**Software Testing**

**LAB MID**

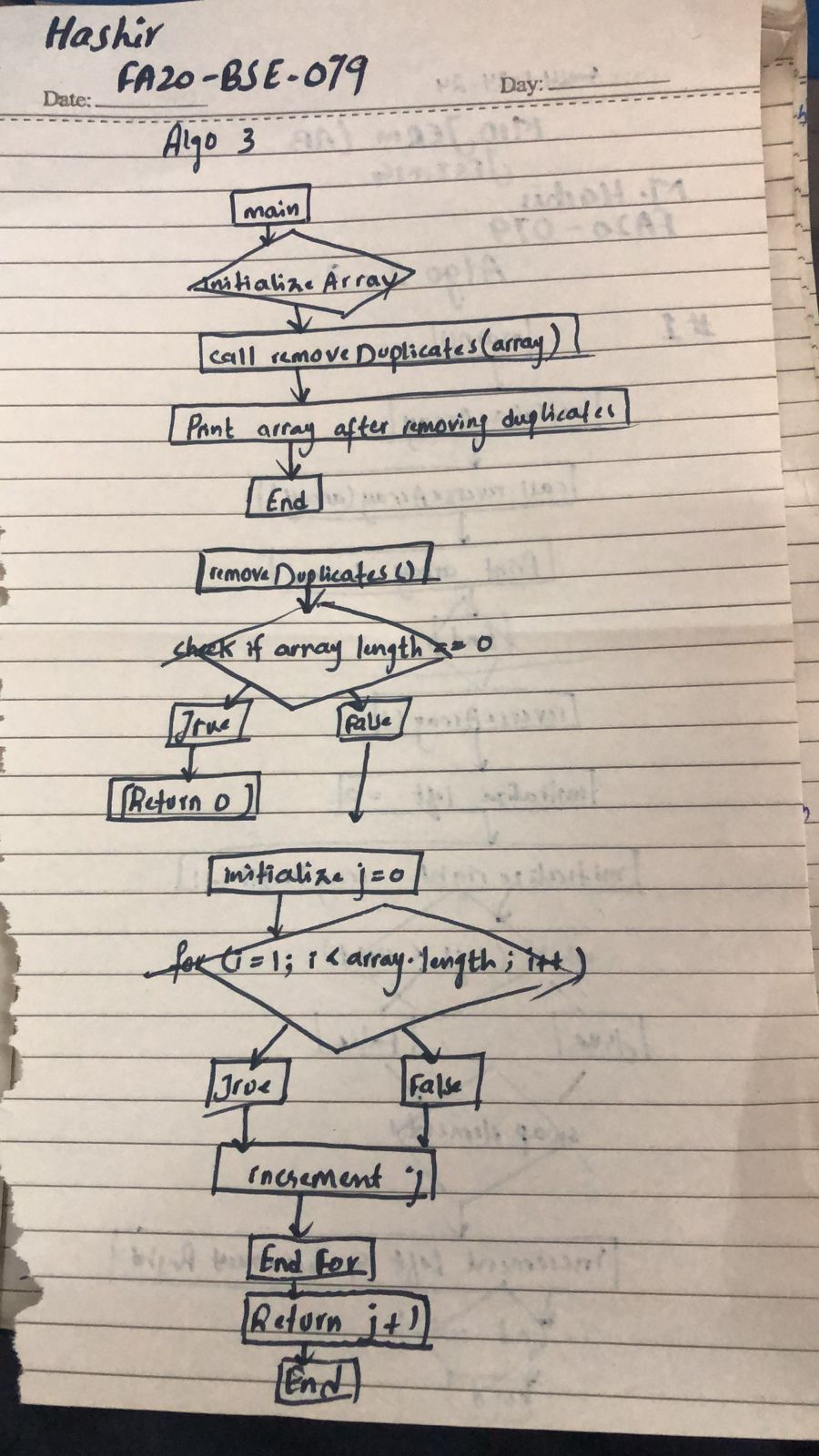
**Name:** Muhammad Hashir

**REG NO**: FA2-BSE-079

**ALGORITHM 1 & 3**

**ALGO # 3**

**Control Flow Graph**



**Test Cases**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | **Description** | **Input** | **Output** | **Status(Pass/fail)** |
| Tc\_01 | Remove duplicates from an empty array | {} | {} | Pass |
| Tc\_02 | Remove duplication from a single element array | {1} | {1} | Pass |
| Tc\_03 | Remove duplication from an array with all duplicates | {1,2,2,3} | {1,2,2,3} | Pass |
| Tc\_04 | Remove duplicates from an array with some duplicates | {1,2,2,3,4,4,5} | {1,2,2,3,4,4,5} | Pass |

