CONCORDIA UNIVERSITY
DEPARTMENT OF COMPUTER SCIENCE AND SOFTWARE ENGINEERING
SOEN 6011: SOFTWARE ENGINEERING PROCESSES
SECTION CC WINTER 2019
F1: $arccos(x)$

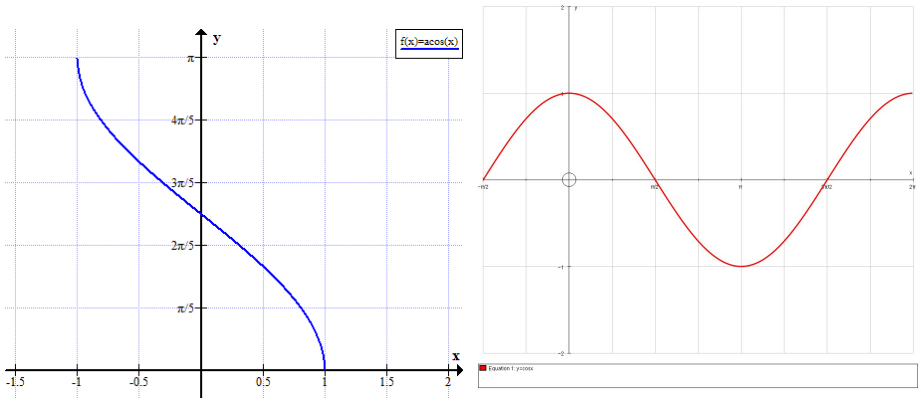
Changes from D_2 to D_3: Section 5 & 6

STUDENT NAME: YONGCONG LEI
STUDENT IDENTIFICATION NUMBER: 40045701
GITHUB REPOSITORY: https://github.com/WingCuengRay/SOEN6011_Summer19

1 Characteristics and Domain

1.1 Characteristics

1. $arccos(x)$ is an inverse trigonometric function (relate an angle of a right-angled triangle to ratios of two side lengths), and it's tightly related to the trigonometric cosine function.



(a) $arccos(x)$ (b) $cos(x)$

Figure 1: $arccos(x)$ and $cos(x)$

2. $arccos(x)$ is an inverse trigonometric functions of $cos(x)$. In order introduce inverse trigonometric functions, we first need to take a look at trigonometric functions. In mathematics, the trigonometric functions are real functions which relate an angle of a right-angled triangle to ratios of two side lengths[**?**]. $cos(x)$ is one of them. Given

$$x = cos(y) = \frac{b}{h} = \frac{adjacent}{hypotenuse}$$

then

$$y = arccos(x)$$

Figure 1 is an example of $cos(x)$. In this case, $x = cos(y) = \frac{OC}{OA}$. Therefore, $y = arccos(x) = arccos(\frac{OC}{OA})$.

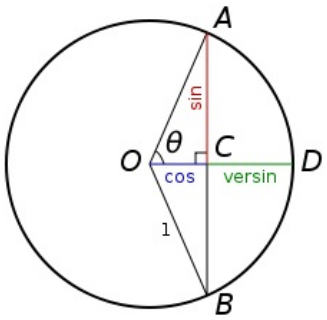


Figure 2: trigonometric functions

3. The function *arccos* has their **principle values**. It's because inverse cosine function are not one-to-one.Therefore, the ranges of the inverse cosine functions are proper subsets of the domains of the original functions. With this restriction, for each x in the domain the expression *arccos* will evaluate only to a single value, called its principal value. The range of principle values of $arccos(x)$ is $[-\frac{1}{2}\pi, \frac{1}{2}\pi]$. As a result, there will be one-to-one relationship for any value in domain.

1.2 Domain and Co-domain

Since none of the six trigonometric functions are one-to-one, they are restricted in order to have inverse functions. The co-domain of $arccos(x)$ in area $[-\infty, \infty]$. However, we usually takes the principle values as its co-domain which is $[-\frac{1}{2}\pi, \frac{1}{2}\pi]$. Therefore, there will be one-to-one relationship between its domain and co-domain.

As a result, the domain is $x \in [-1, 1]$. The co-domain is $y \in [-\frac{1}{2}\pi, \frac{1}{2}\pi]$ for the equation $y = arccos(x)$.

