Allowed Time: 60 Minutes Total Marks: 30 Marks

Instructions:

1. Gossips are not allowed.
2. Teacher assistants are for your help, so be nice with them. Respect them as they are teaching you. Raise your hands if you have some problem and need help from TA. Avoid calling them by raising your voice and disturbing the environment of Lab.
3. TA may deduct your marks for any kind of ill-discipline or misconduct from your side.
4. Evaluation will be considered final and you cannot debate for the marks. So, focus on performing the tasks when the time is given to you.

Task 01: (10 Marks, 20 min)

Question Statement:

You are required to implement a **Number Guessing Game** where the player must guess a predefined secret number (e.g., 45). The game should provide feedback on whether the guess is **too high, too low, or correct** and count the number of attempts made.

Functions to be Implemented:

1. **`void check_guess(int guess, int secret);`**
   - This function takes the guessed number and the secret number as input.
   - It prints whether the guess is **too high, too low, or correct**.

Main Function:

The `main()` function should only call the `game()` function.
The `game()` function should:

- Initialize the secret number (e.g., `45`).
- Continuously ask the user to guess the number.
- Use `check_guess()` to provide feedback.
- Count the number of attempts.
- Display the total number of attempts when the user correctly guesses the number.

Sample Function Call in `main()`:

```
int main() {
    game();
    return 0;
}
```

Sample Output:

```
Enter your guess: 4
Too low! Try again.
Enter your guess: 65
Too high! Try again.
Enter your guess: 23
Too low! Try again.
Enter your guess: 76
Too high! Try again.
Enter your guess: 40
Too low! Try again.
Enter your guess: 45
Correct! You guessed it.
Congratulations! You guessed the number in 6 attempts.
```

Solution:

```c
#include <stdio.h>

// Function to check if the guess is correct
void checkGuess(int secretNumber, int guess) {
    if (guess < secretNumber)
        printf("Your guess is too low. Try again.\n");
    else if (guess > secretNumber)
        printf("Your guess is too high. Try again.\n");
    else
        printf("Congratulations! You guessed the correct number in %d attempts!\n", guess);
}

void game() {
    int secretNumber = 45; // Fixed secret number
    int guess, attempts = 0;


    do {
        printf("Enter your guess: ");
        scanf("%d", &guess);
        attempts++; // Count attempts
        if (guess != secretNumber) {
            checkGuess(secretNumber, guess);
        }
    } while (guess != secretNumber);

    printf("Congratulations! You guessed the correct number in %d attempts!\n", attempts);
}

int main() {
    game();
    return 0;
}
```

Task 02: (10 Marks, 15 min)

BMI (Body Mass Index) & Health Risk Calculator

Question Statement:

The **Body Mass Index (BMI)** is a measure used to assess a person's body weight relative to their height. It is calculated using the formula:

BMI=Weight (kg) / Height (m^2)

Based on the BMI value, a person falls into one of the following categories:

- **Underweight:** BMI < 18.5
- **Normal weight:** 18.5 ≤ BMI <= 24.9
- **Overweight:** 25.0 ≤ BMI <= 29.9
- **Obese:** BMI ≥ 30

Your Task:

Write a **C++ program** that performs the following:

1. **Ask the user** to enter their weight (in kg) and height (in meters).
2. **Implement a function** float calculateBMI(float weight, float height) to compute the BMI.

3. **Implement another function** string categorizeBMI(float bmi) to determine the health category based on the BMI value.
4. **Display** the calculated BMI value along with the corresponding health category.
5. **Use a loop** to allow multiple users to check their BMI until they choose to exit.

Example Output:

Enter weight (kg): 70
Enter height (m): 1.75
Your BMI: 22.86
Health Category: Normal weight
Do you want to check another BMI? (y/n): n
Thank you for using the BMI Calculator!

Solution:

```c
#include<stdio.h>

float calculateBMI(float w , float h){
    return w / (h * h);
}
void categorizeBMI( float bmi){
    if(bmi < 18.5){
        printf("Underweight!\n");
    }
    else if(bmi >= 18.5 && bmi <= 24.9){
        printf("Normal weight!\n");
    }
    else if(bmi >= 25.0 && bmi <= 29.9){
        printf("Over weight!\n");
    }
    else if(bmi >= 30){
        printf("Obese!\n");
    }
    else{
        printf("You have entered invalid wieght or height!");
    }
}
int main(){
    char choice='y';
    float w, h , BMI;

    do{
        printf("Enter your weight (kg): ");
        scanf("%f", &w);
        printf("Enter your height (m): ");
        scanf("%f", &h);

        if(h<=0){
            printf("Invalid height! Height must be greater than zero.\n");
            continue;
        }

        BMI = calculateBMI(w, h);
        printf("Your BMI: %.2f\n", BMI);
```

```
        categorizeBMI(BMI);

        printf("Do you want to calculate another BMI (y/n): ");
        scanf(" %c", &choice);

    } while (choice == 'y' || choice == 'Y');
    printf("Thankyou for using the BMI Calculator!");
}
```

Task 03: (10 Marks, 25 min)

Question Statement:

Write a C program to check whether a given number is an **Armstrong number** using a function.

Requirements:

1. Implement a function:
2. **int is_armstrong(int num);**
    - This function should take an integer as input and return 1 if the number is an Armstrong number and 0 otherwise.
    - An **Armstrong number** (also called a narcissistic number) is a number where the sum of its digits raised to the power of the number of digits equals the number itself.
    - Example:
        - **153** → 1^3 + 5^3 + 3^3 = 153 (Armstrong number)
        - **9474** → 9^4 + 4^4 + 7^4 + 4^4 = 9474 (Armstrong number)
        - **123** → 1^3 + 2^3 + 3^3 = 36 (Not an Armstrong number)
3. In the main() function:
    - Ask the user to input a number.
    - Call is_armstrong() and display whether the number is an Armstrong number or not.

Example Output:

Enter a number: 9474
9474 is an Armstrong number.
Enter a number: 123
123 is not an Armstrong number.

Solution:

```
#include <stdio.h>

int isArmstrong(int num) {
    int originalNum = num, sum = 0, count = 0, temp = num;

    // Count the number of digits
    while (temp > 0) {
        count++;
        temp /= 10;
    }

    temp = num;

    // Calculate the sum of digits raised to the power of count
    while (temp > 0) {
        int digit = temp % 10;
        int power = 1;

        // Compute digit^count without using pow()
        for (int i = 0; i < count; i++) {
```

```
            power *= digit;
        }

        sum += power;
        temp /= 10;
    }

    return (sum == originalNum);
}

int main() {
    int num;
    printf("Enter a number: ");
    scanf("%d", &num);

    if (isArmstrong(num))
        printf("%d is an Armstrong number.\n", num);
    else
        printf("%d is not an Armstrong number.\n", num);

    return 0;
}
```