

Unit-Testing using JUnit

Reported by:

Team	:	13
		<ol style="list-style-type: none">1. 11321003/Gabriel Sigalingging2. 11321023/Emy Sonia Sinambela3. 11321032/Bennedict Tambunan4. 11321043/Suandika Napitupulu

1. Testing for Palindrome_1.java

```
package Modified;

import java.io.BufferedReader;
import java.io.InputStreamReader;

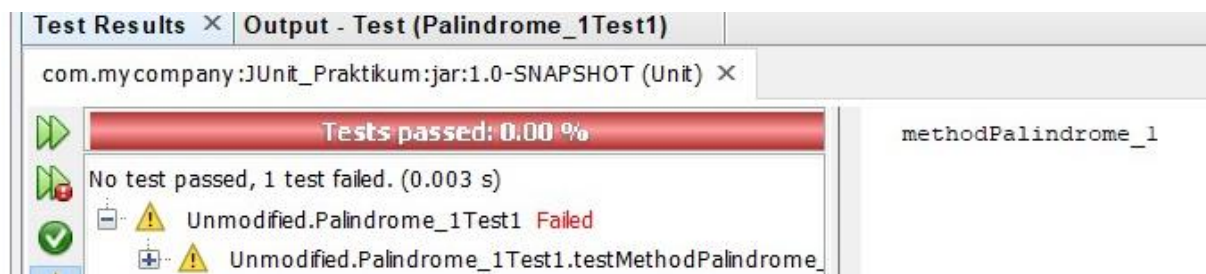
public class Palindrome_1 {
    public String methodPalindrome_1(int n1){
        String hasil;
        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
        int r, n2;    int rev=0;    n2=n1;    while(n1>0){        r = n1%10;
        rev = rev*10+r;
        n1 = n1*10;
    }
    if(rev==n2){
        hasil = "palindrome number!";
    } else{
        hasil = "NOT palindrome number!";
    }
    return hasil;
    }
}
```

a. Input: 1

Snippet of test case

```
@Test
public void testMethodPalindrome_1() {
    System.out.println("methodPalindrome_1");
    int n1 = 1;
    Palindrome_1 instance = new Palindrome_1();
    String expectedResult = "palindrome number!";
    String result = instance.methodPalindrome_1(n1);
    assertEquals(expectedResult, result);
}
```

Snippet of results



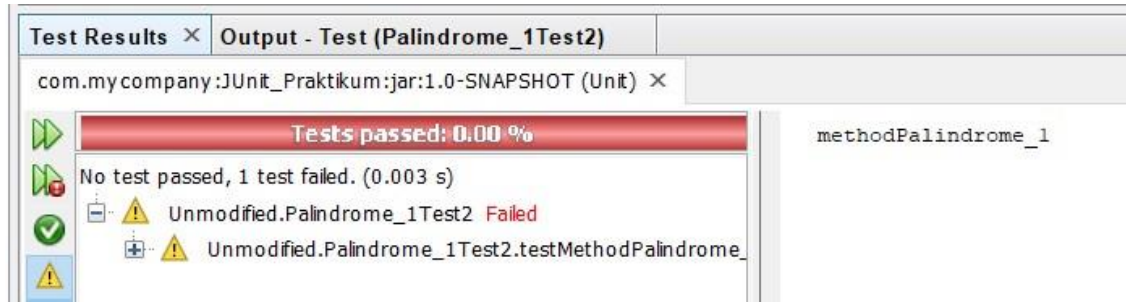
Pengujian pada test suite di atas dengan $n1 = 1$ dan expected result “palindrome number!” hasil dari pengujian tersebut error “Failed: expectednya harus menggunakan NOT palindrome number!”. Pada test suite 1 merupakan angka palindrome, maka kode di atas ada kesalahan dan perlu dilakukan perbaikan. Dimana angka palindrome merupakan angka yang jika dibalik hasilnya tetap sama.

b. Input: 22

Snippet of test case

```
@Test
public void testMethodPalindrome_1() {
    System.out.println("methodPalindrome_1");
    int n1 = 22;
    Palindrome_1 instance = new Palindrome_1();
    String expectedResult = "palindrome number!";
    String result = instance.methodPalindrome_1(n1);
    assertEquals(expectedResult, result);
}
```

Snippet of results



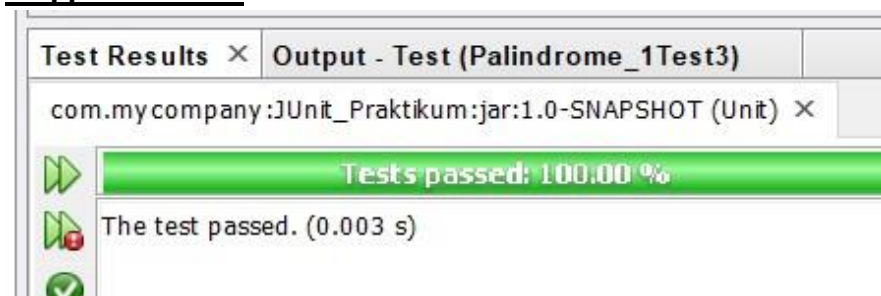
Pengujian pada test suite di atas dengan $n1 = 22$ dan expected result “palindrome number!” hasil dari pengujian tersebut error “Failed: expectednya harus menggunakan NOT palindrome number!”. Pada test suite 22 merupakan angka palindrome, maka kode di atas ada kesalahan dan perlu dilakukan perbaikan.

c. Input: 27

Snippet of test case

```
@Test
public void testMethodPalindrome_1() {
    System.out.println("methodPalindrome_1");
    int n1 = 27;
    Palindrome_1 instance = new Palindrome_1();
    String expectedResult = "NOT palindrome number!";
    String result = instance.methodPalindrome_1(n1);
    assertEquals(expectedResult, result);
}
```

Snippet of results

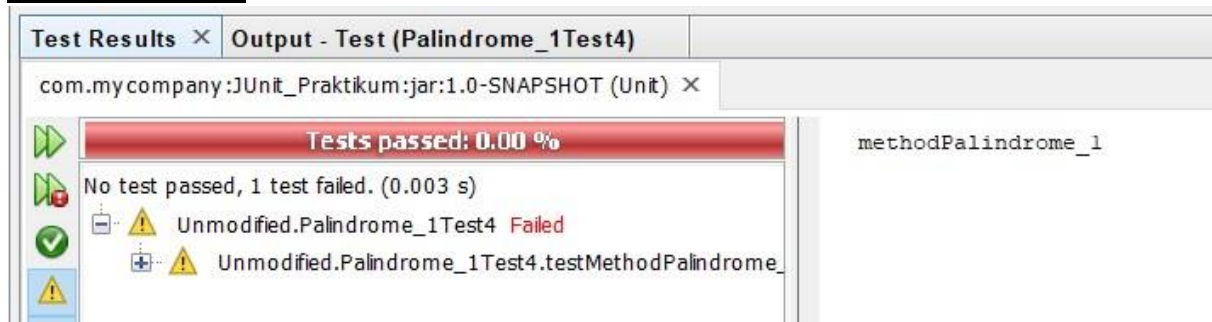


d. Input: 8998

Snippet of test case

```
@Test
| public void testMethodPalindrome_1() {
|     System.out.println("methodPalindrome_1");
|     int n1 = 8998;
|     Palindrome_1 instance = new Palindrome_1();
|     String expectedResult = "palindrome number!";
|     String result = instance.methodPalindrome_1(n1);
|     assertEquals(expectedResult, result);
| }
}
```

