**Muhammad Irfan**

**Reg No. FA21-BSE-027**

**Algo no 2 Palindrome**

## Code

**PalindromeCheck Class**

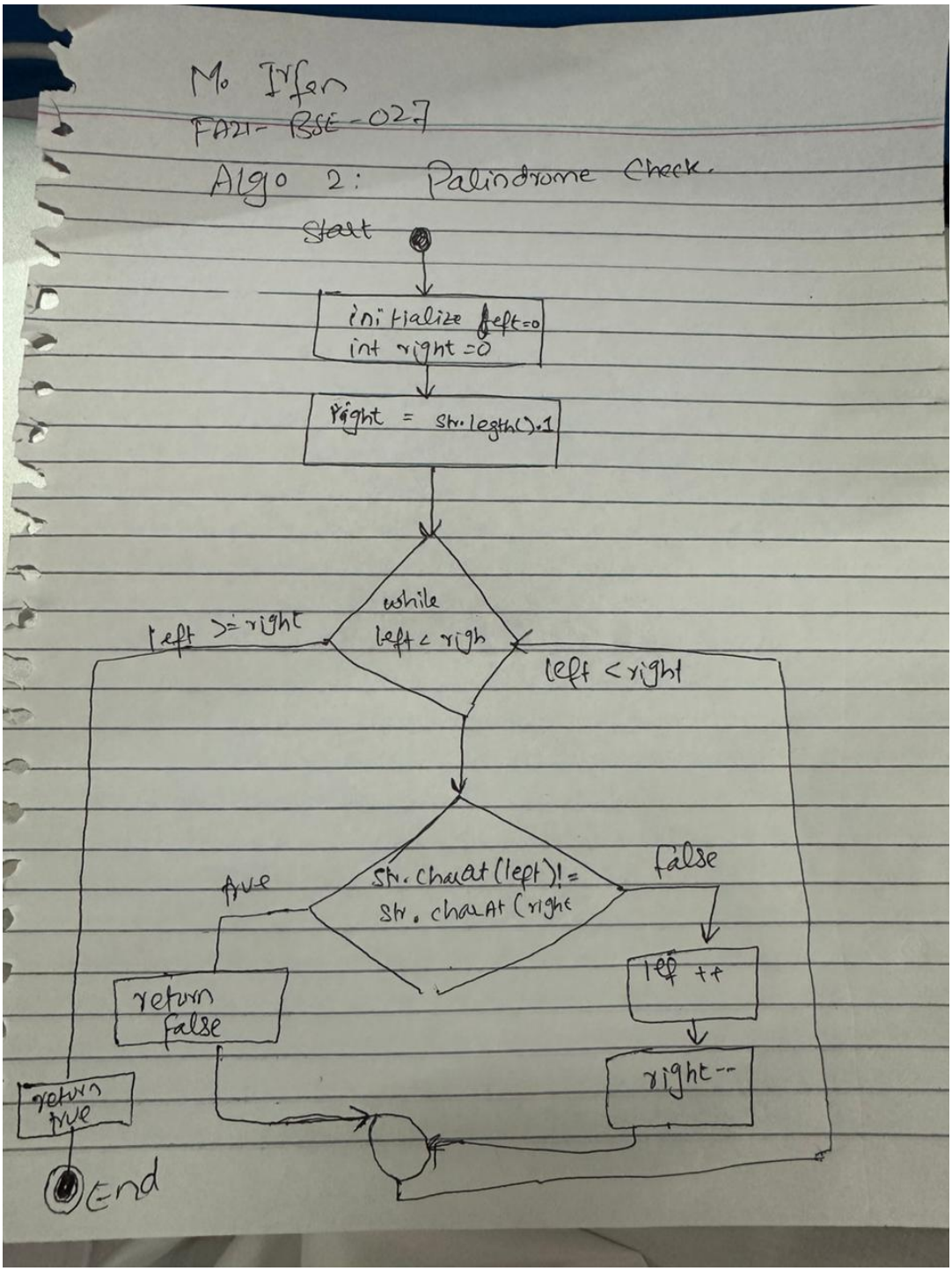
```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 */

package com.mycompany.palindromcheck;

/**
 *
 * @author irfan
 */
public class PalindromeCheck {
   public static boolean isPalindrome(String str) {
      int left = 0, right = str.length() - 1;
      while (left < right) {
         if (str.charAt(left) != str.charAt(right)) {
            return false;
         }
         left++;
         right--;
      }
      return true;
   }

   public static void main(String[] args) {
      String str = "madam";
      boolean isPalindrome = isPalindrome(str);
      System.out.println("Is the string a palindrome? " + isPalindrome);
   }
}
```

**PalindromeCheck Test Class**

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/UnitTests/JUnit5TestClass.java to edit this template
 */

import com.mycompany.palindromcheck.PalindromeCheck;
import org.junit.jupiter.api.AfterEach;
import org.junit.jupiter.api.AfterAll;
import org.junit.jupiter.api.BeforeEach;
import org.junit.jupiter.api.BeforeAll;
import org.junit.jupiter.api.Test;
import static org.junit.jupiter.api.Assertions.*;

/**
 *
 * @author irfan
 */
public class PalindeomJUnitTest {

    public PalindeomJUnitTest() {
    }

    @BeforeAll
    public static void setUpClass() {
    }

    @AfterAll
    public static void tearDownClass() {
    }

    @BeforeEach
    public void setUp() {
    }

    @AfterEach
    public void tearDown() {
    }



    @Test
    public void testPalindrome() {
```

```
//        assertTrue(PalindromeCheck.isPalindrome("madam"));
//       assertFalse(PalindromeCheck.isPalindrome(""));
         assertTrue(PalindromeCheck.isPalindrome("12321"));
        assertFalse(PalindromeCheck.isPalindrome("12345"));
    }

}
```

## CFG


M. Irfan
FA21-BSE-027
Algo 2: Palindrome Check.
start
initialize left=0
int right=0
right = str.length()-1
while
left < righ
left >= right
left < right
str.charAt(left)!=
str.charAt(right
true
false
return false
left ++
right--
return true
end

## Test Cases:

1.

| | |
|---|---|
| Test Case ID | TC01 |
| Test Case Description | Check if the string "madam" is a palindrome. |
| Precondition | Method shall be defined |
| Test Step | Call isPalindrome("madam") |
| Test Data | "madam" |
| Expected Result | True |
| Actual Result | True |
| Status | Pass |

2.

| | |
|---|---|
| Test Case ID | TC02 |
| Test Case Description | Check if the empty string "" is a palindrome |
| Precondition | Method shall be defined |
| Test Step | Call isPalindrome("") |
| Test Data | "” |
| Expected Result | Pass |
| Actual Result | False |
| Status | Fail |

3.

| Test Case ID | TC03 |
|---|---|
| Test Case Description | Check if the numeric string "12345" is a palindrome. |
| Precondition | Method shall be defined |
| Test Step | Call isPalindrome("12345") |
| Test Data | "12345" |
| Expected Result | false |
| Actual Result | False |
| Status | Pass |

4.

| Test Case ID | TC04 |
|---|---|
| Test Case Description | Check if the numeric string "12321" is a palindrome. |
| Precondition | Method shall be defined |
| Test Step | Call isPalindrome("123") |
| Test Data | "123" |
| Expected Result | True |
| Actual Result | True |
| Status | Pass |