Question 1 [10 Marks]

A program that finds all Pythagorean triples (a, b, c) such that a^2 + b^2 = c^2 and all numbers are less than or equal to n. If a, b, or c is divisible by 5, print "Divisible by 5".

```
Public void find_pythagorean_triples( int n)
{
   for (int a = 1 ; a < n+1 ; a++)
  {
     for ( int b = a ; b < n+1 ; b++)
     {
         for( int c = b ; c < n+1 ; c++)
         {
                 if (  ((a * a) + (b * b)) == (c * c))
                 {
                     if ( a % 5 == 0 || b % 5 == 0 || c % 5 == 0)
                     {
                         System.out.println(" a "+a + " b "+ b+ " c " + c + " is divisible by 5 ");
                     }
                     else
                     {
                         System.out.println( "Pythagorean Triple: a "+ a + "b "+b + "c "+c);
                     }
                 }
         }
     }
  }
}
```

1. **[CO4] Illustrate the Control Flow Graph (CFG) for the find_pythagorean_triples() method, and remember to label the relevant node numbers on your question paper. [4]**

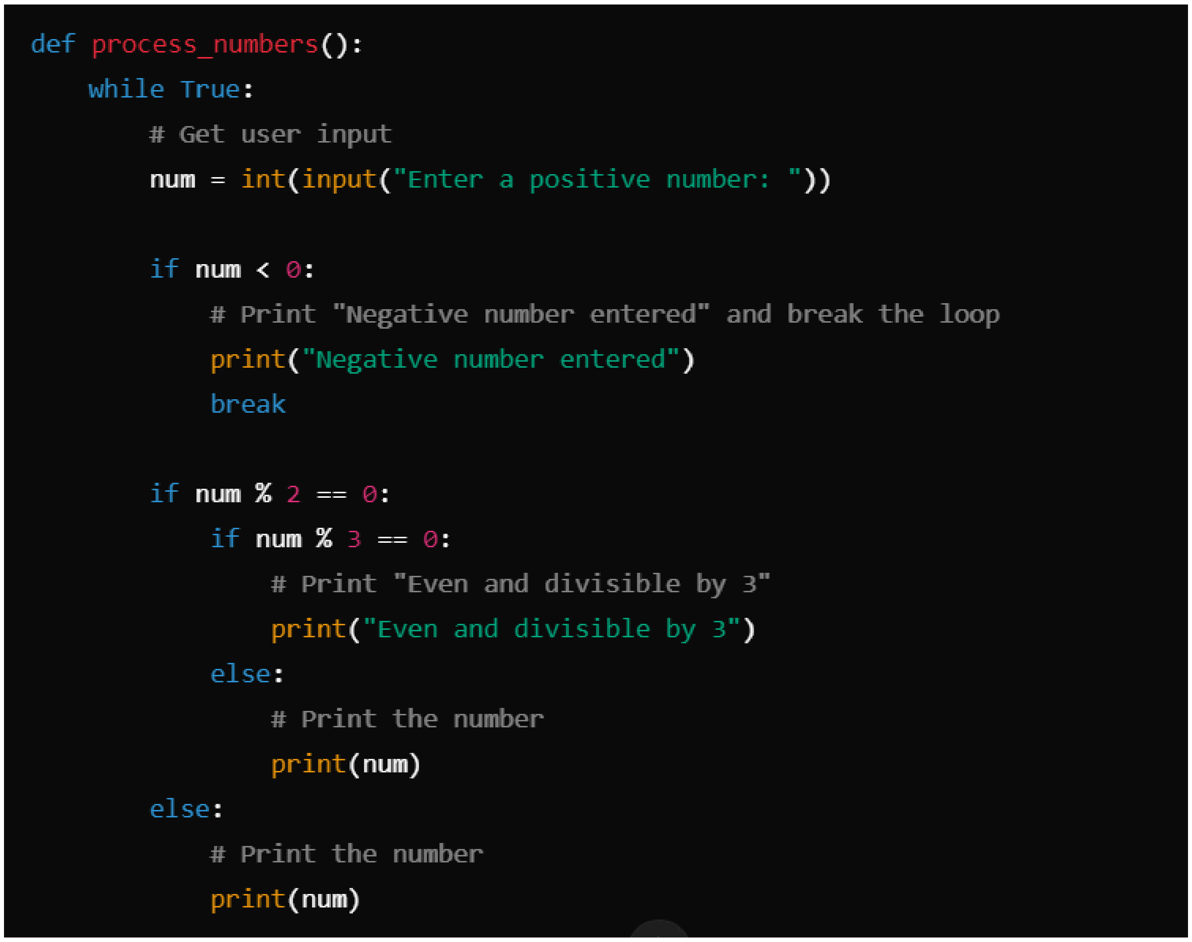
2. **[CO4] Calculate the cyclomatic complexity of the control flow graph drawn for the find_pythagorean_triples() method. [3]**