Snippet of results



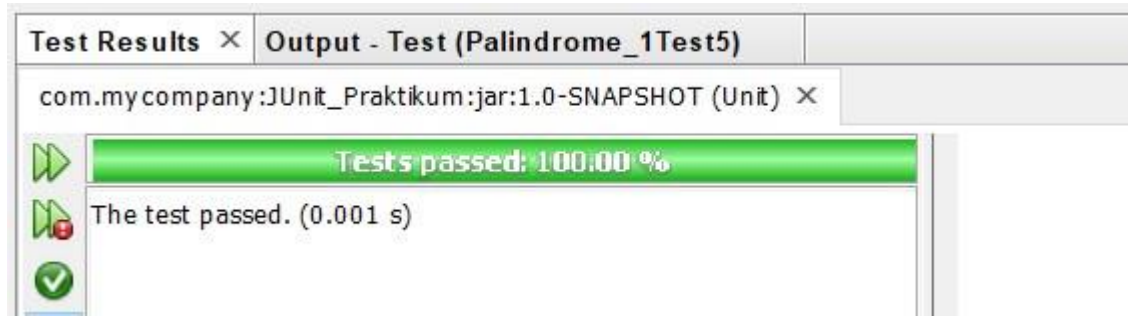
Pengujian pada test suite di atas dengan $n1 = 8998$ dan expected result “palindrome number!” hasil dari pengujian tersebut error “Failed: expectednya harus menggunakan NOT palindrome number!”. Pada test suite 8998 merupakan angka palindrome, maka kode di atas ada kesalahan dan perlu dilakukan perbaikan.

e. Input: 2373

Snippet of test case

```
@Test
| public void testMethodPalindrome_1() {
|     System.out.println("methodPalindrome_1");
|     int n1 = 2373;
|     Palindrome_1 instance = new Palindrome_1();
|     String expectedResult = "NOT palindrome number!";
|     String result = instance.methodPalindrome_1(n1);
|     assertEquals(expectedResult, result);
| }
}
```

Snippet of results

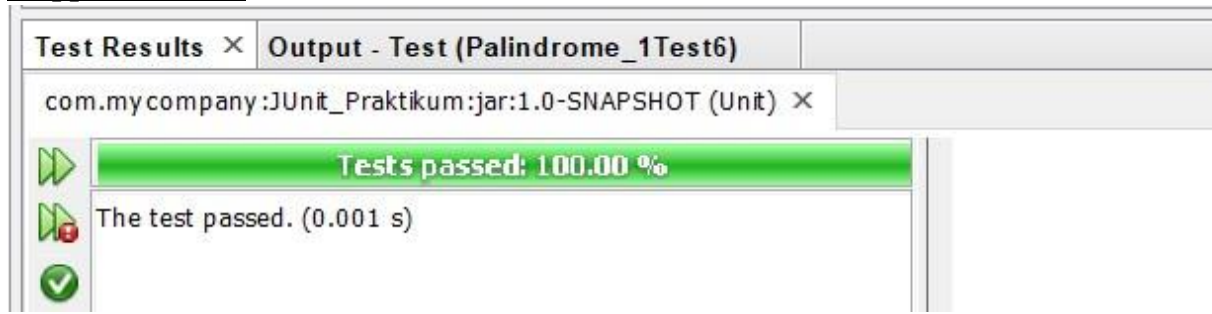


f. Input: 78938

Snippet of test case

```
@Test
[   public void testMethodPalindrome_1() {
        System.out.println("methodPalindrome_1");
        int n1 = 78938;
        Palindrome_1 instance = new Palindrome_1();
        String expectedResult = "NOT palindrome number!";
        String result = instance.methodPalindrome_1(n1);
        assertEquals(expectedResult, result);
    }
}
```

Snippet of results

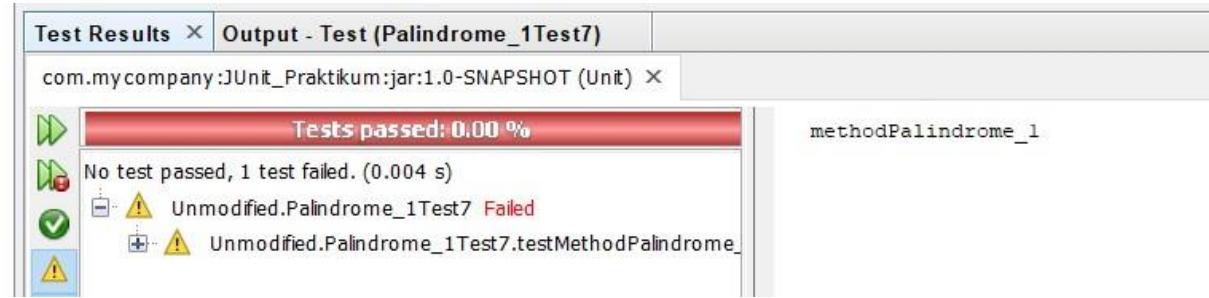


g. Input: 1834554381

Snippet of test case

```
@Test
[   public void testMethodPalindrome_1() {
        System.out.println("methodPalindrome_1");
        int n1 = 1834554381;
        Palindrome_1 instance = new Palindrome_1();
        String expectedResult = "palindrome number!";
        String result = instance.methodPalindrome_1(n1);
        assertEquals(expectedResult, result);
    }
}
```

Snippet of results



Pengujian pada test suite di atas dengan n1= 1834554381 dan expected result “palindrome number!” hasil dari pengujian tersebut error “Failed: expectednya harus menggunakan NOT palindrome number!”. Pada test suite 1834554381 merupakan angka palindrome, maka kode di atas ada kesalahan dan perlu dilakukan perbaikan.

2. Testing for Palindrome_2.java

```
package Unmodified;

import java.util.Scanner; public
class Palindrome_2 {
    public String methodPalindrome_2(String original){
        String reverse = "";
        String hasil;
        Scanner in = new Scanner(System.in);

        int length = original.length();



        for(int i=length-1; i>=0; i--){
            reverse = reverse +
original.charAt(i);
        }
        if(original.equals(reverse))      hasil =
"palindrome string!";      else
            hasil = "NOT palindrome string!";
        return hasil;
    }
}
```


a. Input: a

Snippet of test case

```
@Test
public void testMethodPalindrome_2() {
    System.out.println("methodPalindrome_2");
    String original = "a";
    Palindrome_2 instance = new Palindrome_2();
    String expResult = "palindrome string!";
    String result = instance.methodPalindrome_2(original);
    assertEquals(expResult, result);
    // TODO review the generated test code and remove the default call to fail.
    //fail("The test case is a prototype.");
}
```

Snippet of results

Test Results	Output - Test (Palindrome_2Test1)
com.mycompany:JUnit_Praktikum:jar:1.0-SNAPSHOT (Unit) X	
	Tests passed: 100.00 %
	The test passed. (0.029 s)
	

b. Input: is

Snippet of test case

```
@Test
public void testMethodPalindrome_2() {
    System.out.println("methodPalindrome_2");
    String original = "is";
    Palindrome_2 instance = new Palindrome_2();
    String expResult = "NOT palindrome string!";
    String result = instance.methodPalindrome_2(original);
    assertEquals(expResult, result);
    // TODO review the generated test code and remove the default call to fail.
    //fail("The test case is a prototype.");
}
```

Snippet of results

Test Results	Output - Test (Palindrome_2Test2)
com.mycompany:JUnit_Praktikum:jar:1.0-SNAPSHOT (Unit) X	
	Tests passed: 100.00 %
	The test passed. (0.038 s)
	




Input

Snippet of test case

c. : isi

```
@Test
public void testMethodPalindrome_2() {
    System.out.println("methodPalindrome_2");
    String original = "isi";
    Palindrome_2 instance = new Palindrome_2();
    String expResult = "palindrome string!";
    String result = instance.methodPalindrome_2(original);
    assertEquals(expResult, result);
    // TODO review the generated test code and remove the default call to fail.
    //fail("The test case is a prototype.");
}
```

Snippet of results

Test Results	Output - Test (Palindrome_2Test3)
com.mycompany:JUnit_Praktikum:jar:1.0-SNAPSHOT (Unit) X	
	Tests passed: 100.00 %
	The test passed. (0.036 s)
	

d. Input: radar

Snippet of test case

```
@Test
public void testMethodPalindrome_2() {
    System.out.println("methodPalindrome_2");
    String original = "radar";
    Palindrome_2 instance = new Palindrome_2();
    String expResult = "palindrome string!";
    String result = instance.methodPalindrome_2(original);
    assertEquals(expResult, result);
    // TODO review the generated test code and remove the default call to fail.
    //fail("The test case is a prototype.");
}
```