2 Requirements

1. When a user apply a value $x \in [-1, 1]$ (which is the domain) to function $arccos(x)$, the calculator shall return a value $y \in [0, \pi]$.
2. When a user apply a value $x \notin [-1, 1]$, the calculator shall show an error message.
3. If the result is a infinite decimal, the result shall be shown to the nearest five decimal points.
4. The calculator shall support input of operands such as digits or special irrational number such as π, e etc.
5. When a user enter anything to the calculator, the calculator shall show anything that user input.
6. When a user input an invalid sequence of operands and operators, a error message shall be shown.
7. When a user reset the calculator, all the ongoing transaction and history should be erased.
8. The user shall be able to correct or modify his input to the calculator.
9. The calculator shall be able to save the input sequence and related result to its memory.

3 Algorithms

1. The algorithm uses the Lagrange Polynomial Approximation to calculate the arccos(x). The advantage of the algorithm is its performance. The time complexity is constant time. The advantage is that it has a maximum error of about 0.18 rad. Besides, The polynomial approximation performs pretty bad near $x = -1$ or $x = 1$ where the derivative of the inverse cosine goes to infinity. Here is the pseudocode of the algorithhm.

   LAGRANGE-POLYNOMIAL-APPROXIMATION-ARCCOS(x)
   1: $x\_square = x * x$
   2: $v1 = -0.69813170079773212 * x\_square - 0.87266462599716477$
   3: return $v1 * x + 1.5707963267948966$

2. Besides, there is an algorithm for $arccos(x)$ based on rational function - the quotient of two polynomials. The advantage is that it can give a a much better approximation. It has a maximum absolute error of 0.017 radians (0.96 degrees) on the interval (-1, 1). The disadvantage of the algorithm is that it involves division operator. In some cases, division is more expensive than addition and multiplication. Here is the pseudocode of the algorithm.

   RATIONAL-FUNCTION-APPROXIMATION-ARCCOS(x)
   1: $a = -0.939115566365855$
   2: $b = 0.9217841528914573$
   3: $c = -1.2845906244690837$
   4: $d = 0.295624144969963174$
   5: $x\_square = x * x$
   6: $x\_cube = x\_square * x$
   7: $x\_quad = x\_cube * x$;
   8: return $(\pi/2 + (a * x + b * x\_cube)/(1 + c * x\_square + d * x\_quad))$

4 Problem 4

4.1 Debugger

4.1.1 Description

I choose the debugger tools in IntelliJ IDEA as the debugger of my application. It provides a full range of facilities of debugging in source code, including but no limitting:

1. Breakpoints in Java.
2. Customizable breakpoint properties: conditions, pass count, and so on.
3. Frames, variables, and watches views in the debugger UI.
4. Runtime evaluation of expressions.

4.1.2 Advantage

There are lots of **advantages** with the debugging tool set in IntelliJ IDEA.

1. It provides a simple but powerful user interface for debugging. User can easy create various kinds of breakpoints, such as conditional breakpoints, temporary breakpoints, temporary disabled breakpoints and so on.
2. It provides the functionality of inline values view. It lets you view the values of variables used in your source code right next to their usage
3. It provides the functionality of hot swapping which allows user to insert minor changes in your code without shutting down the process.
4. It provides the functionality of remote debugging. Remote debug means attaching debugger to a process which is already running on a specific port on your or any other's host. This way you can attach the debugger to your application server which is running standalone.

4.1.3 Disadvantage

The biggest **disadvantages** of IntelliJ debugger is because of it's native in IntelliJ IDEA, which it can't be ran as an standalone debugger without IntelliJ IDEA. As a result, you must install the IDE to debug the application and the IDEA is kind of heavy.

4.2 Checkstyle

4.2.1 Description

I choose Checkstyle wtih Google Java Style as the tool to check the quality of your source code. Checkstyle is a development tool to help programmers write Java code that adheres to a coding standard. It automates the process of checking Java code to spare humans of this boring (but important) task. This makes it ideal for projects that want to enforce a coding standard.Checkstyle is highly configurable and can be made to support almost any coding standard. Google Java Style is only of the standard.

4.2.2 Advantage

1. It makes your source control diffs show just actual code changes, and all but eliminates ”diff noise” due to whitespaces and other insignificant formatting choices
2. It makes all code more similar, so that developers are more comfortable pairing and sharing the code bases.
3. It greatly helps to uniform the code in the company, and thanks to that you generally produce a more easily understandable and way more easily maintainable structure for products.

