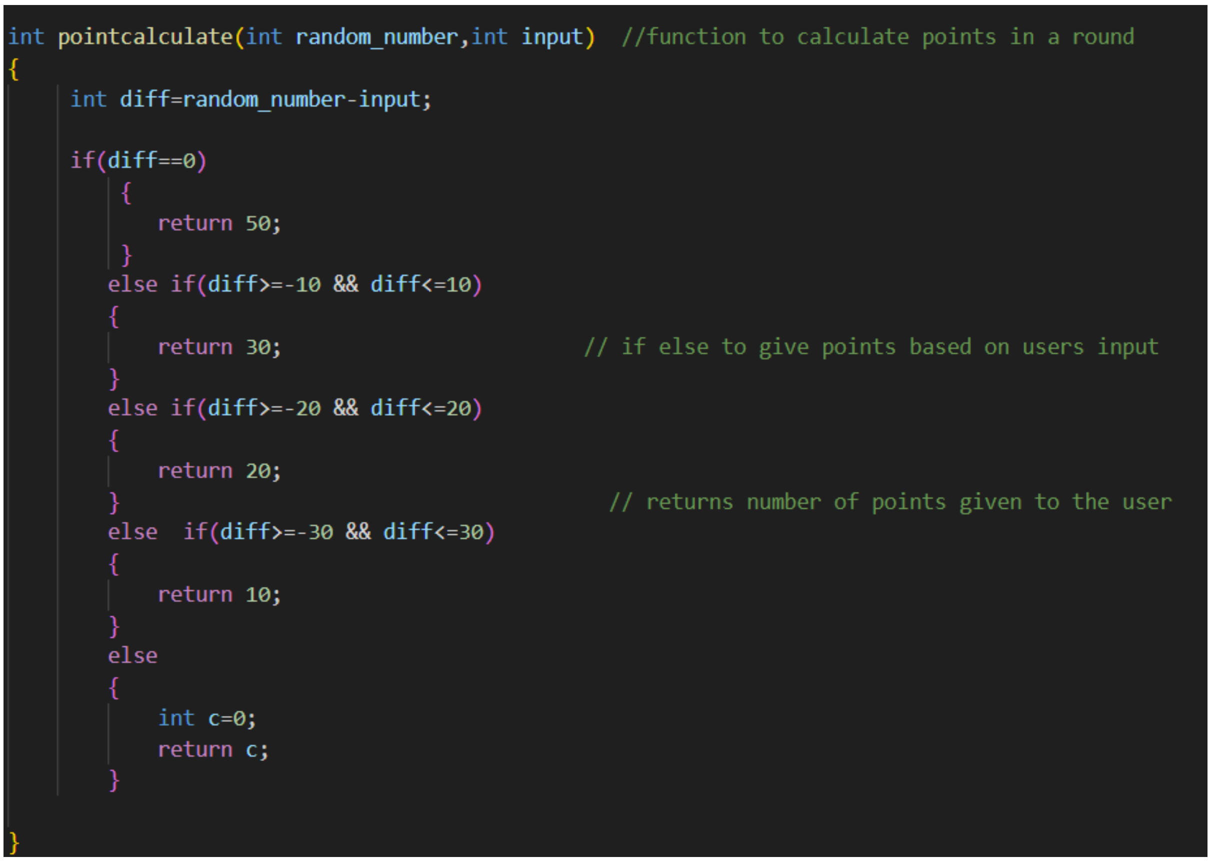
The modulo operation ensures the number stays within the desired range, and adding 1 avoids generating 0.



```
char player1[100],player2[100];        // character
int points1[3],points2[3];                    // in
int input1,input2,sum1=0,sum2=0;            // input
printf(format: "Enter the name of first player :");
scanf("%s",player1);
printf(format: "Enter the name of second player :");
scanf("%s",player2);
```

Player names are stored using character arrays (strings). The program prompts both players to enter their names

Using %s allows storing names without spaces.