Snippet of results

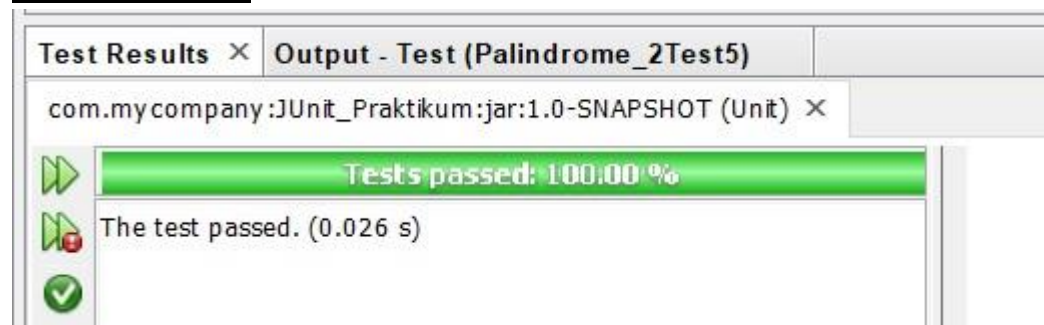
Test Results	Output - Test (Palindrome_2Test4)
com.mycompany:JUnit_Praktikum:jar:1.0-SNAPSHOT (Unit) X	
	Tests passed: 100.00 %
	The test passed. (0.036 s)
	

e. Input: palindrome

Snippet of test case

```
@Test
public void testMethodPalindrome_2() {
    System.out.println("methodPalindrome_2");
    String original = "palindrome";
    Palindrome_2 instance = new Palindrome_2();
    String expectedResult = "NOT palindrome string!";
    String result = instance.methodPalindrome_2(original);
    assertEquals(expResult, result);
    // TODO review the generated test code and remove the default call to fail.
    //fail("The test case is a prototype.");
}
```

Snippet of results



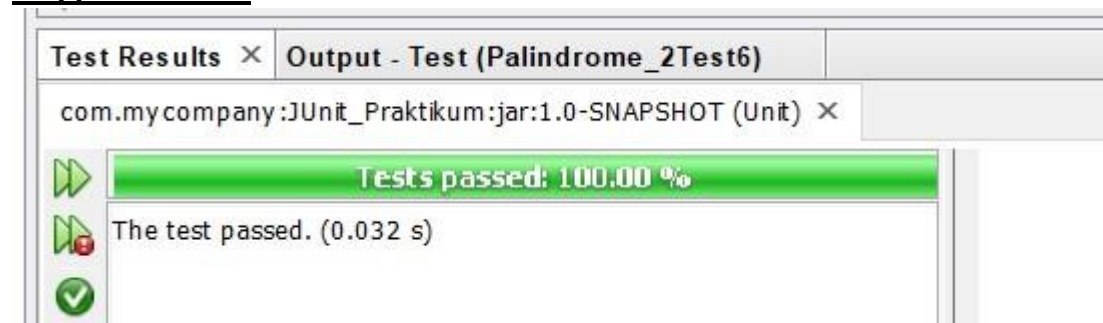
The screenshot shows the JUnit test results for the 'palindrome' input. The window title is 'Test Results x Output - Test (Palindrome_2Test5)'. The package path is 'com.mycompany:JUnit_Praktikum:jar:1.0-SNAPSHOT (Unit) x'. A green progress bar indicates 'Tests passed: 100.00 %'. Below the bar, it says 'The test passed. (0.026 s)'. On the left side, there are three icons: a green arrow pointing right, a green arrow pointing left with a red circle, and a green checkmark.

f. Input: nababan

Snippet of test case

```
@Test
public void testMethodPalindrome_2() {
    System.out.println("methodPalindrome_2");
    String original = "nababan";
    Palindrome_2 instance = new Palindrome_2();
    String expectedResult = "palindrome string!";
    String result = instance.methodPalindrome_2(original);
    assertEquals(expResult, result);
    // TODO review the generated test code and remove the default call to fail.
    //fail("The test case is a prototype.");
}
```

Snippet of results



The screenshot shows the JUnit test results for the 'nababan' input. The window title is 'Test Results x Output - Test (Palindrome_2Test6)'. The package path is 'com.mycompany:JUnit_Praktikum:jar:1.0-SNAPSHOT (Unit) x'. A green progress bar indicates 'Tests passed: 100.00 %'. Below the bar, it says 'The test passed. (0.032 s)'. On the left side, there are three icons: a green arrow pointing right, a green arrow pointing left with a red circle, and a green checkmark.

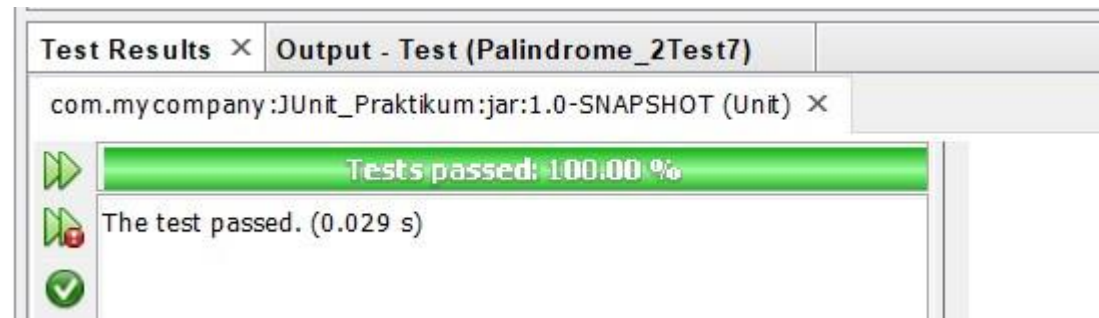
Input

Snippet of test case

g. : read

```
@Test
public void testMethodPalindrome_2() {
    System.out.println("methodPalindrome_2");
    String original = "read";
    Palindrome_2 instance = new Palindrome_2();
    String expectedResult = "NOT palindrome string!";
    String result = instance.methodPalindrome_2(original);
    assertEquals(expectedResult, result);
    // TODO review the generated test code and remove the default call to fail.
    //fail("The test case is a prototype.");
}
```

Snippet of results



3. Testing for Reverse_1.java

```
package Unmodified;
```

```
import java.io.BufferedReader; import
```

```
java.io.InputStreamReader; public
```

```
class Reverse_1 {
```

```
    public String methodReverse_1(int n){
```

```
        String hasil;
```

```
        BufferedReader br = new BufferedReader(new
```

```
InputStreamReader(System.in));        int r;        int rev = 0;
```

```
        int number = n;
```

```
        while(n>0){
```

```
            r = n%10;        rev
```

```
            = rev*10+r;
```

```
            n = n/10;
```

```
        }
```

```
        hasil = "The reverse of "+number+ " is "+rev;
```

```
        return hasil;
```

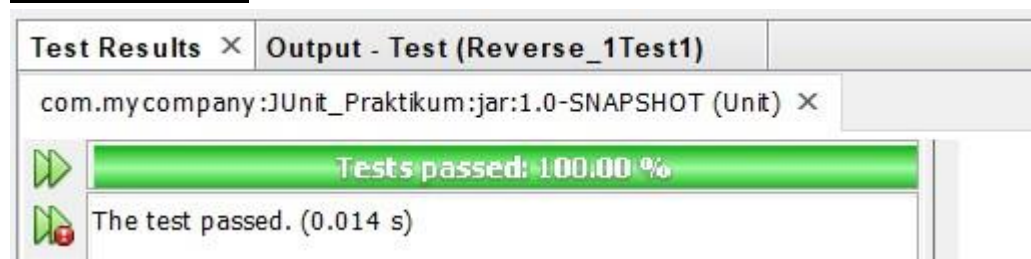
```
    }
```

```
}
```

a. : 1

```
@Test
public void testMethodReverse_1() {
    System.out.println("methodReverse_1");
    int n = 1;
    Reverse_1 instance = new Reverse_1();
    String expectedResult = "The reverse of 1 is 1";
    String result = instance.methodReverse_1(n);
    assertEquals(expResult, result);
    // TODO review the generated test code and remove the default call to fail.
    //fail("The test case is a prototype.");
}
```

