https://github.com/yanzhen940608/YANZHENLISOEN6011

Problem 4

1 Changes from D1 to D2

1.1 github address

Add the github address on the top of the document

2 Debugger

I used debugger which is embedded in eclipse.

Eclipse allows us to start a Java program in Debug mode.It provides a Debug perspective which gives us a pre-configured set of views. Eclipse allows us to control the execution flow via debug commands.

2.1 Advantages

It can Setting Breakpoints:
A breakpoint is a signal that tells the debugger to temporarily suspend execution of your program at a certain point in the code.To define a breakpoint in your source code, right-click in the left margin in the Java editor and select Toggle Breakpoint. The Breakpoints view allows us to delete and deactivate Breakpoints and modify their properties.All breakpoints can be enabled/disabled using Skip All Breakpoints.Breakpoints can also be imported/exported to and from a workspace.

It offers Debug Perspective:
The debug perspective offers additional views that can be used to troubleshoot an application like Breakpoints, Variables, Debug, Console etc. When a Java program is started in the debug mode, we are prompted to switch to the debug perspective.

Debug view —— Visualizes call stack and provides operations on that.

Breakpoints view —— Shows all the breakpoints.

Variables/Expression view —— Shows the declared variables and their values. We can add a permanent watch on an expression/variable that will then be shown in the Expressions view when debugging is on.

Display view —— Allows to Inspect the value of a variable, expression or selected text during debugging.

Console view —– Program output is shown here.

2.2 Disadvantages

Among the advantages mentioned above, there is a function called "Setting Breakpoints", but the operation itself is limited and not ideal. It can backtrack and call the function again, but the changed variable value cannot be restored. Therefore, calling the function again can see the change, but not the actual value of the first execution.

3 Checkstyle

I used tool called Checkstyle, checked the quality of my source code.

Checkstyle is an open source project of SourceForge. The main purpose of it is to check whether the Java source file conforms to the code specification. By checking code coding format, naming conventions, Javadoc, class design and other aspects of code specification and style checks, it effectively constrains developers to better comply with code coding specifications.

3.1 Advantages

Checkstyle provides plug-ins that support most common IDEs. I mainly use the Checkstyle plug-in in Eclipse. Checkstyle checks the coding style of the code and displays the results in the Problems view. Each magnifying glass icon in the code editor represents a code defect found by Checkstyle. We can view error or warning details in the Problems view.

In addition, Checkstyle supports users to customize code checking specifications according to their needs. In the configuration panel, we can add or delete custom checking specifications on the basis of existing checking specifications such as naming conventions, Javadoc, block, class design, etc.

3.2 Disadvantages

The disadvantages of Checkstyle are mainly concentrated in three aspects :
1. Too strict in accordance with Sun's specifications, we need to customize the rules
2. Plug-in custom rules have no lookup function, and it can be very troublesome to find rules.
3. Can only check the code, can not modify the code, if you want to modify, can only cooperate with Jalopy to use together.

4 Quality Attributes

4.1 Correctness

Considered all possible range of value for both base and x value for example, considering the cases where base=1, base$\leq 0andX \leq 0$.

4.2 Efficiency

The algorithm uses Taylor expansion and that is why the speed of run time execution is fast.

4.3 Usability

Implemented the textual user interface for the function.It is easy to use on the console of eclipse.

4.4 Maintainability

Code has written by following the coding standards and it is very easy to understand the code.

4.5 Robustness

All possible states of unexpected termination and operation are considered, and code is implemented to handle such operations by displaying accurate and clear error messages, thus allowing users to debug programs more easily.

References

[1] Quality Attributes
Barbacci, M., Klein, M. H., Longstaff, T. A., Weinstock, C. B. (1995). Quality Attributes (No. CMU/SEI-95-TR-021). CARNEGIE-MELLON UNIV PITTSBURGH PA SOFTWARE ENGINEERING INST.

[2] Checkstyle
Zaidman, A., Van Rompaey, B., Demeyer, S., Van Deursen, A. (2008, April). Mining software repositories to study co-evolution of production test code. In 2008 1st international conference on software testing, verification, and validation (pp. 220-229). IEEE.

[3] Java Debugger
Alsallakh, B., Bodesinsky, P., Gruber, A., Miksch, S. (2012, March). Visual tracing for the eclipse java debugger. In 2012 16th European Conference on Software Maintenance and Reengineering (pp. 545-548). IEEE.