# ACS 560 Software Engineering

# Homework 09

# (Ashwini Kulkarni)

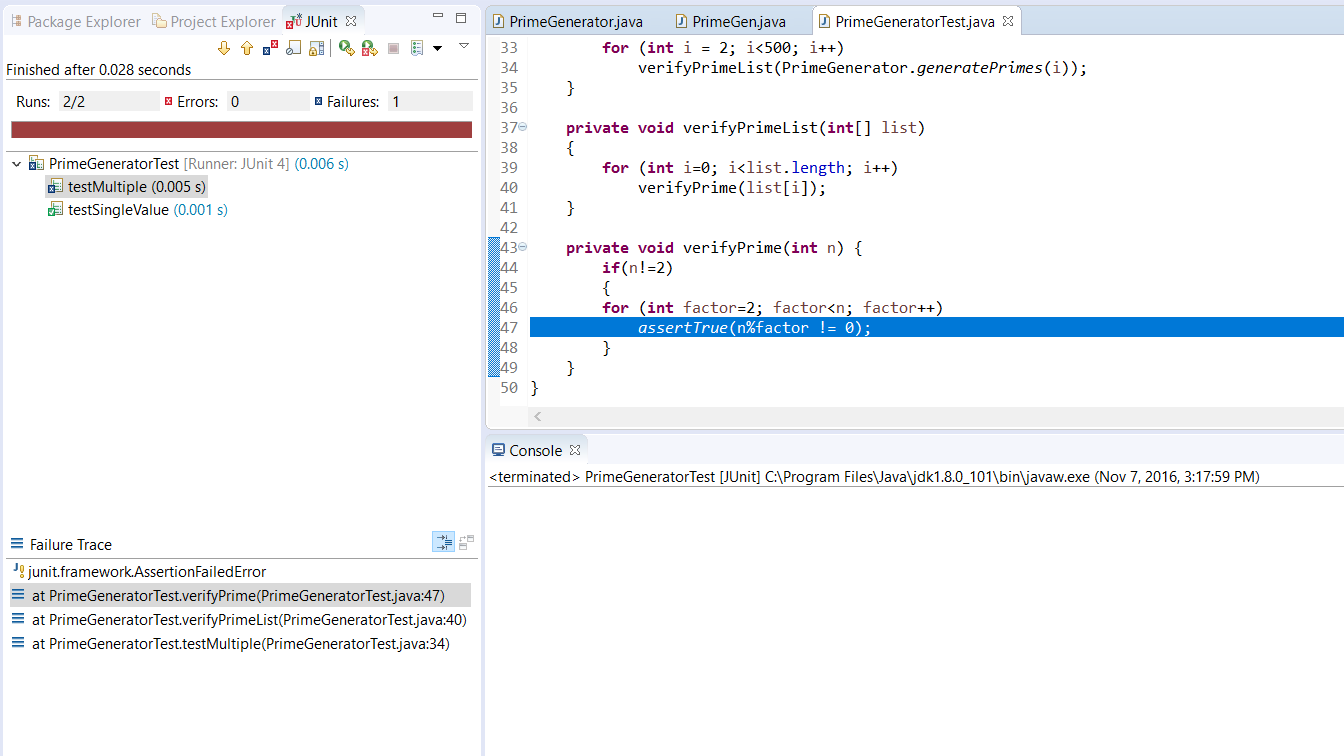
**Qus 1 : A simple stack class.**

Test Cases:

1. At beginning check stack is empty.
2. Test for push an item on the stack and pop it off.
3. Test for multiple distinct items on the stack, repeated popping returns them in reverse order.
4. Test for Pop operation when Stack is empty. (It should through exception)
5. A stack that has an item pushed is non-empty.
6. Test for A stack that has items pushed and subsequently removed is empty using push and pop
7. Test for top operation it should return Correct top after different combination of Push and pop operation
8. If stack has specific length (n) then have to check push operation for boundary check as n-1 th index .and nth should through error.

