Project Name:

"Missing Number Guessing Game"

Team member ID:

1. **242-35-119**
2. **242-35-232**

Project Overview

- **Objective**: The "Missing Number Guessing Game" is an interactive program where users guess the missing number from a sequence of even or odd numbers based on their age. The game is designed to be simple and fun, testing the user's ability to identify the missing number in a given series of numbers.
- **Features**:
  - The program checks if the user's age is odd or even.
  - Based on the age, the program generates an array of either even or odd numbers.
  - A number from the array is randomly removed (the missing number).
  - The user guess the missing number.
  - The program informs the user if they guessed the correct missing number or not.

Structure and Functionality

- **Input**:
  - User is input their age.

```
Enter your age: 30
```

- **Logic**:
  - The program checks if the age is even or odd.
  - If the age is even, the program generates a sequence of even numbers. If the age is odd, it generates a sequence of odd numbers.
  - The middle number in the array is removed (making it the missing number).
- **Output**:
  - The program outputs the generated sequence (with one missing number).
  - The user is asked to guess the missing number.
  - The program responds with a success or failure message based on the user's guess.

```
Your age is EVEN. You will get an EVEN number array.

Here is your array (one number is missing):
2 4 6 8 12 14 16 18 20

Enter the missing number: 10
Booyah! The missing number was 10.
```

Identifying Key Elements

1.Variables & Data Types:

Variable Name	Data Type
`age`	`int`
`ageMessage`	`char[100]`
`missingMessage`	`char[100]`
`start`	`int`
`numbers`	`int[SIZE]`
`missingNumber`	`int`
`missingIndex`	`int`
`userGuess`	`int`

2. Operators Used:

Operator	Type	Usage Example
`%`	Modulus	`age % 2 == 0`
`=`	Assignment	`start = 2;`
`==`	Comparison	`if (userGuess == missingNumber)`
`!=`	Comparison	`if (i != missingIndex)`
`+=`	Arithmetic	`num += 2`
`/`	Division	`SIZE / 2`

3. Conditional Statements:

- `if-else` is used to determine:
  - If the age is **even or odd**.
  - If the user's guess **matches the missing number**.

4. Loops:

Loop Type	Example
`for` loop	`: for (int i = 0, num = start; count < SIZE; i++, num += 2)`
`if` inside `for`	`: if (i == missingIndex) continue;`

5. Arrays and Strings:

- **Arrays (`numbers[SIZE]`)** are used to store the generated sequence.
- **Strings (`char ageMessage[100]`)** store user messages.

Explanation of the Code

Variable Declaration:

```
int age;
char ageMessage[100], missingMessage[100];
int start, numbers[SIZE], missingNumber, missingIndex;
```

User Input for Age:

```
printf("Enter your age: ");
scanf("%d", &age);
```

Determine Even or Odd Sequence:

```
if (age % 2 == 0) {
    strcpy(ageMessage, "\nYour age is EVEN. You will get an EVEN number array.\n");
    start = 2;
} else {
    strcpy(ageMessage, "\nYour age is ODD. You will get an ODD number array.\n");
    start = 1;
}
```

- **Conditional Statement (`if-else`)**:
  - If `age` is **even**, `start = 2`, and an **even number sequence** is generated.
  - If `age` is **odd**, `start = 1`, and an **odd number sequence** is generated.

Remove One Number from the Sequence:

```
missingIndex = SIZE / 2;  // Always remove the middle number
missingNumber = start + (missingIndex * 2);
```

- The program removes the **middle number** of the sequence.
- **Formula Explanation**:
  - The sequence follows **even (2, 4, 6, …)** or **odd (1, 3, 5, …)**.
  - The missing number is calculated using:

    $$\text{start} + (\text{missingIndex} \times 2)$$

  - This formula ensures that the removed number is the **middle number** in the sequence.

Generating the Array:

```
int count = 0;
for (int i = 0, num = start; count < SIZE; i++, num += 2) {
    if (i == missingIndex) {
        continue;
    }
    numbers[count++] = num;
}
```

- **For Loop (`for` loop)**:
  - Generates an array of **even or odd numbers**.
  - **Skips** the missing number (using `continue`).
  - Fills the remaining numbers in the array.

Display the Array:

```
strcpy(missingMessage, "\nHere is your array (one number is missing):\n");
printf("%s", missingMessage);
for (int i = 0; i < SIZE; i++) {
    printf("%d ", numbers[i]);
}
printf("\n");
```

User Guess the Missing Number:

```
int userGuess;
printf("\nEnter the missing number: ");
scanf("%d", &userGuess);
```

Check if User's Guess is Correct:

```
if (userGuess == missingNumber) {
    printf("Booyah! The missing number was %d. \n", missingNumber);
} else {
    printf("You are an ox. The missing number was %d.", missingNumber);
}
```

- **Conditional Statement (`if-else`)**:
  - If the guess is **correct**, the program prints `"Booyah! The missing number was X."`
  - If the guess is **wrong**, it prints `"You are an ox. The missing number was X."`

Write the main code:

```
#include <stdio.h>
#include <string.h>
#define SIZE 10
int main() {
    int age;
    char ageMessage[100], missingMessage[100];
    int start, numbers[SIZE], missingNumber, missingIndex;
    printf("Enter your age: ");
    scanf("%d", &age);
    if (age % 2 == 0) {
        strcpy(ageMessage, "\nYour age is EVEN. You will get an EVEN number array.\n");
        start = 2;
    }
else {
        strcpy(ageMessage, "\nYour age is ODD. You will get an ODD number array.\n");
        start = 1;
    }
    printf("%s", ageMessage);
    missingIndex = SIZE / 2;
    missingNumber = start + (missingIndex * 2);
    int count = 0;
    for (int i = 0, num = start; count < SIZE; i++, num += 2) {
```

```
        if (i == missingIndex) {
            continue;
        }
        numbers[count++] = num;
    }
    strcpy(missingMessage, "\nHere is your array (one number is missing):\n");
    printf("%s", missingMessage);
    for (int i = 0; i < SIZE; i++) {
        printf("%d ", numbers[i]);
    }
    printf("\n");
    int userGuess;
    printf("\nEnter the missing number: ");
    scanf("%d", &userGuess);
    if (userGuess == missingNumber) {
        printf("Booyah! The missing number was %d. \n", missingNumber);
    } else {
        printf("You are an ox. The missing number was %d.", missingNumber);
    }
 return 0;
}
```

This project is a simple yet engaging **number pattern recognition game** that tests the user's ability to identify missing numbers in a sequence. By incorporating **conditional statements, loops, arrays, and string operations**, it effectively demonstrates core C programming concepts in a practical way.