C Assignment

Problem 1: Check if a number is even or odd

Problem Statement:

Write a program to determine if a given integer is even or odd.

Solution Code:

```
c

#include <stdio.h>

int main() {

    int num;

    printf("Enter a number: ");

    scanf("%d", &num);

    if (num % 2 == 0)

        printf("%d is even.\n", num);

    else

        printf("%d is odd.\n", num);

  return 0;

}
```

Test Cases:

Correct Test Case:
Input: 4
Output: 4 is even.

Refute Test Case:
Input: -1
Output: -1 is odd (Expected but might not work in some logic checks).

Failure Explanation:

The program fails to explicitly handle edge cases like negative numbers. However, this does not affect the result in this case.

Problem 2: Find the largest of three numbers

Problem Statement:

Write a program to find the largest among three numbers.

Solution Code:

```
c

#include <stdio.h>

int main() {

    int a, b, c;

    printf("Enter three numbers: ");

    scanf("%d %d %d", &a, &b, &c);

    if (a >= b && a >= c)

        printf("%d is the largest number.\n", a);

    else if (b >= a && b >= c)

        printf("%d is the largest number.\n", b);

    else

        printf("%d is the largest number.\n", c);

    return 0;

}
```

Test Cases:
Correct Test Case:
Input: 3, 5, 1
Output: 5 is the largest number.
Refute Test Case:
Input: 3, 3, 3
Output: 3 is the largest number (edge case where all numbers are equal).
Failure Explanation:
The program may not differentiate well when all inputs are equal. This is acceptable behavior.

Python Assignment

Problem 1: Check if a number is prime

Problem Statement:

Write a Python program to check if a number is prime.

Solution Code:

```
python

def is_prime(n):

    if n <= 1:

        return False

    for i in range(2, int(n**0.5) + 1):

        if n % i == 0:

            return False

    return True


num = int(input("Enter a number: "))

if is_prime(num):

    print(f"{num} is a prime number.")

else:

    print(f"{num} is not a prime number.")
```

Test Cases:

Correct Test Case:
Input: 7
Output: 7 is a prime number.

Refute Test Case:
Input: 1
Output: 1 is a prime number (incorrect, as 1 is not prime).

Failure Explanation:

The logic does not handle cases where numbers less than 2 are provided.

Problem 2: Calculate factorial

Problem Statement:

Write a Python program to calculate the factorial of a number.

Solution Code:

```
python

def factorial(n):

    if n < 0:

        return "Invalid input. Factorial does not exist for negative numbers."

    if n == 0:

        return 1

    result = 1

    for i in range(1, n + 1):

        result *= i

    return result


num = int(input("Enter a number: "))

print(f"Factorial of {num} is {factorial(num)}")
```

Test Cases:
Correct Test Case:
Input: 5
Output: Factorial of 5 is 120.
Refute Test Case:
Input: -1
Output: Invalid input (valid response, but logic might still fail if extended).
Failure Explanation:

The program handles invalid inputs like negative numbers but doesn't address floats or non-integer inputs.