**14. Perform white box testing – Cyclomatic complexity, data flow testing, control flow testing for the given problem statement: Write algorithm for Fibonacci series**

**Algorithm for Fibonacci Series**

Here’s a simple iterative algorithm to generate the Fibonacci series:

java

public class Fibonacci {

public static void main(String[] args) {

int n = 10; // Number of terms in the Fibonacci series

int[] fibSeries = new int[n];

fibSeries[0] = 0;

fibSeries[1] = 1;

for (int i = 2; i < n; i++) {

fibSeries[i] = fibSeries[i-1] + fibSeries[i-2];

}

for (int i = 0; i < n; i++) {

System.out.print(fibSeries[i] + " ");

}

}

}

**1. Cyclomatic Complexity**

Cyclomatic complexity is measured using the formula: $$ \text{CC} = E - N + 2P $$

**Control Flow Graph (CFG):**

* **Nodes (N)**:
  + Start
  + fibSeries[0] = 0
  + fibSeries[1] = 1
  + for loop initial condition (i = 2)
  + for loop body (fibSeries[i] = fibSeries[i-1] + fibSeries[i-2])
  + for loop increment condition (i < n)
  + second for loop initial condition (i = 0)
  + second for loop body (System.out.print(fibSeries[i] + " "))
  + second for loop increment condition (i < n)
  + End
* **Edges (E)**: 11

**Cyclomatic Complexity (CC)**: $$ \text{CC} = E - N + 2P = 11 - 10 + 2 \times 1 = 3 $$

So, the Cyclomatic Complexity for this algorithm is 3, indicating there are 3 independent paths through the code.

**2. Data Flow Testing**

Data Flow Testing focuses on the points where variables are defined (D) and used (U).

**Variables:**

* n: Defined at the start and used in the for loops.
* fibSeries: Defined as an array and used to store Fibonacci numbers.
* i: Loop counter.

**Key Points:**

* **Definitions (D)**: n, fibSeries, i
* **Uses (U)**:
  + n is used in the for loop conditions.
  + fibSeries is used to assign and retrieve values within the loops.
  + i is defined at the start of loops and used in loop conditions and body.

**3. Control Flow Testing**

Control Flow Testing involves creating a control flow graph and testing all possible paths.

**Control Flow Graph:**

1. **Start**: Enter function
2. **Node 1**: Set fibSeries[0] = 0
3. **Node 2**: Set fibSeries[1] = 1
4. **Node 3**: Enter first for loop (i = 2)
5. **Node 4**: Check i < n
   * **True**: Go to Node 5
   * **False**: Go to Node 6
6. **Node 5**: Set fibSeries[i] = fibSeries[i-1] + fibSeries[i-2]
7. **Node 6**: Go to second for loop (i = 0)
8. **Node 7**: Check i < n
   * **True**: Go to Node 8
   * **False**: End
9. **Node 8**: Print fibSeries[i]
10. **Node 9**: Increment i and go back to Node 7

**Test Cases**

**Path 1: Small Series (n = 2)**

* Input: n = 2
* Expected Output: 0 1

**Path 2: Typical Case (n = 5)**

* Input: n = 5
* Expected Output: 0 1 1 2 3

**Path 3: Larger Series (n = 10)**

* Input: n = 10
* Expected Output: 0 1 1 2 3 5 8 13 21 34