SOEN 6011 - Software Engineering Processes

Summer 2019

Scientific Calculator- ETERNITY: FUNCTIONS

Project Report

Source Code Review of Eternity Function F1: arccos(x) developed by Himansi Patel

Instructor: PANKAJ KAMTHAN

By
Prashanthi Ramesh

August 3, 2019

Contents

Chapter 1

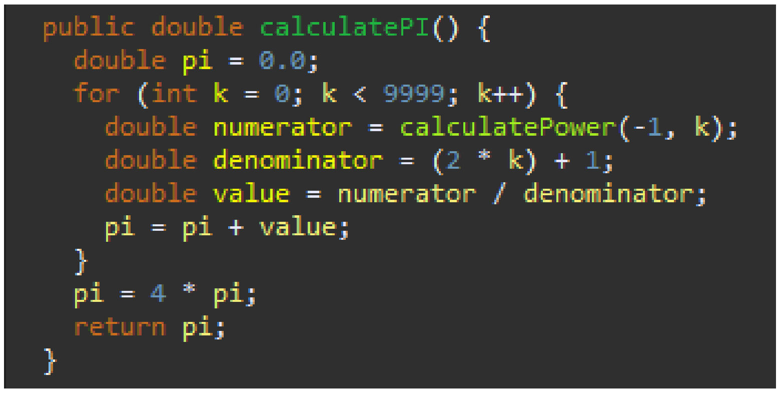
Source Code Review

Eternity Function F1: arccos(x) developed by Himansi Patel

1.1 Manual Code Review

Use of Static Methods

Non-static methods defined in classes HelperFunction and ArccosFunction - calculatePI(), calculate-Factorial(int number), calculatePower(double base, int exponent), arccos(double num) can be declared static.



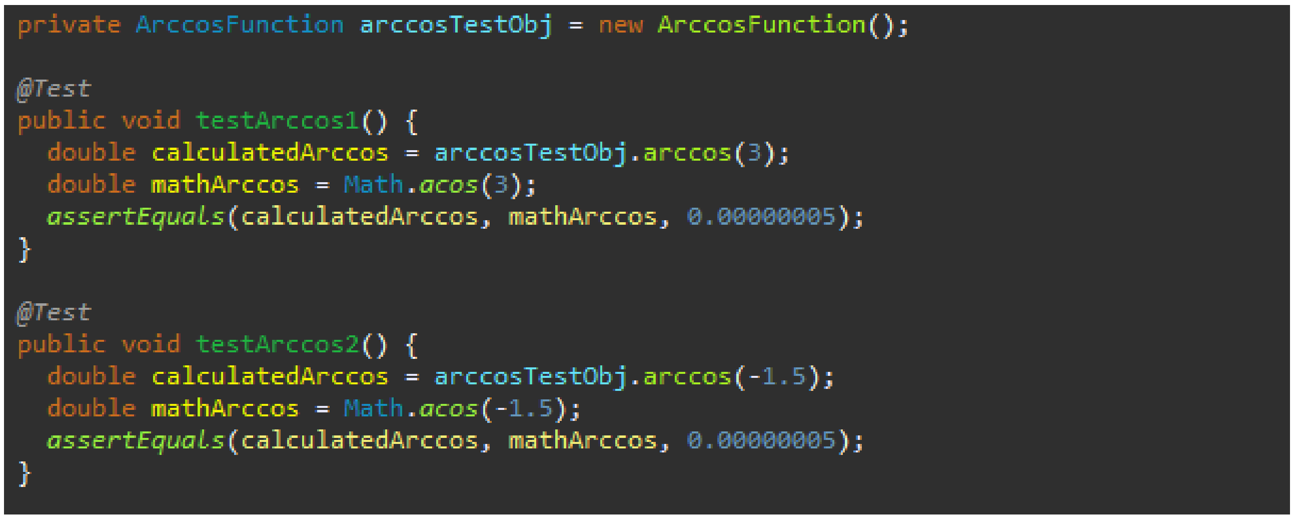
```
public double calculatePI() {
  double pi = 0.0;
  for (int k = 0; k < 9999; k++) {
    double numerator = calculatePower(-1, k);
    double denominator = (2 * k) + 1;
    double value = numerator / denominator;
    pi = pi + value;
  }
  pi = 4 * pi;
  return pi;
}
```

Figure 1.1: A non-static method in HelperFunction class that can be declared as static

- Methods are declared static as they do not hold the state of an object of the class

Missing Javadoc Comments

Unit test methods in classes ArccosTest and HelperFunctionTest can include javadoc comments to quickly understand the purpose of the tests.

