CSCI1530 Computer Principles and Java Programming

Tutorial 3

NetBeans Debugger

Zheng Qingqing
SHB 911
qqzheng@cse.cuhk.edu.hk

Content

- NetBeans Debugger
- Debug Assignment 1
- Debug exercises

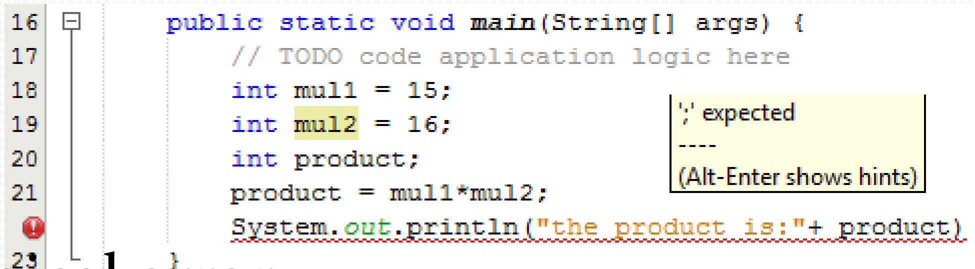
NetBeans Debugger

Run Project vs Debug Project

- Run Project
  - Execute your program until it terminates
  - Program may not quit if it contains bugs, like infinite loop
- Debug Project
  - Allow you to choose when to pause the program
  - Either let the program <u>runs to a line</u> or <u>execute a line at a time</u>
  - In a pause, you can <u>check variable values</u>

NetBeans debugger

- NetBeans Editor provides a comprehensive syntax checking
  - Not initialize a variable
  - Missing a ';'...



```
public static void main(String[] args) {
    // TODO code application logic here
    int mul1 = 15;
    int mul2 = 16;
    int product;
    product = mul1*mul2;
    System.out.println("the product is:"+ product)
}
```

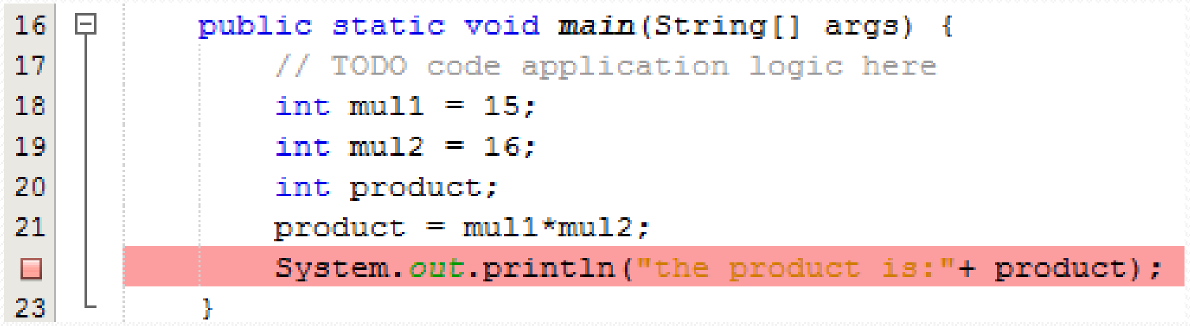
- But it cannot detect the logical error...
  - e.g. Infinite loop

```
for(int i=0; i<5; i--){ ... }
```

- Debugger can help to investigate in it
  - Use Breakpoint and Step!
  - Check the variable status during execution!

NetBeans Debugger -- Breakpoint

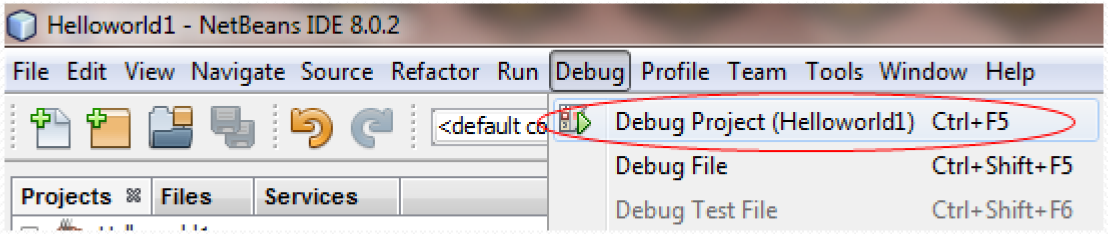
- Breakpoint: an intentional stopping or pausing place in a program
- Create a breakpoint on the program
  - Click the left margin of the editor or Press Ctrl +F8;
  - If pink square is appeared and the line has pink background highlighting, break point is created.