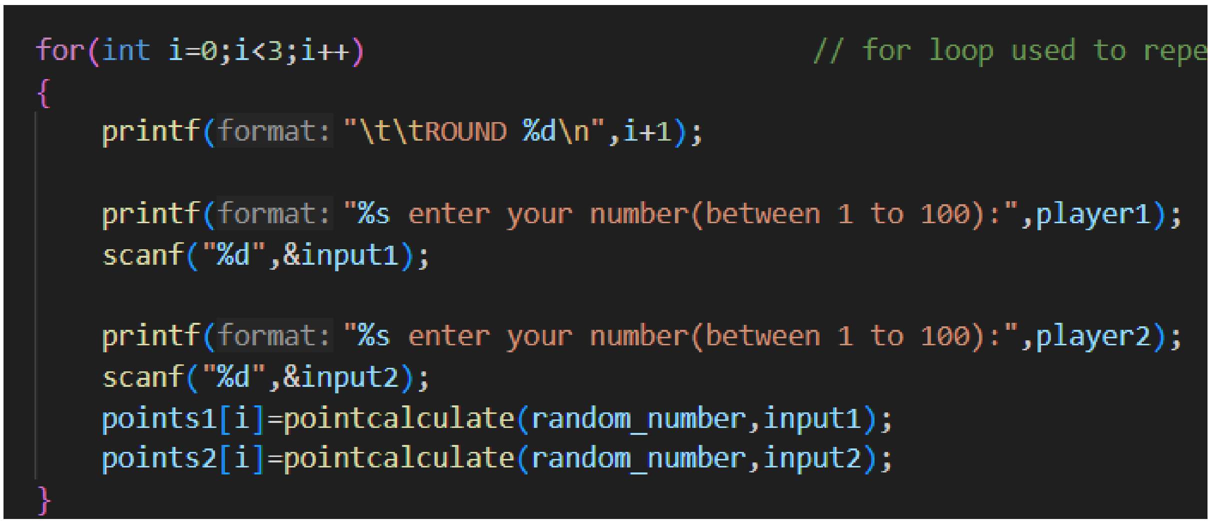
```
int pointcalculate(int random_number,int input)  //function to calculate points in a round
{
    int diff=random_number-input;

    if(diff==0)
       {
           return 50;
       }
      else if(diff>=-10 && diff<=10)
      {
           return 30;                    // if else to give points based on users input
      }
      else if(diff>=-20 && diff<=20)
      {
           return 20;
      }                                    // returns number of points given to the user
      else  if(diff>=-30 && diff<=30)
      {
           return 10;
      }
      else
      {
           int c=0;
           return c;
      }

}
```

A separate function pointcalculate() is implemented to determine how many points a player receives based on how close their guess is to the random number.

The function uses conditional statements to award points according to predefined difference ranges. A perfect guess earns the highest score (50 points).



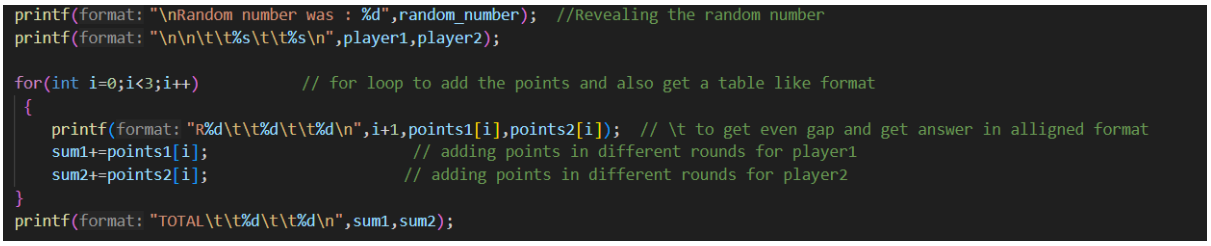
```c
for(int i=0;i<3;i++)                           // for loop used to repe
{
    printf(format: "\t\tROUND %d\n",i+1);

    printf(format: "%s enter your number(between 1 to 100):",player1);
    scanf("%d",&input1);

    printf(format: "%s enter your number(between 1 to 100):",player2);
    scanf("%d",&input2);
    points1[i]=pointcalculate(random_number,input1);
    points2[i]=pointcalculate(random_number,input2);
}
```

The game consists of **three rounds**, implemented using a for loop. In each round, both players input a number between 1 and 100.