3. **[CO4] Identify all the independent paths from the CFG and show one test case of any independent path you identified. Assume any data you need to make that test case. [2]**

4. **[CO4] How can we understand if our path-based testing is done correctly? [1]**

Question 2 [5 Marks]

The program continues to prompt the user to enter a positive integer. If the user enters a negative number, the program prints "Negative number entered" and breaks the loop. If the number is even, it checks if the number is divisible by 3. If both conditions are met, print "Even and divisible by 3", otherwise print the number.



```python
def process_numbers():
    while True:
        # Get user input
        num = int(input("Enter a positive number: "))

        if num < 0:
            # Print "Negative number entered" and break the loop
            print("Negative number entered")
            break

        if num % 2 == 0:
            if num % 3 == 0:
                # Print "Even and divisible by 3"
                print("Even and divisible by 3")
            else:
                # Print the number
                print(num)
        else:
            # Print the number
            print(num)
```

1. **[CO4] Illustrate the Control Flow Graph (CFG) for the process_numbers() method, and remember to label the relevant node numbers on your question paper. [1.5]**

2. **[CO4] Calculate the cyclomatic complexity of the control flow graph drawn for the process_numbers() method. [1.5]**

3. **[CO4] Identify all the independent paths from the CFG and show one test case of any independent path you identified. Assume any data you need to make that test case. [1]**

4. **[CO4] How can we understand if our path-based testing is done correctly? [1]**

Question 3 [10 Marks] [CO4]

1. Calculate Cyclomatic Complexity for the given codes and compare the complexity between code 1 and code 2, which one has better cyclomatic complexity? (Lower is better).
2. Prepare the test cases for each independent path for each code.

Find the sum of all even numbers in an array.
Code 1:

```
int sum = 0; ….1
for (int i = 0; i < A.length; i++)
{
        if (A[i] % 2 == 0) {
                sum += A[i];
        }
        else
        {
            if (A[i] % 2 != 0)
            {
                  System.out.println(A[i] + " is odd, skipping...");
            }
       }
}
System.out.println(sum);
```

Find the sum of all even numbers in an array.
Code 2:

```
int sum = 0;
for (int i = 0; i < A.length; i++)
 {
    if (A[i] % 2 == 0)
    {
          sum += A[i];
    }
}
System.out.println(sum);
```

Question 4 [5 Marks] [CO4]

Calculate SIX for class bird from the code snippet below (Specialization Index for the code snippet below)

```
public class FlyingThing {

    public void fly( )
    {
        System.out.println(“ let’s fly in the sky ”);
    }

    public void dance( )
    {
        System.out.println(“ let’s dance and fly ”);
    }
}

public class Animal {

     public void eat( )
    {
        System.out.println(“ Hungry let’s eat ”);
    }

     public void sound( )
     {
         System.out.println(“ Chirping ”);
     }
}

public class Bird extends FlyingThing {

     public void sound( )  { System.out.println(“ Chirping ”);  }

     public void fly( ) { System.out.println(“ let’s fly in the sky  ”); }

     public void hunt( ) { System.out.println(“ let’s hunt some food!  ”); }
}
```