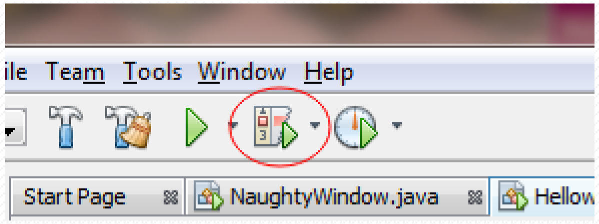


```
    public static void main(String[] args) {
        // TODO code application logic here
        int mul1 = 15;
        int mul2 = 16;
        int product;
        product = mul1*mul2;
        System.out.println("the product is:"+ product);
    }
```

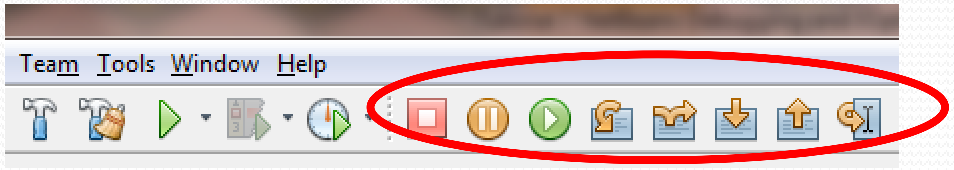
NetBeans Debugger – Debug Project

- Create a breakpoint before the end of the program
- Begin to 'Debug Project'


Helloworld1 - NetBeans IDE 8.0.2
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
<default co
Debug Project (Helloworld1) Ctrl+F5
Debug File Ctrl+Shift+F5
Debug Test File Ctrl+Shift+F6
Projects Files Services



- After we click 'Debug Project'



NetBeans Debugger – Step

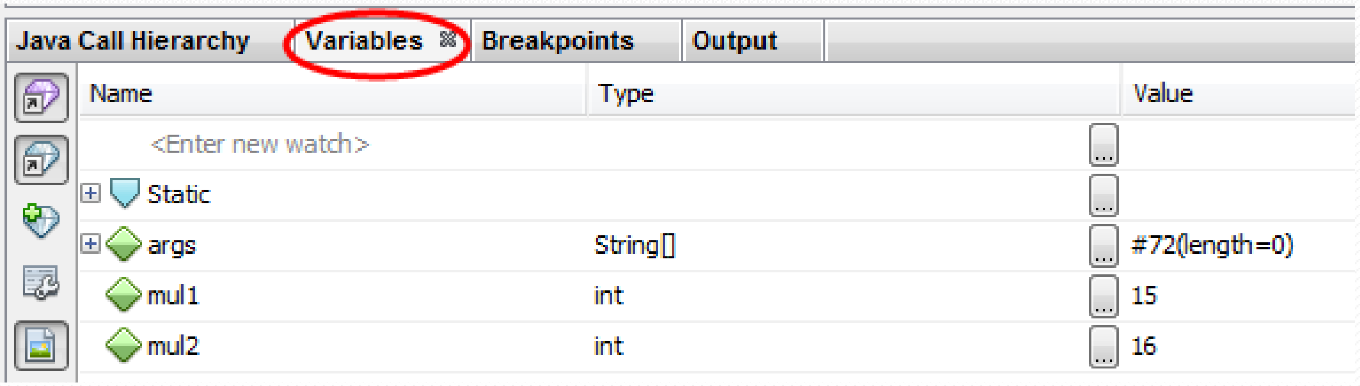
- Step into
  - Execute the next line
  - Move into the execution of the method if next line calls a method
- Step over
  - Execute the next line
  - Do not move into the methods
- Step out
  - Move out from current execution method