Snippet of results



b. Input: 22

Snippet of test case

```
@Test
public void testMethodReverse_1() {
    System.out.println("methodReverse_1");
    int n = 22;
    Reverse_1 instance = new Reverse_1();
    String expectedResult = "The reverse of 22 is 22";
    String result = instance.methodReverse_1(n);
    assertEquals(expResult, result);
    // TODO review the generated test code and remove the default call to fail.
    //fail("The test case is a prototype.");
}
```

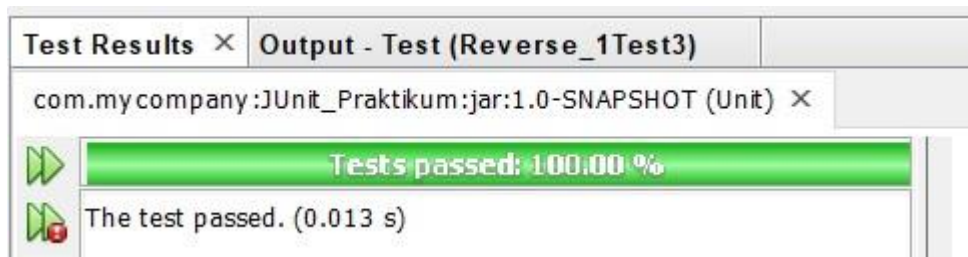
Snippet of results



c. : 27

```
@Test
public void testMethodReverse_1() {
    System.out.println("methodReverse_1");
    int n = 27;
    Reverse_1 instance = new Reverse_1();
    String expResult = "The reverse of 27 is 72";
    String result = instance.methodReverse_1(n);
    assertEquals(expResult, result);
    // TODO review the generated test code and remove the default call to fail.
    //fail("The test case is a prototype.");
}
```

Snippet of results

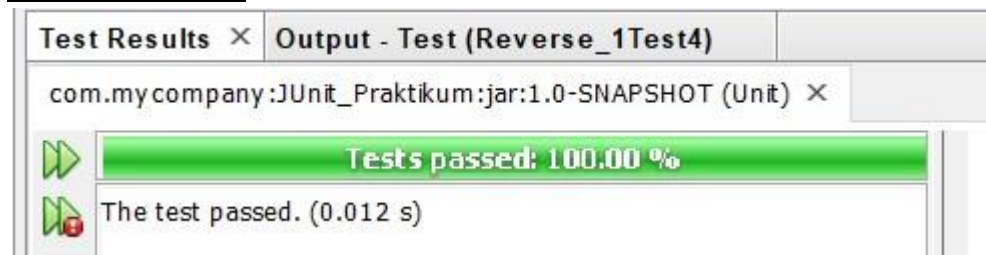


d. Input: 8998

Snippet of test case

```
@Test
public void testMethodReverse_1() {
    System.out.println("methodReverse_1");
    int n = 8998;
    Reverse_1 instance = new Reverse_1();
    String expResult = "The reverse of 8998 is 8998";
    String result = instance.methodReverse_1(n);
    assertEquals(expResult, result);
    // TODO review the generated test code and remove the default call to fail.
    //fail("The test case is a prototype.");
}
```

Snippet of results



e. : 2373

```
@Test
public void testMethodReverse_1() {
    System.out.println("methodReverse_1");
    int n = 2373;
    Reverse_1 instance = new Reverse_1();
    String expectedResult = "The reverse of 2373 is 3732";
    String result = instance.methodReverse_1(n);
    assertEquals(expResult, result);
    // TODO review the generated test code and remove the default call to fail.
    //fail("The test case is a prototype.");
}
```

Snippet of results

Test Results	Output - Test (Reverse_1Test5)
com.mycompany:JUnit_Praktikum:jar:1.0-SNAPSHOT (Unit)	
Tests passed: 100.00 %	
The test passed. (0.011 s)	

f. Input: 78938

Snippet of test case

```
@Test
public void testMethodReverse_1() {
    System.out.println("methodReverse_1");
    int n = 78938;
    Reverse_1 instance = new Reverse_1();
    String expectedResult = "The reverse of 78938 is 83987";
    String result = instance.methodReverse_1(n);
    assertEquals(expResult, result);
    // TODO review the generated test code and remove the default call to fail.
    //fail("The test case is a prototype.");
}
```

Snippet of results

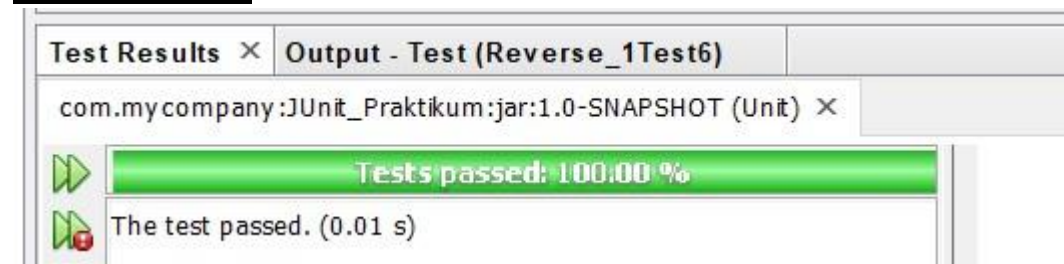
Test Results	Output - Test (Reverse_1Test6)
com.mycompany:JUnit_Praktikum:jar:1.0-SNAPSHOT (Unit)	
Tests passed: 100.00 %	
The test passed. (0.011 s)	

g. : 1834554381

Snippet of test case

```
@Test
public void testMethodReverse_1() {
    System.out.println("methodReverse_1");
    int n = 1834554381;
    Reverse_1 instance = new Reverse_1();
    String expResult = "The reverse of 1834554381 is 1834554381";
    String result = instance.methodReverse_1(n);
    assertEquals(expResult, result);
    // TODO review the generated test code and remove the default call to fail.
    //fail("The test case is a prototype.");
}
```

Snippet of results



4. Testing for Reverse_2.java

```
package Unmodified;

public class Reverse_2 {
    public String methodReverse_2(String original){
        String hasil;
        String reverse = "";
        int length = original.length();
        for(int i=length-1; i>0; i--){
            reverse = reverse + original.charAt(i);
        }
        hasil = "The reverse of "+original+" is "+reverse;
        return hasil;
    }
}
```


a. : a

Snippet of test case

```
@Test
public void testMethodReverse_2() {
    System.out.println("methodReverse_2");
    String original = "a";
    Reverse_2 instance = new Reverse_2();
    String expResult = "The reverse of a is a";
    String result = instance.methodReverse_2(original);
    assertEquals(expResult, result);
    // TODO review the generated test code and remove the default call to fail.
    //fail("The test case is a prototype.");
}
```

Snippet of results

Test Results	Output - Test (Reverse_2Test1)
com.mycompany:JUnit_Praktikum:jar:1.0-SNAPSHOT (Unit) X	
Tests passed: 0.00 %	
No test passed, 1 test failed. (0.017 s)	
Unmodified.Reverse_2Test1 Failed	
Unmodified.Reverse_2Test1.testMethodReverse_2 Failed	

Pengujian pada test suite di atas dengan string= a dan expected result "The reserve of a is a" hasil dari pengujian tersebut error, maka kode di atas terdapat kesalahan dan perlu dilakukan perbaikan.

b. Input: is

Snippet of test case

```
@Test
public void testMethodReverse_2() {
    System.out.println("methodReverse_2");
    String original = "is";
    Reverse_2 instance = new Reverse_2();
    String expResult = "The reverse of is is si";
    String result = instance.methodReverse_2(original);
    assertEquals(expResult, result);
    // TODO review the generated test code and remove the default call to fail.
    //fail("The test case is a prototype.");
}
```

Snippet of results

Test Results	Output - Test (Reverse_2Test2)
com.mycompany:JUnit_Praktikum:jar:1.0-SNAPSHOT (Unit) X	
Tests passed: 0.00 %	
No test passed, 1 test failed. (0.023 s)	
Unmodified.Reverse_2Test2 Failed	
Unmodified.Reverse_2Test2.testMethodReverse_2 Failed	