Here, points1[] and points2[] store points for each of the three rounds.

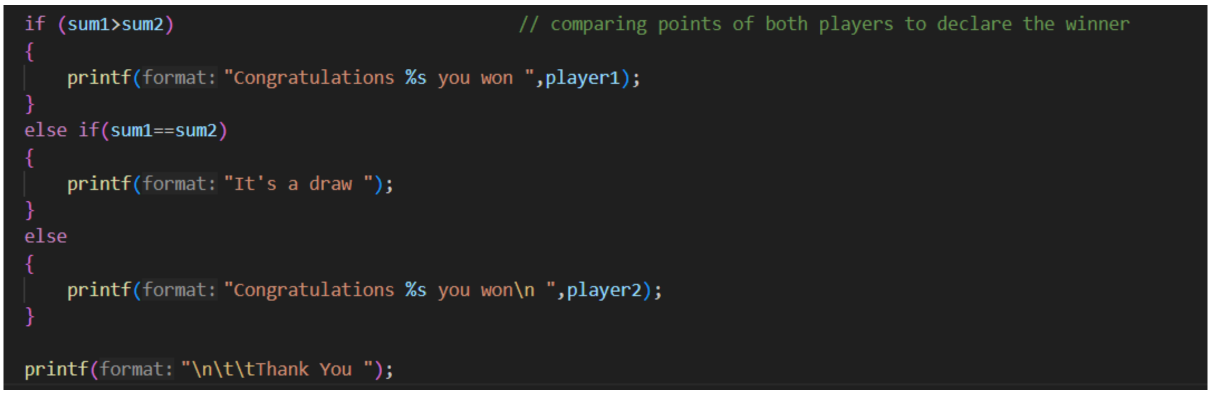


```c
printf(format: "\nRandom number was : %d",random_number);  //Revealing the random number
printf(format: "\n\n\t\t%s\t\t%s\n",player1,player2);

for(int i=0;i<3;i++)           // for loop to add the points and also get a table like format
 {
    printf(format: "R%d\t\t%d\t\t%d\n",i+1,points1[i],points2[i]);  // \t to get even gap and get answer in alligned format
    sum1+=points1[i];                          // adding points in different rounds for player1
    sum2+=points2[i];                         // adding points in different rounds for player2
}
printf(format: "TOTAL\t\t%d\t\t%d\n",sum1,sum2);
```

After all rounds, the random number is revealed, and points of each round are printed in a neat table format using \t for alignment.

Totals are calculated using cumulative addition within the same loop.



```c
if (sum1>sum2)                              // comparing points of both players to declare the winner
{
    printf(format: "Congratulations %s you won ",player1);
}
else if(sum1==sum2)
{
    printf(format: "It's a draw ");
}
else
{
    printf(format: "Congratulations %s you won\n ",player2);
}

printf(format: "\n\t\tThank You ");
```

Finally, total scores are compared to declare the winner or
conclude a draw.
A simple if-else ladder determines the result based on total points
accumulated.

5. Testing & Results

Round	Player1	Player2	random number	points p1	points p2
1	45	70	52	30	20
2	52	48	52	50	30
3	30	55	52	10	30

6. Conclusion & Future Work

Conclusion

The program successfully simulates a two player number-guessing game which is a totally entertaining game. It demonstrates how functions, loops, arrays, and conditions work together . The scoring system is very clear, and the output is cleanly formatted.

Future Work

- Add input validation to prevent numbers outside 1–100 being input
- Add difficulty modes
- Add an option for more rounds
- Add more number of players
- Convert the game into a graphical interface
- Use srand(time(NULL)) for better randomness in each round

7. References

1. *Let Us C* – Yashavant Kanetkar
2. TutorialsPoint: C Programming
3. GeeksForGeeks – C Language