**Junit Test:**

qimport org.junit.jupiter.api.Test;

import static org.junit.jupiter.api.Assertions.assertArrayEquals;

public class RemoveDuplicatesTest {

@Test

public void testRemoveDuplicatesEmptyArray() {

int[] input = {};

int expectedLength = 0;

int newLength = RemoveDuplicates.removeDuplicates(input);

assertArrayEquals(input, Arrays.copyOf(input, newLength));

}

@Test

public void testRemoveDuplicatesSingleElementArray() {

int[] input = {1};

int expectedLength = 1;

int newLength = RemoveDuplicates.removeDuplicates(input);

assertArrayEquals(input, Arrays.copyOf(input, newLength));

}

@Test

public void testRemoveDuplicatesAllDuplicates() {

int[] input = {1, 1, 1, 1};

int expectedLength = 1;

int[] expectedArray = {1};

int newLength = RemoveDuplicates.removeDuplicates(input);

assertArrayEquals(expectedArray, Arrays.copyOf(input, newLength));

}

@Test

public void testRemoveDuplicatesSomeDuplicates() {

int[] input = {1, 2, 2, 3, 4, 4, 5};

int expectedLength = 5;

int[] expectedArray = {1, 2, 3, 4, 5};

int newLength = RemoveDuplicates.removeDuplicates(input);

assertArrayEquals(expectedArray, Arrays.copyOf(input, newLength));

}

}

ALGO#1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test Case ID | Description | Inputs | Expected Result | Actual Result (Pass/Fail) |
| TC-001 | Reverse an array with odd length | {1, 2, 3, 4, 5} | {5, 4, 3, 2, 1} | Pass |
| TC-002 | Reverse an array with even length | {1, 2, 3, 4, 5, 6} | {6, 5, 4, 3, 2, 1} | Pass |
| TC-003 | Reverse an array with a single element | {1} | {1} | Pass |
| TC-004 | Reverse an empty array | {} | {} | Pass |
| TC-005 | Reverse an array with negative numbers | {-1, -2, -3, -4, -5} | {-5, -4, -3, -2, -1} | Pass |
| TC-006 | Reverse an array with mixed positive and negative numbers | {-1, 2, -3, 4, -5} | {-5, 4, -3, 2, -1} | Pass |
| TC-007 | Reverse an array with repeated elements | {1, 2, 2, 1} | {1, 2, 2, 1} | Pass |

**Junit Test:**

import org.junit.jupiter.api.Test;

import static org.junit.jupiter.api.Assertions.assertArrayEquals;

public class ReverseArrayTest {

@Test

public void testReverseOddLengthArray() {

int[] input = {1, 2, 3, 4, 5};

int[] expected = {5, 4, 3, 2, 1};

ReverseArray.reverseArray(input);

assertArrayEquals(expected, input);

}

@Test

public void testReverseEvenLengthArray() {

int[] input = {1, 2, 3, 4, 5, 6};

int[] expected = {6, 5, 4, 3, 2, 1};

ReverseArray.reverseArray(input);

assertArrayEquals(expected, input);

}

@Test

public void testReverseSingleElementArray() {

int[] input = {1};

int[] expected = {1};

ReverseArray.reverseArray(input);

assertArrayEquals(expected, input);

}

@Test

public void testReverseEmptyArray() {

int[] input = {};

int[] expected = {};

ReverseArray.reverseArray(input);

assertArrayEquals(expected, input);

}

@Test

public void testReverseNegativeNumbersArray() {

int[] input = {-1, -2, -3, -4, -5};

int[] expected = {-5, -4, -3, -2, -1};

ReverseArray.reverseArray(input);

assertArrayEquals(expected, input);

}

@Test

public void testReverseMixedNumbersArray() {

int[] input = {-1, 2, -3, 4, -5};

int[] expected = {-5, 4, -3, 2, -1};

ReverseArray.reverseArray(input);

assertArrayEquals(expected, input);

}

@Test

public void testReverseRepeatedElementsArray() {

int[] input = {1, 2, 2, 1};

int[] expected = {1, 2, 2, 1};

ReverseArray.reverseArray(input);

assertArrayEquals(expected, input);

}

}