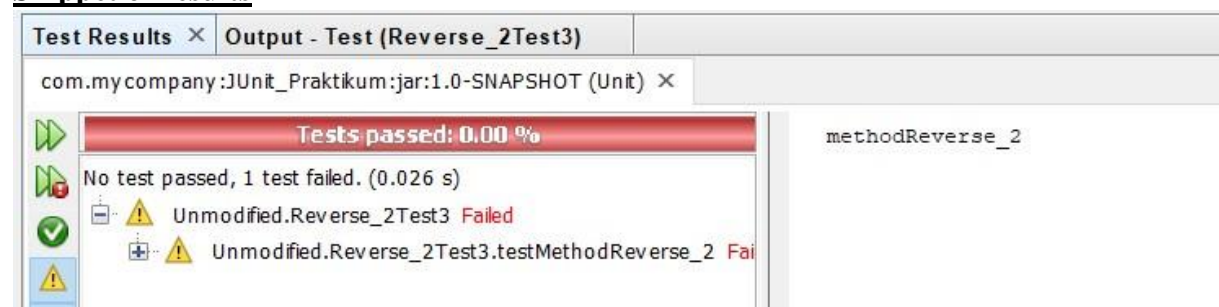
Pengujian pada test suite di atas dengan string= isi dan expected result “The reserve of isi is si” hasil dari pengujian tersebut error, maka kode di atas terdapat kesalahan dan perlu dilakukan perbaikan.

c. Input: isi

Snippet of test case

```
@Test
public void testMethodReverse_2() {
    System.out.println("methodReverse_2");
    String original = "isi";
    Reverse_2 instance = new Reverse_2();
    String expectedResult = "The reverse of isi is isi";
    String result = instance.methodReverse_2(original);
    assertEquals(expectedResult, result);
    // TODO review the generated test code and remove the default call to fail.
    //fail("The test case is a prototype.");
}
```

Snippet of results



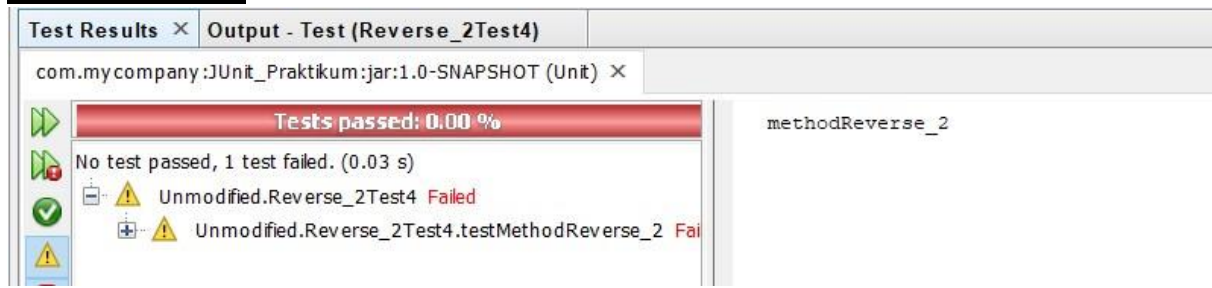
Pengujian pada test suite di atas dengan string= isi dan expected result “The reserve of isi is isi ” hasil dari pengujian tersebut error, maka kode di atas terdapat kesalahan dan perlu dilakukan perbaikan.

d. Input: radar

Snippet of test case

```
@Test
public void testMethodReverse_2() {
    System.out.println("methodReverse_2");
    String original = "radar";
    Reverse_2 instance = new Reverse_2();
    String expectedResult = "The reverse of radar is radar";
    String result = instance.methodReverse_2(original);
    assertEquals(expectedResult, result);
    // TODO review the generated test code and remove the default call to fail.
    //fail("The test case is a prototype.");
}
```

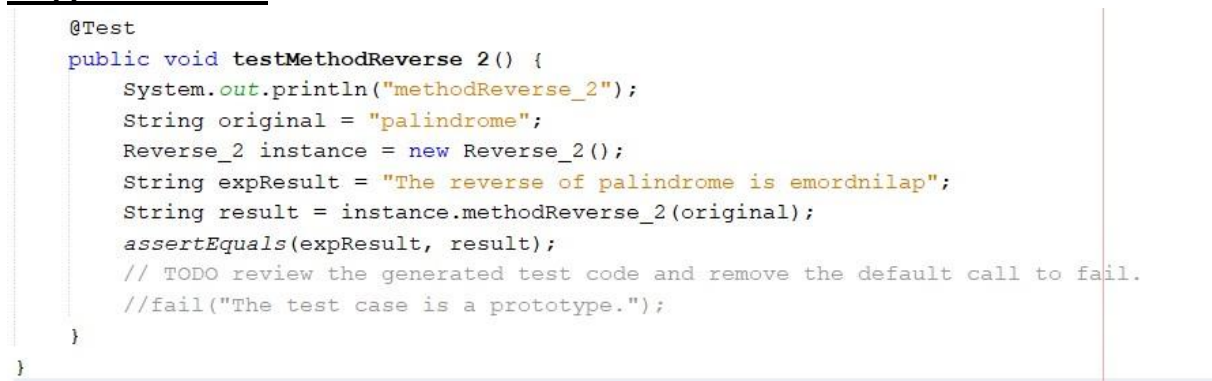
Snippet of results



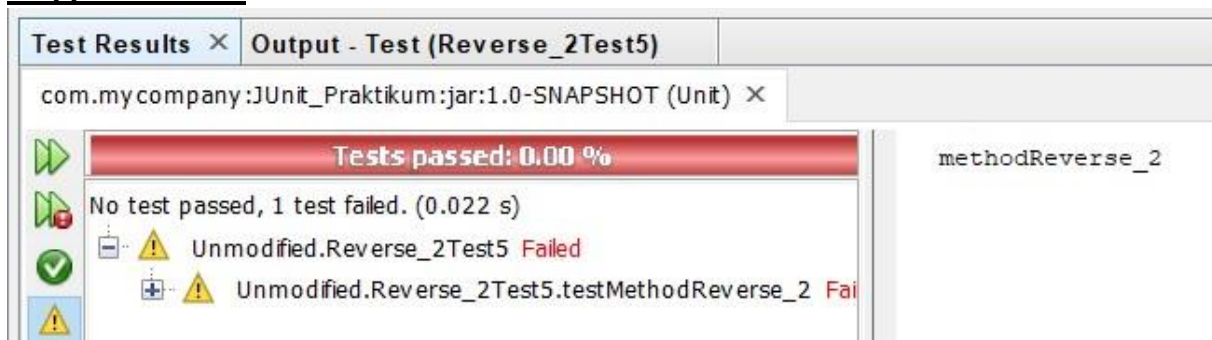
Pengujian pada test suite di atas dengan string= radar dan expected result “The reserve of radar is radar ” hasil dari pengujian tersebut error, maka kode di atas terdapat kesalahan dan perlu dilakukan perbaikan.

e. Input: palindrome

Snippet of test case



Snippet of results



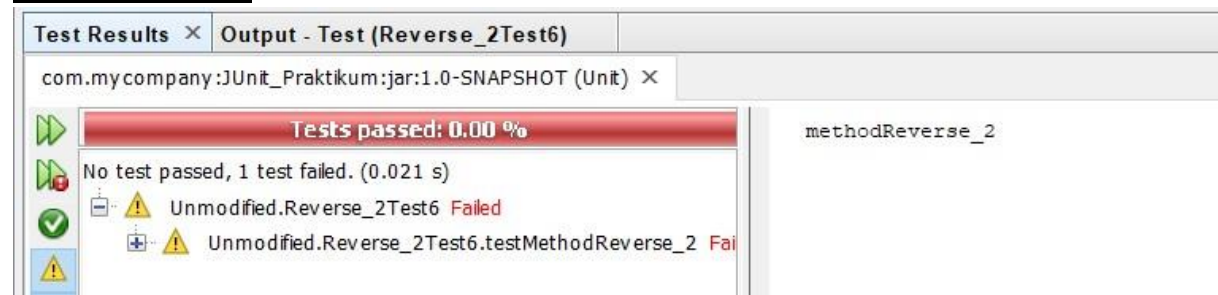
Pengujian pada test suite di atas dengan string= palindrome dan expected result “The reserve of palindrome is emordnilap ” hasil dari pengujian tersebut error, maka kode di atas terdapat kesalahan dan perlu dilakukan perbaikan.