**Qus 2: Unit testing (JUnit)**

Junit test

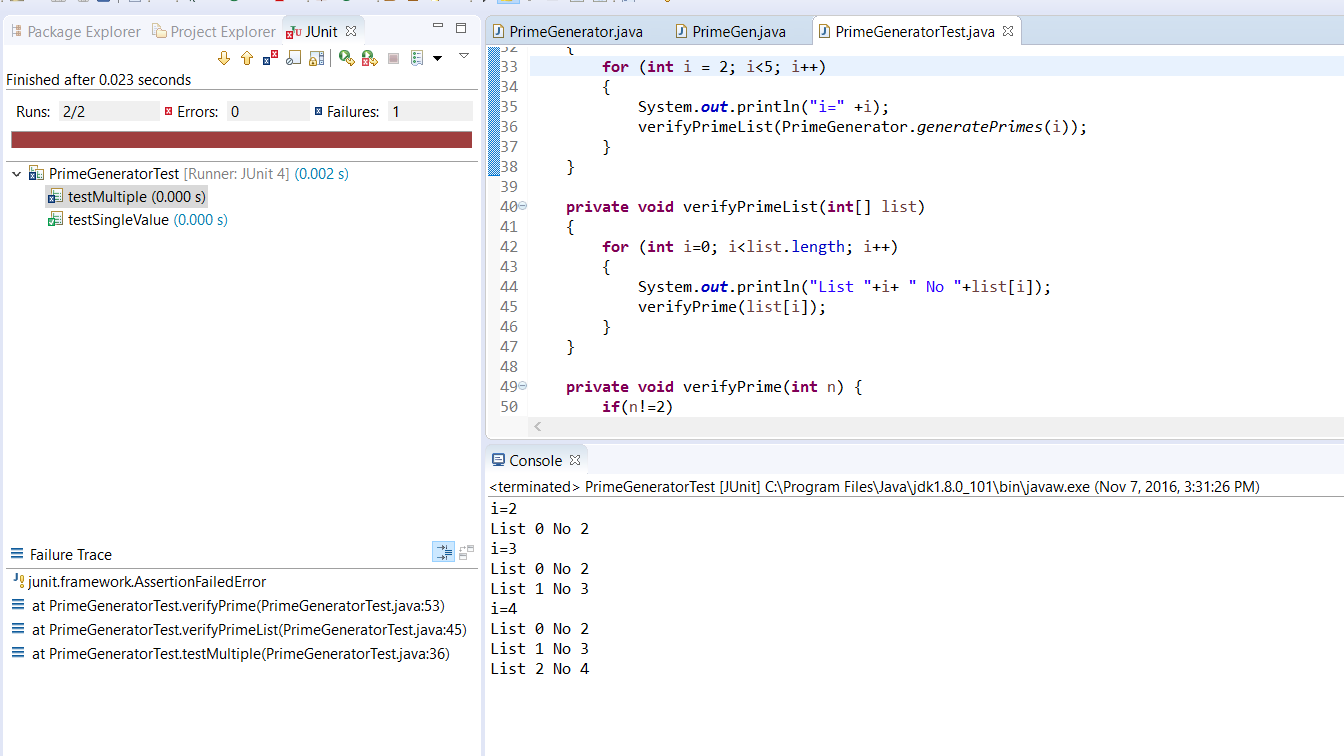


It Fails on assertTrue that means result showing non prime values as prime.

Way to fix:

Rerun test for small int range. i.e. <5

Also added “println” statement to get current value which is generated by program.



Here Output shows that generating prime nos. till 4 also showing 4 as prime no. which is actually not.