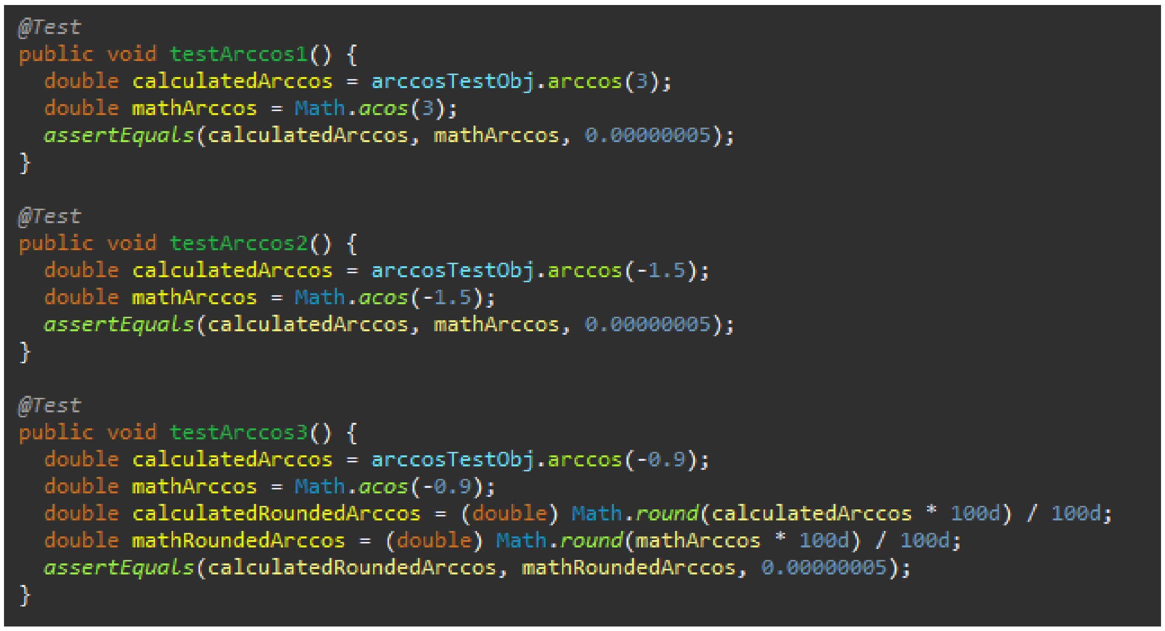


```java
private ArccosFunction arccosTestObj = new ArccosFunction();

@Test
public void testArccos1() {
  double calculatedArccos = arccosTestObj.arccos(3);
  double mathArccos = Math.acos(3);
  assertEquals(calculatedArccos, mathArccos, 0.00000005);
}

@Test
public void testArccos2() {
  double calculatedArccos = arccosTestObj.arccos(-1.5);
  double mathArccos = Math.acos(-1.5);
  assertEquals(calculatedArccos, mathArccos, 0.00000005);
}
```

Figure 1.2: Missing comments in Unit Test Methods

Naming Conventions

Unit test methods in classes ArccosTest and HelperFunctionTest can include self-descriptive method names to quickly understand the purpose of the tests.



```java
@Test
public void testArccos1() {
  double calculatedArccos = arccosTestObj.arccos(3);
  double mathArccos = Math.acos(3);
  assertEquals(calculatedArccos, mathArccos, 0.00000005);
}

@Test
public void testArccos2() {
  double calculatedArccos = arccosTestObj.arccos(-1.5);
  double mathArccos = Math.acos(-1.5);
  assertEquals(calculatedArccos, mathArccos, 0.00000005);
}

@Test
public void testArccos3() {
  double calculatedArccos = arccosTestObj.arccos(-0.9);
  double mathArccos = Math.acos(-0.9);
  double calculatedRoundedArccos = (double) Math.round(calculatedArccos * 100d) / 100d;
  double mathRoundedArccos = (double) Math.round(mathArccos * 100d) / 100d;
  assertEquals(calculatedRoundedArccos, mathRoundedArccos, 0.00000005);
}
```