f. Input: nababan

Snippet of test case

```
@Test
public void testMethodReverse_2() {
    System.out.println("methodReverse_2");
    String original = "nababan";
    Reverse_2 instance = new Reverse_2();
    String expectedResult = "The reverse of nababan is nababan";
    String result = instance.methodReverse_2(original);
    assertEquals(expectedResult, result);
    // TODO review the generated test code and remove the default call to fail.
    //fail("The test case is a prototype.");
}
```

Snippet of results



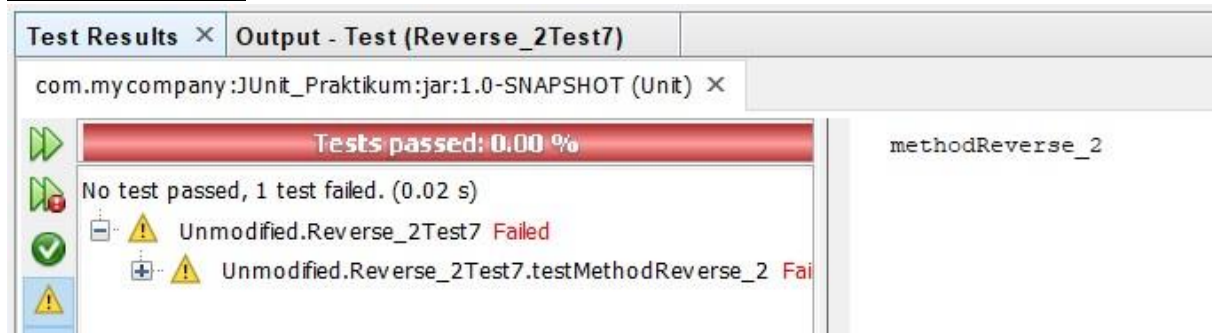
Pengujian pada test suite di atas dengan string= nababan dan expected result “The reserve of nababan is nababan ” hasil dari pengujian tersebut error, maka kode di atas terdapat kesalahan dan perlu dilakukan perbaikan.

g. Input: read

Snippet of test case

```
@Test
public void testMethodReverse_2() {
    System.out.println("methodReverse_2");
    String original = "read";
    Reverse_2 instance = new Reverse_2();
    String expectedResult = "The reverse of read is daer";
    String result = instance.methodReverse_2(original);
    assertEquals(expectedResult, result);
    // TODO review the generated test code and remove the default call to fail.
    //fail("The test case is a prototype.");
}
```

Snippet of results



Pengujian pada test suite di atas dengan string= read dan expected result “The reserve of read is daer ” hasil dari pengujian tersebut error, maka kode di atas terdapat kesalahan dan perlu dilakukan perbaikan.

Testing for Modified Program

Testing yang sebelumnya telah diuji pada ke empat file java yang sudah disediakan terdapat 2 file java yang memiliki failure diantaranya:

1. Testing for palindrome_1.java

```
package Modified;

import java.io.BufferedReader;
import java.io.InputStreamReader;

public class Palindrome_1 {
    public String methodPalindrome_1(int n1){
        String hasil;
        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
        int r, n2;    int rev=0;    n2=n1;    while(n1>0){        r = n1%10;
        rev = rev*10+r;        n1 = n1/10;
    }

    if(rev==n2){
        hasil = "palindrome number!";
    } else{
        hasil = "NOT palindrome number!";
    }
    return hasil;
}
}
```

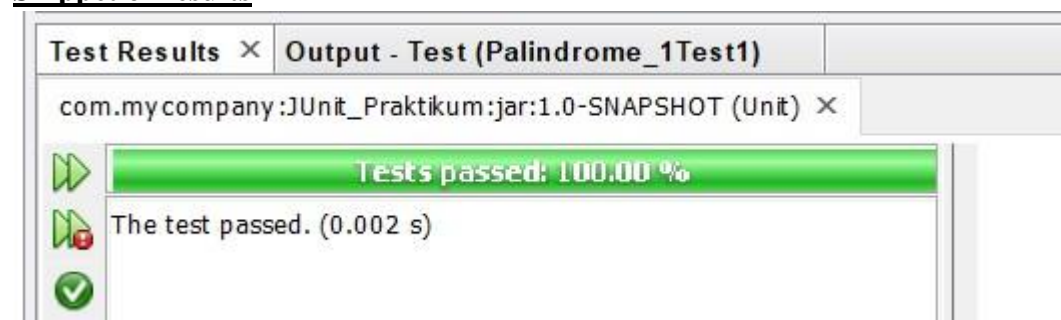

Perubahan code terletak pada $n1 = n1/10$. Sehingga ketika dilakukan pengujian dengan membuat inputan dengan tipe data int dapat mengecek apakah data yang diinput merupakan data palindrome maupun tidak.

a. Input: 1

Snippet of test case

```
@Test
public void testMethodPalindrome_1() {
    System.out.println("methodPalindrome_1");
    int n1 = 1;
    Palindrome_1 instance = new Palindrome_1();
    String expectedResult = "palindrome number!";
    String result = instance.methodPalindrome_1(n1);
    assertEquals(expResult, result);
    // TODO review the generated test code and remove the default call to fail.
    //fail("The test case is a prototype.");
}
```

Snippet of results

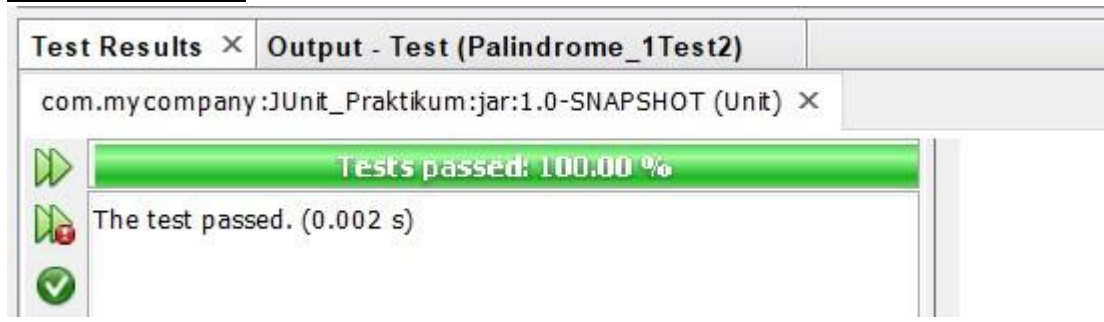


b. Input: 22

Snippet of test case

```
@Test
public void testMethodPalindrome_1() {
    System.out.println("methodPalindrome_1");
    int n1 = 22;
    Palindrome_1 instance = new Palindrome_1();
    String expectedResult = "palindrome number!";
    String result = instance.methodPalindrome_1(n1);
    assertEquals(expResult, result);
    // TODO review the generated test code and remove the default call to fail.
    //fail("The test case is a prototype.");
}
```


Snippet of results

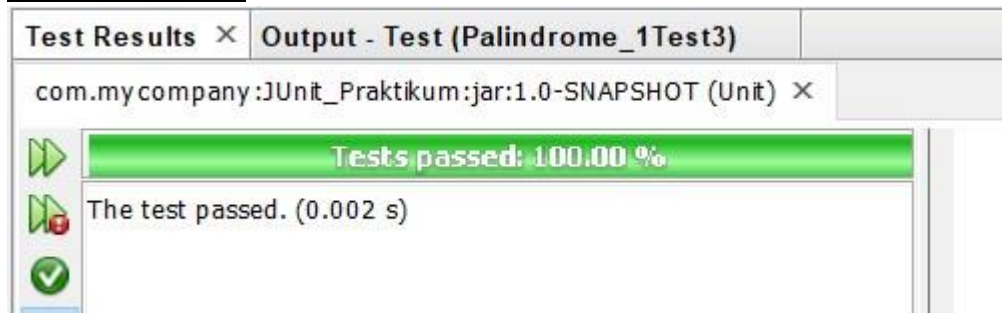


c. Input: 27

Snippet of test case

```
@Test
public void testMethodPalindrome_1() {
    System.out.println("methodPalindrome_1");
    int n1 = 27;
    Palindrome_1 instance = new Palindrome_1();
    String expResult = "NOT palindrome number!";
    String result = instance.methodPalindrome_1(n1);
    assertEquals(expResult, result);
    // TODO review the generated test code and remove the default call to fail.
    //fail("The test case is a prototype.");
}
```

