CSDS 132 Programming Project III - Report
Alvisa Krasniqi

The Variable Class testing

1. Method testValueWithInput:
   I use the input value 14.0 to test the method and see if the outcome matches.
   The expected behavior was that the same input value should be returned by the method this did indeed happen.

2. Method testValueWithoutInput:
   When you run the method without any input, an UnsupportedOperationException should occur. Expected behavior is that the UnsupportedOperationException should be thrown by the method.

3. Method testDerivative:
   This verifies whether the derivative method gives a value of 1. for the variable.

4. Method testToString:
   This method in JUnit tests the toString method and checks if it returns the expected string representation.

5. Method testEquals:
   Test the equals method with two instances of Variable for comparison. Variables with the same class and no unique properties should be equal.

The Number Class testing

1. Method testValueWithInput:
   Test for the value method with a double input. Create a Number instance with a specific value. Test the method with input value another value and check if the result matches the initial value. The new value of the input should not affect the result of the method

2. Method testValueWithoutInput:
   Test the method without any input and expect it to return the previous stored value that we create when we do a number instance with a specific value.

3. Method testDerivative:
   Tests by creating a Number instance with a specific value and test that the derivative method returns a constant value of 0.

4. Method testToString:

Test by creating a Number instance and this method should return the string representation of the number. Check if string representation is as expected.

5. Method testEquals:
   Tests by creating two instances of Variable for comparison, and if the two instances are equal, then the method should return true.

The BinaryOp Class testing

1. Method testValueAddition
   Tests by creating a BinaryOp instance with PLUS operation, a left operand and right operand. The expected result is going to be the addition of the left and right operands.

2. Method testValueSubtraction
   Tests by creating a BinaryOp instance with SUB operation, a left operand and right operand. The expected result is going to be the subtraction of the left and right operands.

3. Method testValueMultiplication
   Test by creating a BinaryOp instance with MULT operation, a left and right operand which is an instance of Function. We expect to see the left and right operands to be multiplied together for the result.

4. Method testValueDivision
   Test by creating a BinaryOp instance with DIV operation, a left and right operand which is an instance of Function. We expect to see the left to be divided by the right operand together for the result.

5. Method testValueDivisionByZero
   Same process as the method testValueDivision above, besides the right operand is now zero so that we test division by zero.

6. Method testDerivativePlus
   Test by creating a BinaryOp instance with PLUS operation, a left operand and right operand.

7. Method testValueSubtraction
   Test by creating a BinaryOp instance with SUB operation, left operand and right operand.

8. Method testValueMultiplication

Test by creating a BinaryOp instance with MULT operation, left operand as a variable, and right operand.

9. Method testValueDivision
   Create a BinaryOp instance with DIV operation, left operand, and right operand as a variable.

10. Method testToString
    Test by creating a BinaryOp instance with SUB operation, left operand as a variable, and right operand. Check if string representation is as expected.

11. Method testEquals
    Test by creating two BinaryOp instances with the same operation and operands and expect that they return the same value.

The Polynomial Class testing

1. Method testGetOperand

Test by creating a Polynomial instance with a specific operand and specific degree, get the operand using getOperand method. Assert that the result is not null, and has the specific value of the instance Number.

2. Method testValueWithoutInput (throws UnsupportedOperationException).

Testing this by creating a Polynomial instance with an operand and degree, and when we call the value we expect an UnsupportedOperationException.

3. Method testValue:

Testing by creating a Polynomial instance with an operand of 4 and we expect that for example if we call a value(2) method and should get the result to be 16.0 (2^4).

4. Method testDerivative

Test by creating a testOperand and polynomial instance with an operand and degree.

5. Method testToString

Test by creating a Polynomial instance with an operand and degree, and check whether we have the correct.

6. Method testEquals()
   Test by creating two polynomial instances that have the same operand and degree, which we expect to be equal.

The Log Class testing

1. Method testValue
   Test by is creating a log object with the operand. Use Math.log(operand) to assert that the result matches.

2. Method testDerivative
   Test by using a variable operand to create an instance of Log. Determine the logarithm function's derivative and assert that the derivative result is an instance of BinaryOp and that it is not null.

3. Method testToString
   Test by creating a Log instance with a variable operand. Check if string representation is as expected.

4. Method testEquals
   Test by creating two Log instances with the same variable operand and expect that they will be equal.

The Exp Class testing

1. Method testValue
   Test by creating a new variable operand and an exponential function with the operand as a variable.

2. Method testDerivative
   Test by creating a new variable operand and then create an exponential function with the operand as a variable. Afterwards, calculate the derivative of the exponential function.

3. Method testToString
   Test by creating a new variable operand and then create an exponential function with the variable operand.

4. Method testEquals
   Test by creating two variable operands and create two exponential functions with the same variable operands which should be equal.

The Sin Class testing

1. Method testValue
   Test by creating a variable operand, a sine function with the variable operand and calculate the value of the sine function at certain input.

2. Method testDerivative
   Test by creating a variable operand, a sine function with the variable operand and calculate the derivative of the sine function.

3. Method testToString
   Test by creating a variable operand, and a sine function with the variable operand. Check if the string representation of the sine function is as expected.

4. Method testEquals
   Test by creating two variable operands and two sine functions with the same variable operands. Test if the two sine functions are equal.

The Cos Class testing

1. Method testValue
   Test by creating a variable operand and a cosine function with the variable operand. Calculate the value of the cosine function at a certain input.

2. Method testDerivative

Test the derivative method of Cos by creating a variable operand and a cosine function with the variable operand. Then, calculate the derivative of the cosine function for various input values

3. Method testToString

Test by creating a variable operand and a cosine function with the variable operand. Afterwards, check if the string representation of the cosine function is as expected

4. Method testEquals

Test equality of the two cosine functions same variable operands