NetBeans Debugger – Others

- Continue
  - execute the program until the next break point or it terminates
- Run to cursor
  - use your cursor as a breakpoint
- Finish Debugger Session
  - quit the debugging process

NetBeans Debugger — Variables

- To look up the variable values during debugging
- At "Variables" tab on the bottom, list of variables are shown



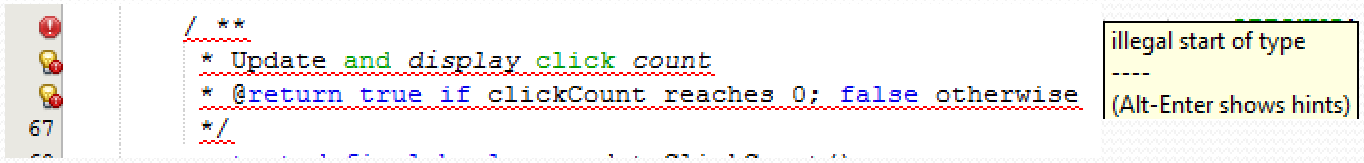
Name	Type	Value
<Enter new watch>		
Static		
args	String[]	#72(length=0)
mul1	int	15
mul2	int	16

- With the 'step' tools in debugger
  - We can keep an eye on whether the variables change

Debug Assignment 1

Debug Assignment 1

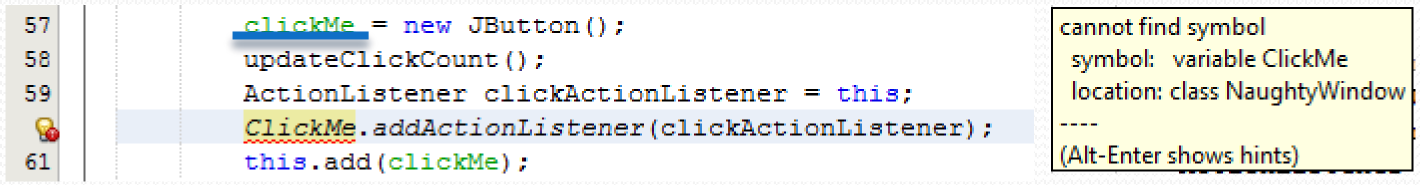
- Comment: no space between /**….*/ or // or /*…*/



```
/ **
 * Update and display click count
 * @return true if clickCount reaches 0; false otherwise
 */
```

```
illegal start of type
----
(Alt-Enter shows hints)
```

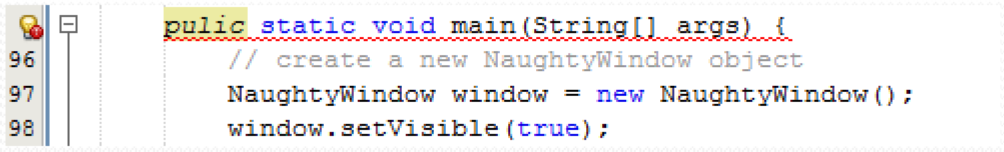
- Java is case sensitive



```
clickMe = new JButton();
updateClickCount();
ActionListener clickActionListener = this;
ClickMe.addActionListener(clickActionListener);
this.add(clickMe);
```

```
cannot find symbol
  symbol:   variable ClickMe
  location: class NaughtyWindow
----
(Alt-Enter shows hints)
```

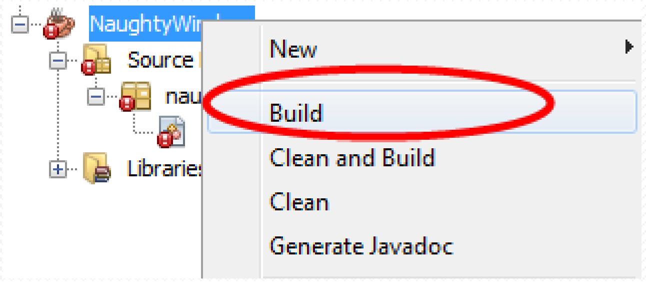
- Typos: reserved word is in blue highlighting