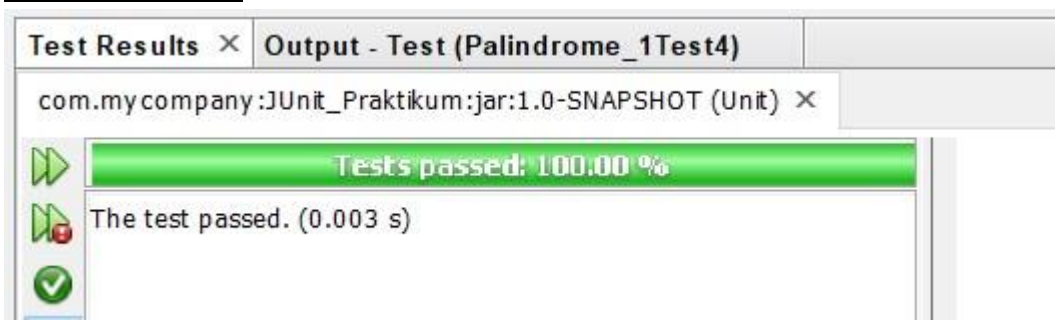
Snippet of results



d. Input: 8998 Snippet of test case

```
@Test
public void testMethodPalindrome_1() {
    System.out.println("methodPalindrome_1");
    int n1 = 8998;
    Palindrome_1 instance = new Palindrome_1();
    String expResult = "palindrome number!";
    String result = instance.methodPalindrome_1(n1);
    assertEquals(expResult, result);
    // TODO review the generated test code and remove the default call to fail.
    //fail("The test case is a prototype.");
}
```

Snippet of results

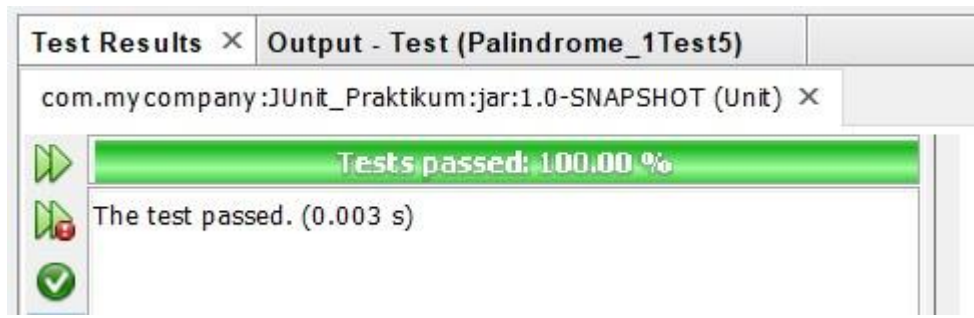


e. Input: 2373

Snippet of test case

```
@Test
public void testMethodPalindrome_1() {
    System.out.println("methodPalindrome_1");
    int n1 = 2373;
    Palindrome_1 instance = new Palindrome_1();
    String expResult = "NOT palindrome number!";
    String result = instance.methodPalindrome_1(n1);
    assertEquals(expResult, result);
    // TODO review the generated test code and remove the default call to fail.
    //fail("The test case is a prototype.");
}
```

Snippet of results

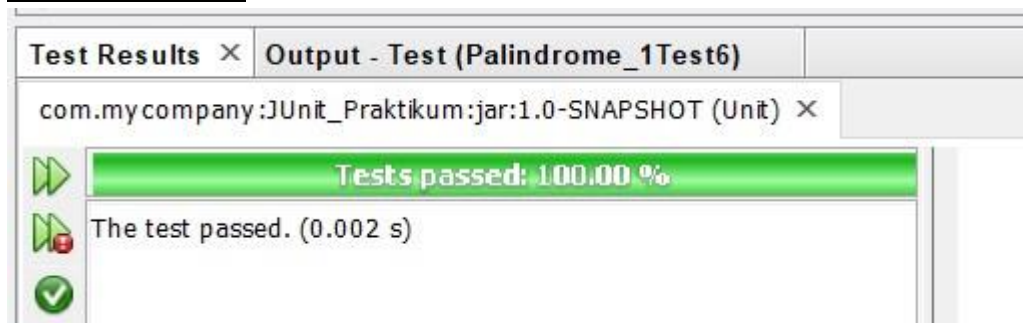


f. Input: 78938

Snippet of test case

```
@Test
public void testMethodPalindrome 1() {
    System.out.println("methodPalindrome_1");
    int n1 = 78938;
    Palindrome_1 instance = new Palindrome_1();
    String expResult = "NOT palindrome number!";
    String result = instance.methodPalindrome_1(n1);
    assertEquals(expResult, result);
    // TODO review the generated test code and remove the default call to fail.
    //fail("The test case is a prototype.");
}
```

Snippet of results

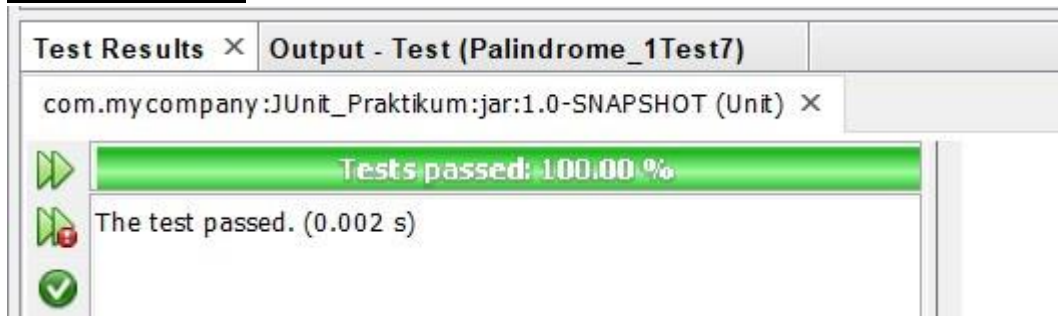


g. Input: 1834554381

Snippet of test case

```
@Test
public void testMethodPalindrome_1() {
    System.out.println("methodPalindrome_1");
    int n1 = 1834554381;
    Palindrome_1 instance = new Palindrome_1();
    String expectedResult = "palindrome number!";
    String result = instance.methodPalindrome_1(n1);
    assertEquals(expectedResult, result);
    // TODO review the generated test code and remove the default call to fail.
    //fail("The test case is a prototype.");
}
```

Snippet of results



2. Testing for Reverse_2.java

```
package Modified;

public class Reverse_2 {
    public String methodReverse_2(String original){
        String hasil;
        String reverse = "";    int
length = original.length();
for(int i=length-1; i>=0; i--)
    reverse = reverse + original.charAt(i);

        hasil = "The reverse of "+original+" is "+reverse;
return hasil;
    }
}
```

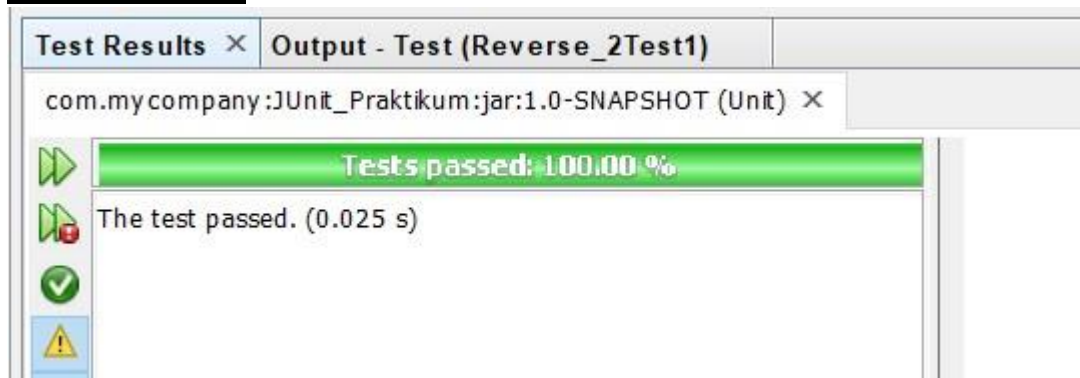
Perubahan code terletak pada `for(int i=length-1; i>=0; i--)` . Sehingga ketika dilakukan pengujian dengan membuat inputan dengan tipe data string akan dapat di reverse (membalikkan) string tersebut. Misalnya : buku menjadi ukub

a. Input: a

Snippet of test case

```
@Test
public void testMethodReverse_2() {
    System.out.println("methodReverse_2");
    String original = "a";
    Reverse_2 instance = new Reverse_2();
    String expResult = "The reverse of a is a";
    String result = instance.methodReverse_2(original);
    assertEquals(expResult, result);
    // TODO review the generated test code and remove the default call to fail.
    //fail("The test case is a prototype.");
}
```