4.2.3 Disadvantage

I would say there is hardly an disadvantage of Checksyle. The only possible disadvantages is the lack of a human eye after the formatting, to add some exceptions, etc.

4.3 Effort made to achieving attributes

- correctness and robust - I wrote test case with JUnit to ensure there is no bug in each method of my applicaiton and reached a test report with high-coverage .
- efficiency - I did some research about the implementation of my function and made a comparison between mutiple implementation. Then I chose the one with lowest time complexity.
- usability - I provided textual user interface for my calculator so that users can user it easily.
- maintainable - I added comments and documentation in the code, such as Javadoc.

5 Problem 5 - Source Code Review

The function which I review is F2 $tan(x)$ by Qing Li. I choose the native review tool in GitHub to generate the source code review report automatically. It's available when you make a pull request. The reviewer can make comments on the code and either approve or reject the pull request.

5.1 Code Review Practices

- check if application can run successfully
- check the correctness and accuracy of her function
- check if formatting is consistent with Google CheckStyle
- check if unnecessary white-space / unused import don't exist in source code
- check if variables are named properly
- check if no magic numbers or hard-coded Strings are used .(Constants or Enums should be used instead)
- check if code is well-documented

5.2 Code Review Result

Qing's source code is clean and well-documented. Each method is documented in Javadoc style. There is no error in the code and the application can be ran successfully. In spite of these, some small improvements in my opinion. The report below will show the details of them.

Review Practices	Status(Pass/Fail)
application can run	Pass
the correctness and accuracy of her function	Pass
formatting is consistent with Google CheckStyle	Pass
unnecessary white-space / unused import don't exist in source code	Pass
variables are named properly	Pass
no magic numbers or hard-coded Strings are used	Pass
code is well-documented	Pass

5.3 Checkstyle report

Qing maintained a good style of code based on Google CheckStyle. There is only one on warning in her source files.

```
file: TangentFunction.java
Warning: '\}' at column 29 should be alone on a line. (13:29) [RightCurly]
```

5.4 Code Review Report

WingCuengRay / SOEN6011_Summer19

code by Qing #1

Edit

Open **WingCuengRay** wants to merge 1 commit into `master` from `source_review`

Conversation 7 Commits 1 Checks 0 Files changed 3 +231 −0

Changes from all commits File filter... Jump to...

114 source code review/CalculatorPanel.java

```
@@ -0,0 +1,114 @@
+ package tangentfunction;
+
+ import java.awt.BorderLayout;
+ import java.awt.Color;
+ import java.awt.Font;
+ import java.awt.GridLayout;
+ import java.awt.event.ActionEvent;
+ import java.awt.event.ActionListener;
+ import javax.swing.BorderFactory;
+ import javax.swing.JButton;
+ import javax.swing.JFrame;
+ import javax.swing.JPanel;
+ import javax.swing.JTextField;
+
+ /**
+  * User interface of the tangent calculator.
+  *
+  * @author Qing Li SID:40082701
+  */
+ public class CalculatorPanel extends JFrame {
+
+   static JFrame frame = new JFrame("TangentCalculator");
+   JTextField display = new JTextField("");
+   Font font = new Font("Consolas", Font.BOLD, 40);
+   Font font1 = new Font("Consolas", Font.BOLD, 20);
+   Actionlistener al = new Actionlistener(); // create an object of action listener
+
+   // calculator buttons
+   JButton buttonClear = new JButton("C"); // button "clear"
+   JButton buttonTan = new JButton("tan"); // button "tan"
+
+   /**
+    * Constructor.
+    */
+   public CalculatorPanel() {
+     super();
+
+     // set button's color
+     buttonClear.setBackground(Color.white);
+     buttonTan.setBackground(Color.pink);
+
+     // set button's font
+     buttonClear.setFont(font1);
+     buttonTan.setFont(font1);
+
+     // set background
+     display.setHorizontalAlignment(JTextField.RIGHT);
```

```
48  +     display.setFont(font);
49  +     display.setBackground(Color.white);
50  +     display.setEditable(true);
51  +
52  +     // set input textbox
53  +     JPanel pan1 = new JPanel();
54  +     pan1.setLayout(new GridLayout(1, 1, 5, 5));
55  +     pan1.setBorder(BorderFactory.createEmptyBorder(5, 5, 0, 5));
56  +     pan1.add(display);
57  +
58  +     // add buttons into the whole panel
59  +     JPanel panBody = new JPanel();
60  +     panBody.setLayout(new GridLayout(1, 2));
61  +     panBody.add(buttonTan);
62  +     panBody.add(buttonClear);
63  +     panBody.setBorder(BorderFactory.createEmptyBorder(0, 5, 5, 5));
64  +
65  +     // set layout
66  +     frame.setLayout(new BorderLayout());
67  +     frame.add(pan1, BorderLayout.NORTH);
68  +     frame.add(panBody, BorderLayout.CENTER);
69  +     frame.setSize(780, 180);
70  +     frame.setLocation(260, 150);
71  +     frame.setVisible(true);
72  +     frame.setResizable(false);
73  +     frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
74  +
75  +     // give each button an action listener
76  +     buttonClear.addActionListener(al);
77  +     buttonTan.addActionListener(al);
78  +   }
79  +
80  +   /**
81  +    * Action Listener.
82  +    */
83  +   public class Actionlistener implements ActionListener {
84  +
85  +     TangentFunction tan = new TangentFunction();
```