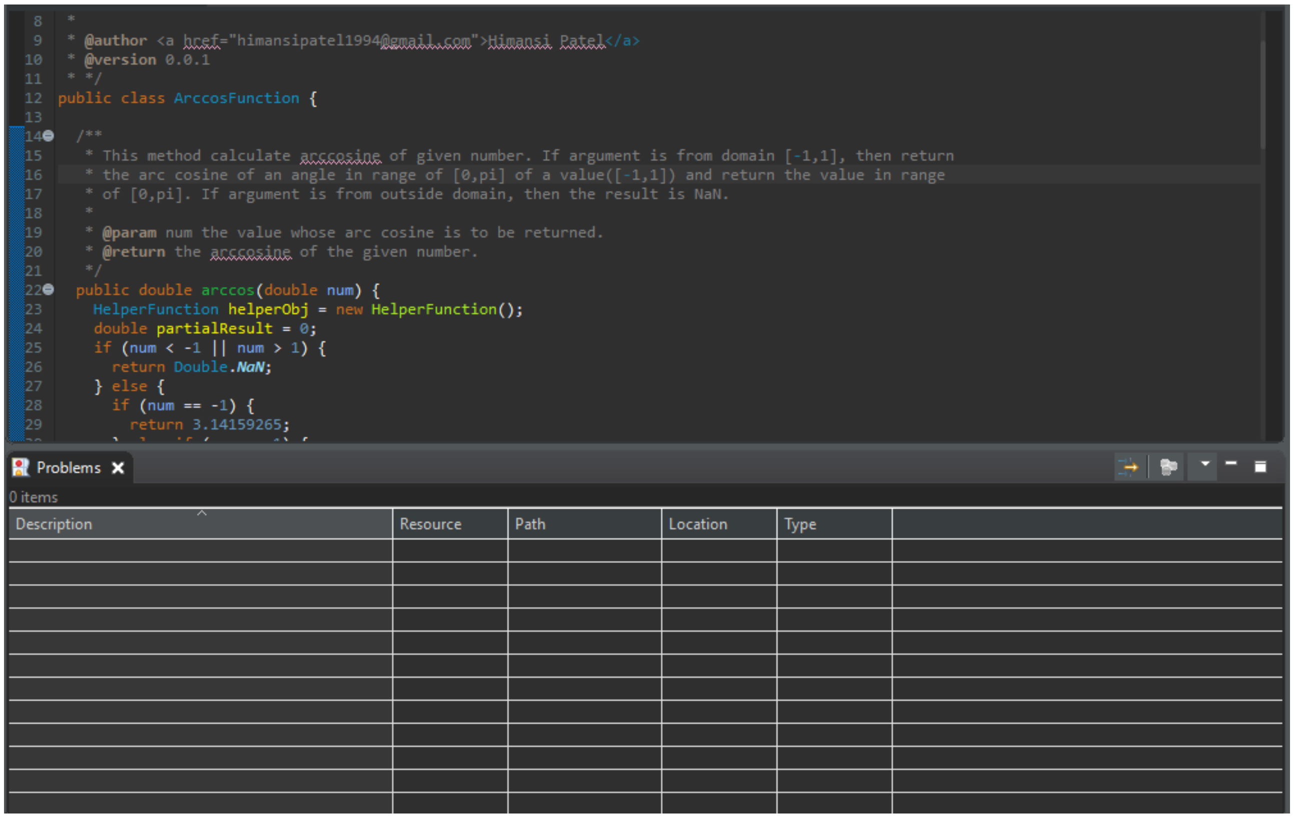
Figure 1.3: Method names that can be changed to self-descriptive names

1.2 Automatic Code Review

- For automatic source code review CheckStyle plugin for eclipse IDE was used.
- The Google Checks configuration was used to check if there are any deviations from the defined set of coding rules.
- The analysis showed no violations in the source code.

CheckStyle Plugin

Checkstyle inspects your Java source code and pointing out items that deviate from a defined set of coding rules.



```
 *
 * @author <a href="himansipatel1994@gmail.com">Himansi Patel</a>
 * @version 0.0.1
 * */
public class ArccosFunction {

  /**
   * This method calculate arccosine of given number. If argument is from domain [-1,1], then return
   * the arc cosine of an angle in range of [0,pi] of a value([-1,1]) and return the value in range
   * of [0,pi]. If argument is from outside domain, then the result is NaN.
   *
   * @param num the value whose arc cosine is to be returned.
   * @return the arccosine of the given number.
   */
  public double arccos(double num) {
    HelperFunction helperObj = new HelperFunction();
    double partialResult = 0;
    if (num < -1 || num > 1) {
      return Double.NaN;
    } else {
      if (num == -1) {
        return 3.14159265;
```

Problems

0 items

Description	Resource	Path	Location	Type

Figure 1.4: No violations generated by CheckStyle Plugin in Eclipse