Snippet of results

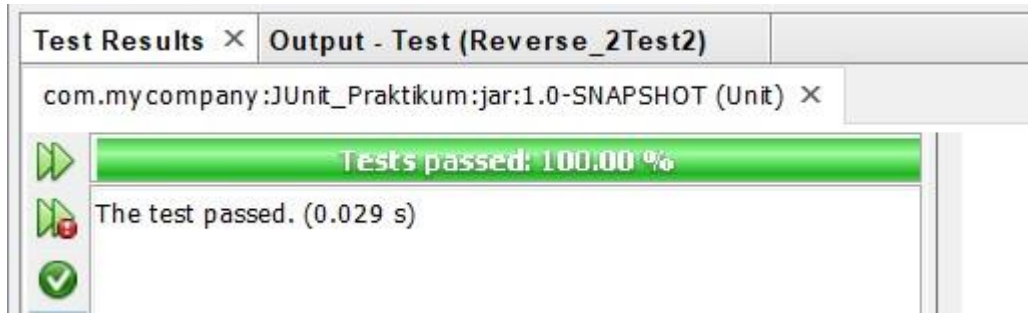


b. Input: is

Snippet of test case

```
@Test
public void testMethodReverse_2() {
    System.out.println("methodReverse_2");
    String original = "is";
    Reverse_2 instance = new Reverse_2();
    String expResult = "The reverse of is is si";
    String result = instance.methodReverse_2(original);
    assertEquals(expResult, result);
    // TODO review the generated test code and remove the default call to fail.
    //fail("The test case is a prototype.");
}
```

Snippet of results



c. Input: isi

Snippet of test case

```
@Test
public void testMethodReverse_2() {
    System.out.println("methodReverse_2");
    String original = "isi";
    Reverse_2 instance = new Reverse_2();
    String expectedResult = "The reverse of isi is isi";
    String result = instance.methodReverse_2(original);
    assertEquals(expectedResult, result);
    // TODO review the generated test code and remove the default call to fail.
    //fail("The test case is a prototype.");
}
```

Snippet of results

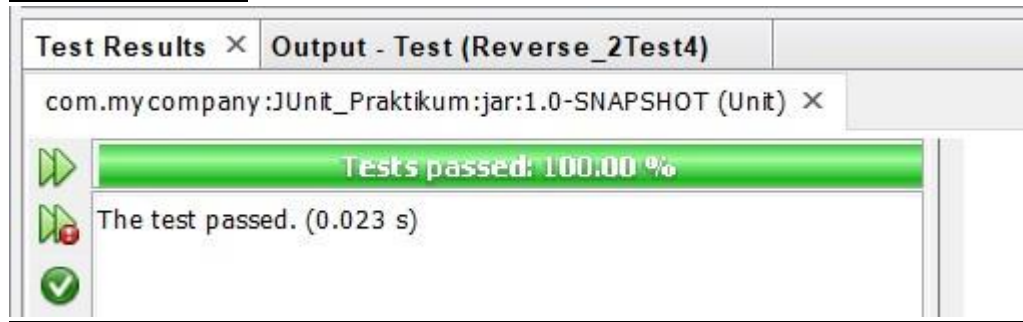


d. Input: radar

Snippet of test case

```
@Test
public void testMethodReverse_2() {
    System.out.println("methodReverse_2");
    String original = "radar";
    Reverse_2 instance = new Reverse_2();
    String expectedResult = "The reverse of radar is radar";
    String result = instance.methodReverse_2(original);
    assertEquals(expectedResult, result);
    // TODO review the generated test code and remove the default call to fail.
    //fail("The test case is a prototype.");
}
```


Snippet of results

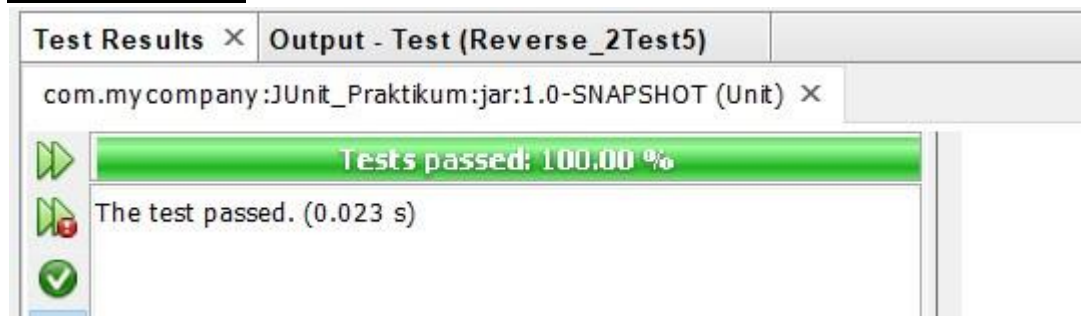


e. Input: palindrome

Snippet of test case

```
@Test
public void testMethodReverse_2() {
    System.out.println("methodReverse_2");
    String original = "palindrome";
    Reverse_2 instance = new Reverse_2();
    String expResult = "The reverse of palindrome is emordnilap";
    String result = instance.methodReverse_2(original);
    assertEquals(expResult, result);
    // TODO review the generated test code and remove the default call to fail.
    //fail("The test case is a prototype.");
}
```

Snippet of results

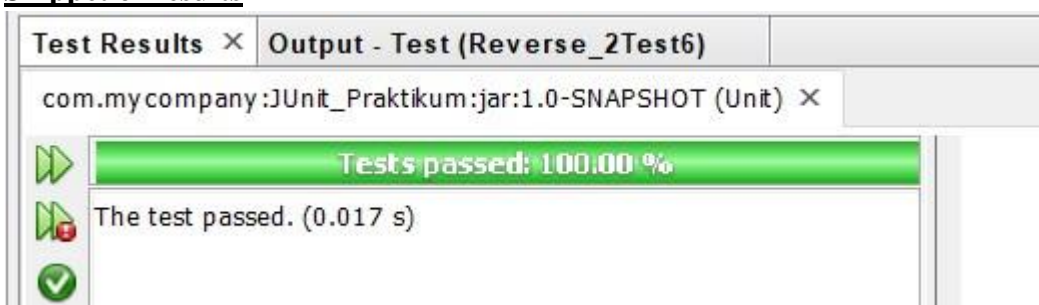


f. Input: nababan

Snippet of test case

```
@Test
public void testMethodReverse_2() {
    System.out.println("methodReverse_2");
    String original = "nababan";
    Reverse_2 instance = new Reverse_2();
    String expResult = "The reverse of nababan is nababan";
    String result = instance.methodReverse_2(original);
    assertEquals(expResult, result);
    // TODO review the generated test code and remove the default call to fail.
    //fail("The test case is a prototype.");
}
```

Snippet of results



g. Input: read

Snippet of test case

```
@Test
public void testMethodReverse_2() {
    System.out.println("methodReverse_2");
    String original = "read";
    Reverse_2 instance = new Reverse_2();
    String expResult = "The reverse of read is daer";
    String result = instance.methodReverse_2(original);
    assertEquals(expResult, result);
    // TODO review the generated test code and remove the default call to fail.
    //fail("The test case is a prototype.");
}
```

Snippet of results

