Quiz# 6

Q.1) Consider the following method that takes two 2D int arrays as input and returns a 2D array containing the sum of corresponding elements of the input arrays, with the condition that the sum is even.

```
public static int[][] sumEven(int[][] arr1, int[][] arr2) {
    int rows = arr1.length;
    int cols = arr1[0].length;
    int i = 0, j = 0;

    if (rows != arr2.length || cols != arr2[0].length) {
        throw new IllegalArgumentException("Input arrays must have the same dimensions");
    }

    int[][] result = new int[rows][cols];

    while (i < rows) {
        j = 0;
        while (j < cols) {
            int sum = arr1[i][j] + arr2[i][j];
            if (sum % 2 == 0) {
                result[i][j] = sum;
            } else {
                result[i][j] = 0;
            }
            j++;
        }
        i++;
    }

    return result;
}
```

Commented [AR1]: 1

Commented [AR2]: 2

Commented [AR3]: 3

Commented [AR4]: 4

Commented [AR5]: 5

Commented [AR6]: 6

Commented [AR7]: 7

Commented [AR8]: 8

Commented [AR9]: 9

Commented [AR10]: 10

Commented [AR11]: 11

Commented [AR12]: 12

Commented [AR13]: 13

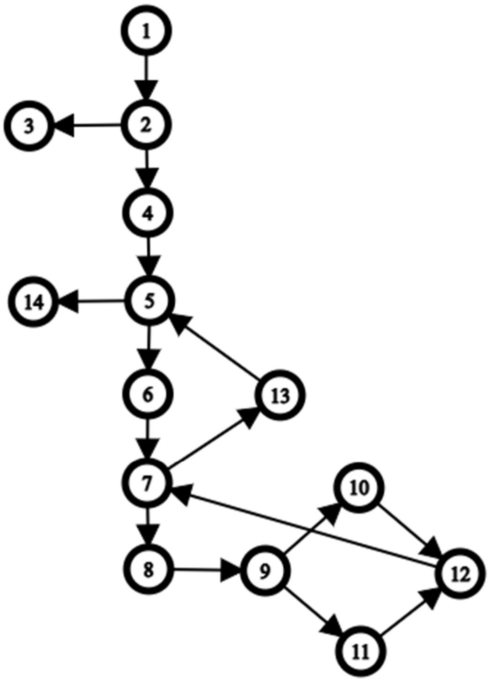
Commented [AR14]: 14

a. Fill out the following table for Edge Coverage criterion. **[10 marks]**

<table>
<tr><th colspan="2">TR (Test Requirements)</th></tr>
<tr><td colspan="2">{(1, 2), (2, 3), (2, 4), (4, 5), (5, 6), (5, 14), (6, 7), (7, 8), (7, 13), (8, 9), (9, 10), (9, 11), (11, 12), (10, 12), (12, 7), (13, 5)}</td></tr>
<tr><th>Test Paths that together fulfill test requirements</th><th>Input corresponding to the test path</th></tr>
<tr><td><b>1.</b> [1, 2, 3]</td><td>({ { 2, 2 } }, { {4}, {5} })</td></tr>
</table>

2. [1, 2, 4, 5, 6, 7, 8, 9, 10, 12, 7, 13, 5, 14]	({2}, {4})
3. [1, 2, 4, 5, 6, 7, 8, 9, 11, 12, 7, 13, 5, 14]	({3}, {6})

b. Draw CFG of the code. Annotate each node (and edge, where applicable) with the code statements. **[15 marks]**



Quiz# 6

Q.1) Consider the following method that takes two 2D int arrays as input and returns a 2D array containing the product of corresponding elements from two 2D arrays and stores it in the result array only if the product is greater than a threshold value.

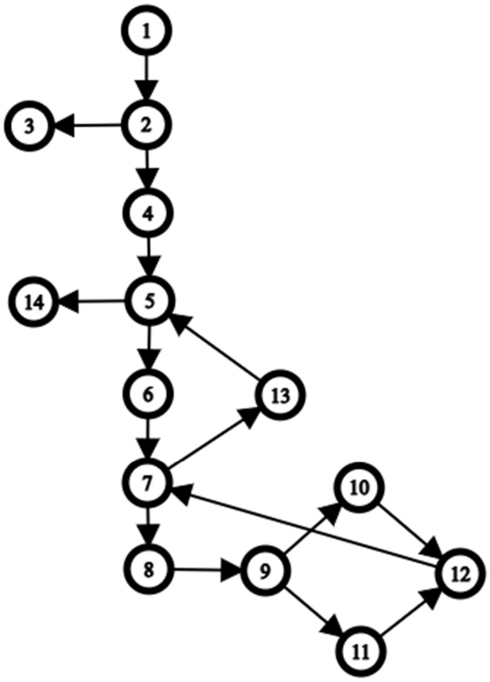
```
public static int[][] productAboveThreshold(int[][] arr1, int[][] arr2, int threshold) {
    int rows = arr1.length;
    int cols = arr1[0].length;
    int i = 0, j = 0;

    if (rows != arr2.length || cols != arr2[0].length) {
        throw new IllegalArgumentException("Input arrays must have the same dimensions");
    }

    int[][] result = new int[rows][cols];

    while (i < rows) {
        j = 0;
        while (j < cols) {
            int product = arr1[i][j] * arr2[i][j];
            if (product > threshold) {
                result[i][j] = product;
            } else {
                result[i][j] = 0;
            }
            j++;
        }
        i++;
    }

    return result;
}
```

a. Fill out the following table for Edge Coverage criterion. **[10 marks]**

<table>
<tr><th colspan="2">TR (Test Requirements)</th></tr>
<tr><td colspan="2">{(1, 2), (2, 3), (2, 4), (4, 5), (5, 6), (5, 14), (6, 7), (7, 8), (7, 13), (8, 9), (9, 10), (9, 11), (11, 12), (10, 12), (12, 7), (13, 5)}</td></tr>
<tr><th>Test Paths that together fulfill test requirements</th><th>Input corresponding to the test path</th></tr>
<tr><td>1. [1, 2, 3]</td><td>({ { 2, 2 } }, { {4}, {5} }, 55)</td></tr>
</table>

2. [1, 2, 4, 5, 6, 7, 8, 9, 10, 12, 7, 13, 5, 14]	({ { 9 } }, { 10 }, 55)
3. [1, 2, 4, 5, 6, 7, 8, 9, 11, 12, 7, 13, 5, 14]	({ { 9 } }, { 6 }, 55)

b. Draw CFG of the code. Annotate each node (and edge, where applicable) with the code statements. **[15 marks]**


1
2
3
4
5
14
6
13
7
10
8
9
12
11