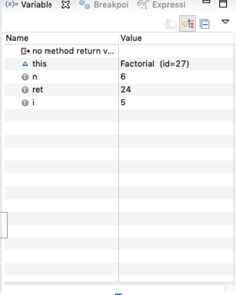
**Java Lab 07**

Sehyung (Eric) Kim Instructor: Keith Tang

A01089966

Page 9: Screen Capture One:

Show variables windows here below:

Page 9: Table One

**Question**: What is the Eclipse keyboard shortcut for toggling a breakpoint?

**Answer**: command + shift + b (macbook) or ctrl + shift + b (windows)

**Question**: What is the difference between “Step-Over”, and “Step-Into”, and “Step-Return”?

**Answer**: “Step Over [ F6 ] is used to execute individual lines of code. If you “Step Over” a method call then the method is run and the debugger continues on after the method call.

Step Into command [ F5 ] steps into the next method call at the currently executing line of code.

Step Return command [ F7] is used to return from a method which has been stepped into. Even though we return from the method, the remainder of the code inside the method will be executed normally.

**Question:** Based on your best understanding of the program, provide a list of methods that are called when the program executes (from start to end, in order of being called). You can skip library methods (like println, for example).

**Answer:**

DebugStar.main(String[]) line: 16

DebugStar.run(String, int, int) line: 32

DebugStar.getOperation(String) line: 22

DebugStar.getOperation(String) line: 23

Add.<init>() line: 47

DebugStar.getOperation(String) line: 23

DebugStar.getOperation(String) line: 24

DebugStar.getOperation(String) line: 27

DebugStar.run(String, int, int) line: 32

DebugStar.run(String, int, int) line: 33

Add.perform(int, int) line: 49

DebugStar.run(String, int, int) line: 33

DebugStar.run(String, int, int) line: 34

DebugStar.main(String[]) line:17

DebugStar.run(String, int, int) line: 32

DebugStar.getOperation(String) line: 22

DebugStar.getOperation(String) line: 25

Subtract.<init>() line: 51

DebugStar.getOperation(String) line: 25

DebugStar.getOperation(String) line: 27

DebugStar.run(String, int, int) line: 32

DebugStar.run(String, int, int) line: 33

Subtract.perform(int, int) line: 53

DebugStar.run(String, int, int) line: 33

DebugStar.run(String, int, int) line: 34

DebugStar.main(String[]) line: 18

DebugStar.run(int) line: 39

Factorial.<init>() line: 55

DebugStar.run(int) line: 39

DebugStar.run(int) line: 40

Factorial.perform(int) line: 58

Factorial.perform(int) line: 59

Factorial.perform(int) line: 60

Factorial.perform(int) line: 59

Factorial.perform(int) line: 60

Factorial.perform(int) line: 59

Factorial.perform(int) line: 60

Factorial.perform(int) line: 59

Factorial.perform(int) line: 60

Factorial.perform(int) line: 59

Factorial.perform(int) line: 60

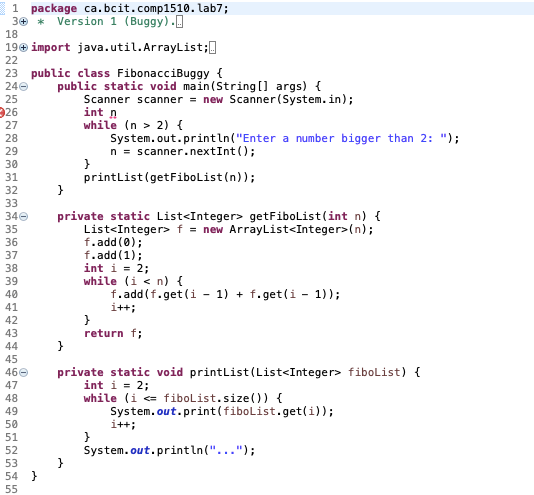
Factorial.perform(int) line: 59

Factorial.perform(int) line: 62

DebugStar.run(int) line: 40

DebugStar.run(int) line: 41

Page 9/10: Screen Capture: Original Code with line numbers:

Screen Capture of Original Code Here:

Page 10:

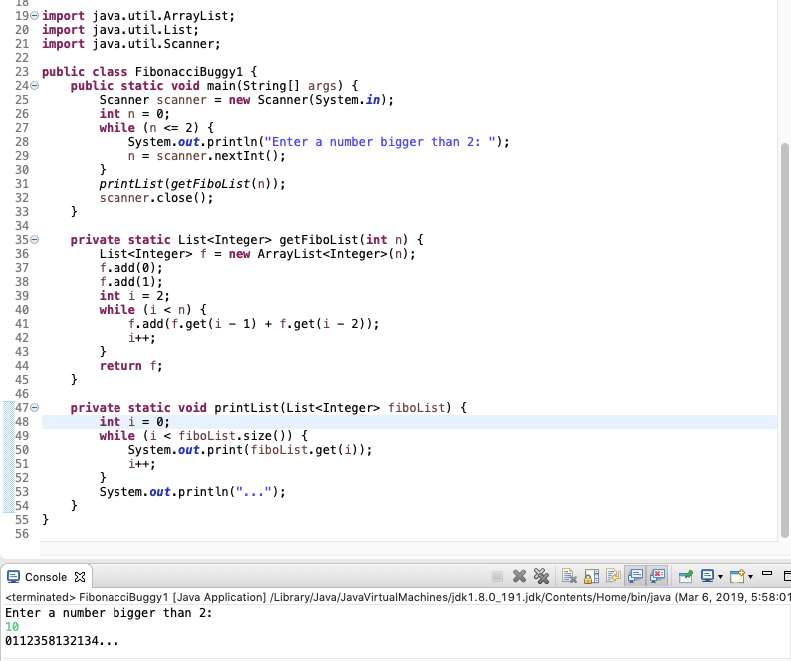
Question: Provide an error log table (such as the one below) indicating error details (line number,

type of error, and explain error and show correction).

|  |  |  |  |
| --- | --- | --- | --- |
| Line Number | Type of error (compile-time, run-time, or logical) | Description | Correction |
| 26 | compile-time | need to initialize n as 0 and close with semicolon | int n = 0; |
| 25 | run-time | resource leak, scanner needs to be closed | scaner.close(); |
| 27 | logical | n condition needs to be changed to fit what it’s asking for in the while loop, in this case will printList and end program as n = 0 and doesn’t fit in while loop with condition. | n <= 2; |
| 41 | logical | f.get(i - 1) because i - 2 twice is f.add(1) twice, but should be i-1 and i-2 for f.add(1) + f.add(0). | f.get(i - 2) |
| 48 | logical | the list size should have from 1 up, rather than from 2 and above. This would output the wrong fibonacci sequence number as a result. | int i = 0; |
| 49 | run-time | If i is equal to the fiboList.size(), then it won’t run because size array method of fiboList can’t be 0. | while ( i < fiboList.size()) |

Page 10

Question: Provide screen capture of fixed code (with line numbers), and sample run, using n=10:



Page 12

Question 8, 9, 10