Go to main program:

Following fuction is responsible for checking value is prime or not

**private** **static** **void** crossOutMultiplesOf(**int** i)

{

**for** (**int** multiple = 2\*i;

multiple < *crossedOut*.length;

multiple += i)

*crossedOut*[multiple] = **true**;

}

Here ‘i’ value comes from following function:

**private** **static** **void** crossOutMultiples()

{

**int** limit = *determineIterationLimit*();

**for** (**int** i = 2; i < limit; i++)

**if** (*notCrossed*(i))

*crossOutMultiplesOf*(i);

}

Consider for value 4, determineIterationLimit() will return value 2 as limit.

For loop is i<2 and starting with 2 only. That mean this loop will not run, so there will be no call for function crossOutMultiplesOf(int i).

As per algorithm it should run for limit range too so it can call crossOutMultiplesOf(int i). on limit value.

So next is code fix.

**Code Fix:**

Old Code

**private** **static** **void** crossOutMultiples()

{

**int** limit = *determineIterationLimit*();

**for** (**int** i = 2; i <limit; i++)

**if** (*notCrossed*(i))

*crossOutMultiplesOf*(i);

}

New Code:

**private** **static** **void** crossOutMultiples()

{

**int** limit = *determineIterationLimit*();

**for** (**int** i = 2; i <= limit; i++)

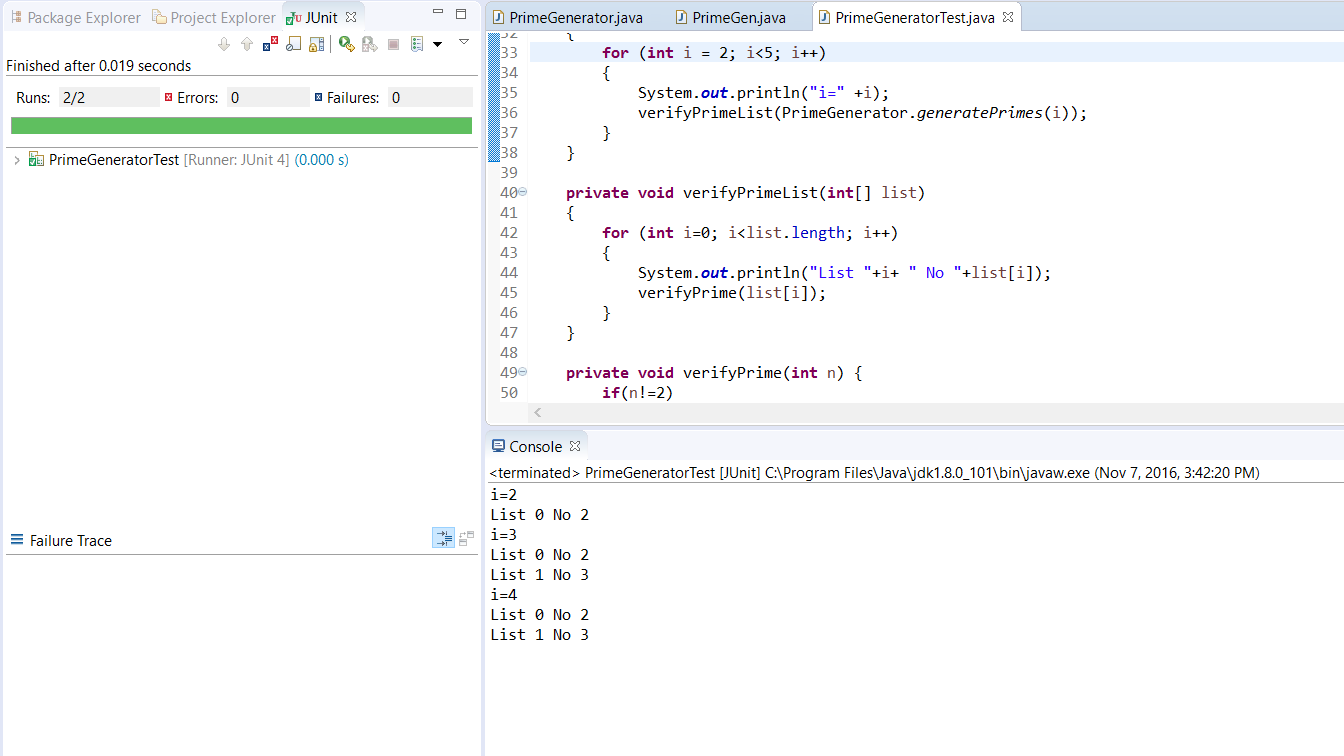
**if** (*notCrossed*(i))