WingCuengRay 23 hours ago Author Owner

This object can be placed in class-level. It can be reused in each action.

Reply...

Resolve conversation

```
86  +     public String preDisplay = "";
87  +
88  +     /**
89  +      * Action controller.
90  +      */
91  +     @Override
92  +     public void actionPerformed(ActionEvent e) {
93  +       Object click1 = e.getSource(); // get the button's name
94  +       if (click1 == buttonClear) {
95  +         preDisplay = "";
96  +         display.setText(""); // clear everything
97  +       } else if (click1 == buttonTan) {
98  +
99  +         String result = tan.tangent(display.getText()); // get the calculated result
100 +
101 +         if (result.equals("Error: Empty input!")) {
```

WingCuengRay 23 hours ago Author Owner

It's bettern not to compare the error message string. A regex, `"Error:.*+"` can be used to match the error message.

Reply...

Resolve conversation

```
+            display.setText("Error: Empty input!"); // error report
+          } else if (result.equals("Error: This is not a real number!")) {
+            display.setText("Error: This is not a real number!"); // error report
+          } else if (result.equals("Error: The value is not existing!")) {
+            display.setText("Error: The value is not existing!"); // error report
+          } else {
+            display.setText(String.valueOf(result));
+          }
+          preDisplay = "";
+        }
+      }
+    }
+ }
```

7 source code review/Run.java

```
@@ -0,0 +1,7 @@
+ package tangentfunction;
+
+ public class Run {
+   public static void main(String[] args) {
+     CalculatorPanel cp = new CalculatorPanel();
+   }
+ }
```

110 source code review/TangentFunction.java

```
@@ -0,0 +1,110 @@
+ package tangentfunction;
+
+ /**
+  * F2_tan(x).
+  *
+  * @author Qing Li SID:40082701
+  */
+ public class TangentFunction {
+
+   /**
+    * Constructor.
+    */
+   public TangentFunction() {}
+
+   /**
+    * Power function.
+    *
+    * @param x is the number of base
+    * @param n is the number of power
+    * @return the result represented in double
+    */
+   public static double pwr(double x, int n) {
```

WingCuengRay 23 hours ago Author Owner

This method should not be exposed. It can be a private non-static method.

Reply...

Resolve conversation

```
+     int i = 0;
+     double powers = 1;
+     if (n == 0) {
+       return 1;
+     }
+     for (i = 1; i <= n; i++) {
+       powers = powers * x;
+     }
+     return powers;
+   }
+
+   /**
+    * Factorial function.
+    *
+    * @param n is an integer number
+    * @return the result represented in double
+    */
+   public static double fac(int n) {
```

WingCuengRay 23 hours ago Author Owner

This method should not be exposed. It can be a private non-static method.

Reply...