*crossOutMultiplesOf*(i);

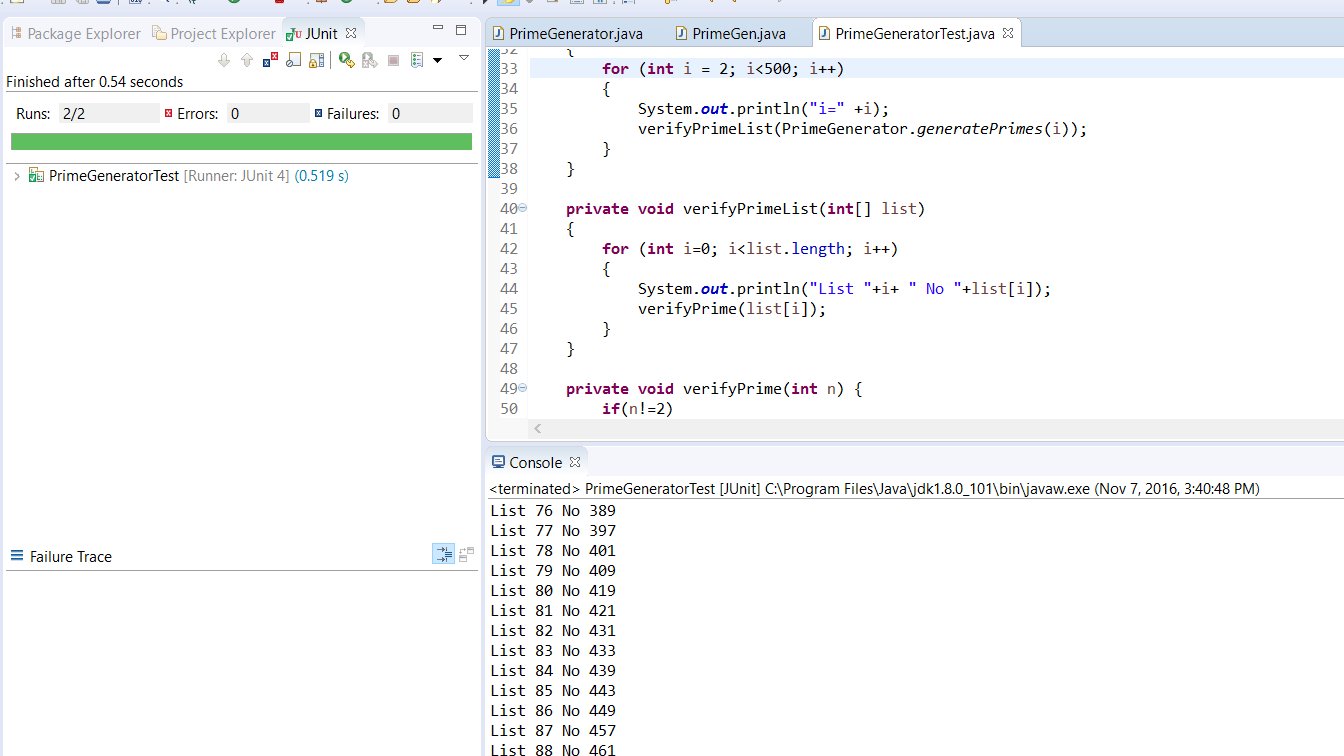
}

**After Fixing Bug**

**For Range <5**

****

**For range <500**



\*\*\*