Resolve conversation

```
+     int i = 0;
+     double pdt = 1;
+     if (n == 0 || n == 1) {
+       return 1;
+     }
+     for (i = 2; i <= n; i++) {
+       pdt = pdt * i;
+     }
+     return pdt;
+   }
+
+   /**
+    * Tangent function.
+    *
+    * @param line is the input stream
+    * @return the result represented in String
+    */
+   public String tangent(String line) {
+     int i = 0;
+     int j = 0;
+     int cons = 1;
+     double s = 0;
+     double c = 0;
+     double sin = 0;
+     double cos = 0;
+     double y = 0;
+     final double pi = 3.1415926;
```

WingCuengRay 23 hours ago Author Owner

PI can be a final static class member

Reply...

Resolve conversation

```
68  +     String msg;
69  +
70  +     // error handling: to detect the empty input
71  +     if (line.isEmpty()) {
72  +       msg = "Error: Empty input!";
73  +       return msg;
74  +     }
75  +
76  +     // error handling: to detect non real number input
77  +     for (int n = 0; n < line.length(); n++) {
78  +       if (line.charAt(n) < 48 || line.charAt(n) > 57) {
```

WingCuengRay 23 hours ago Author Owner

`line.charAt(n) < '0' || line.charAt(n) > '9'` can be used for bettern readability.

Reply...

Resolve conversation

```
79  +         if (line.charAt(n) != 45 && line.charAt(n) != 46) {
80  +           msg = "Error: This is not a real number!";
81  +           return msg;
82  +         }
83  +       }
84  +     }
85  +
86  +     y = Double.valueOf(line);
87  +
88  +     // error handling: to detect non exist value
89  +     if (y % 90 == 0) {
90  +       msg = "Error: The value is not existing!";
91  +       return msg;
92  +     }
93  +
94  +     double x = y * pi / 180;
95  +
96  +     for (int k = 1; k <= 24; k++) {
97  +       i = 2 * k - 1;
98  +       j = 2 * k - 2;
99  +       s = cons * pwr(x, i) / fac(i); // calculate sin based on Taylor series
100 +       c = cons * pwr(x, j) / fac(j); // calculate cos based on Taylor series
101 +       cons = -1 * cons;
102 +       sin = sin + s;
103 +       cos = cos + c;
104 +     }
105 +
106 +     double tan = sin / cos; // tangent function based tan=sin/cos
107 +     msg = "tan(" + (float) y + ")= " + String.format("%.6f", tan);
108 +     return msg;
109 +   }
110 + }
```

ProTip! Use `n` and `p` to navigate between commits in a pull request.

6 Problem 6 - Test Case Report

The test case I need to review is F3: $sinh(x)$ by Xueying Li. Here is the testing environment on which I ran the tests. I ran the tests with test suit in IntelliJ IDEA which support various kind of funtionalitiy, including debugging, test report generation, etc.

- Operating System: Windows 10 OS
- Memory: 8G
- Disk: 256G SSD
- JDK version: Oracle JDK 1.8.0_161
- IDE: IntelliJ IDEA 2018.02.05 (ULtimate Edition)

There are three classes in her source code and one test class in test test. Here is the test coverage.

Class	Class,%	Method,%	Line,%
CalculatorController	100% (1/ 1)	90.9% (10/ 11)	65.3% (111/ 170)
CalculatorModel	100% (1/ 1)	80% (4/ 5)	85.7% (6/ 7)
CalculatorView	100% (1/ 1)	50% (2/ 4)	6.2% (2/ 32)

Here is the detail of each test case and the related requirement.

Test Case ID	Description	Module Name	Requirement ID	Test Result
TFR1	Testing executation application	CalculatorController	FR1	Passed
TFR2	Testing user input	CalculatorController	FR2	Passed
TFR3	Testing result output	CalculatorController	FR3	Passed
TFR4	Testing input validating	CalculatorController	FR4	Passed
TQR1	Testing accuracy	ALL	QR1	Passed
TQR2	Testing testability	CalculatorController	QR2	Passed
TQR3	Testing system reliability	CalculatorController	QR3	Passed

References

[1] E. P. Dolzhenko *A comparison of rates of rational and polynomial approximation*. Mathematical notes of the Academy of Sciences of the USSR, March 1967, Volume 1, Issue 3, pp 208–212

[2] Juan L. Varona *Rational values of the arccosine function*. Central European Journal of Mathematics, June 2006, Volume 4, Issue 2, pp 319–322