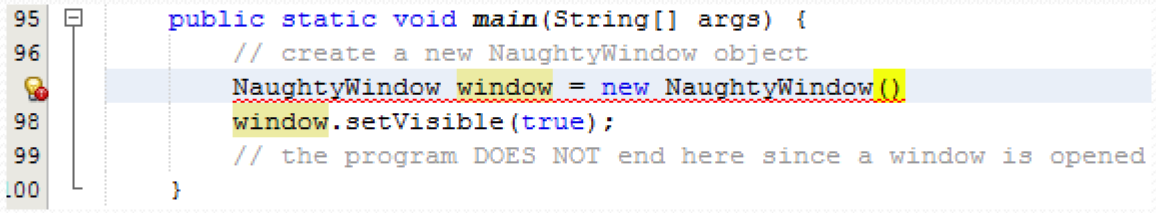


```
pulic static void main(String[] args) {
    // create a new NaughtyWindow object
    NaughtyWindow window = new NaughtyWindow();
    window.setVisible(true);
```

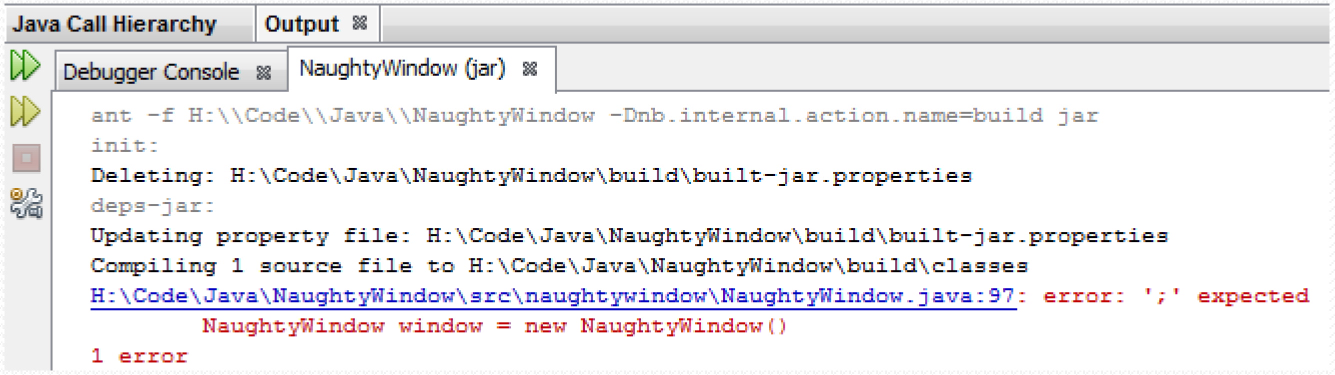
Debug Assignment 1

- If the error info couldn't fix the bugs, try 'build' to show error message
- For example:





```
    public static void main(String[] args) {
        // create a new NaughtyWindow object
        NaughtyWindow window = new NaughtyWindow()
        window.setVisible(true);
        // the program DOES NOT end here since a window is opened
    }
```



```
ant -f H:\\Code\\Java\\NaughtyWindow -Dnb.internal.action.name=build jar
init:
Deleting: H:\Code\Java\NaughtyWindow\build\built-jar.properties
deps-jar:
Updating property file: H:\Code\Java\NaughtyWindow\build\built-jar.properties
Compiling 1 source file to H:\Code\Java\NaughtyWindow\build\classes
H:\Code\Java\NaughtyWindow\src\naughtywindow\NaughtyWindow.java:97: error: ';' expected
        NaughtyWindow window = new NaughtyWindow()
1 error
```

Some tips for Assignment 1

- Clean: deletes the “build\classes\” and the “dist\” folders
- Build: compiles your program and generates “build\classes\*.class” files; generates “dist\NaughtyWindow.jar”
- Run: compiles your program and updates “build\classes\*.class” files; *NO JAR;* runs your classes directly
- Generate JavaDoc: generates documents under “dist\” folder

Q: Right sequence to create both “.jar” file and JavaDoc ?

Debug Exercise 1

Debug Exercise 1

- Fibonacci Sequence

$F(0) = 1$

$F(1) = 1$

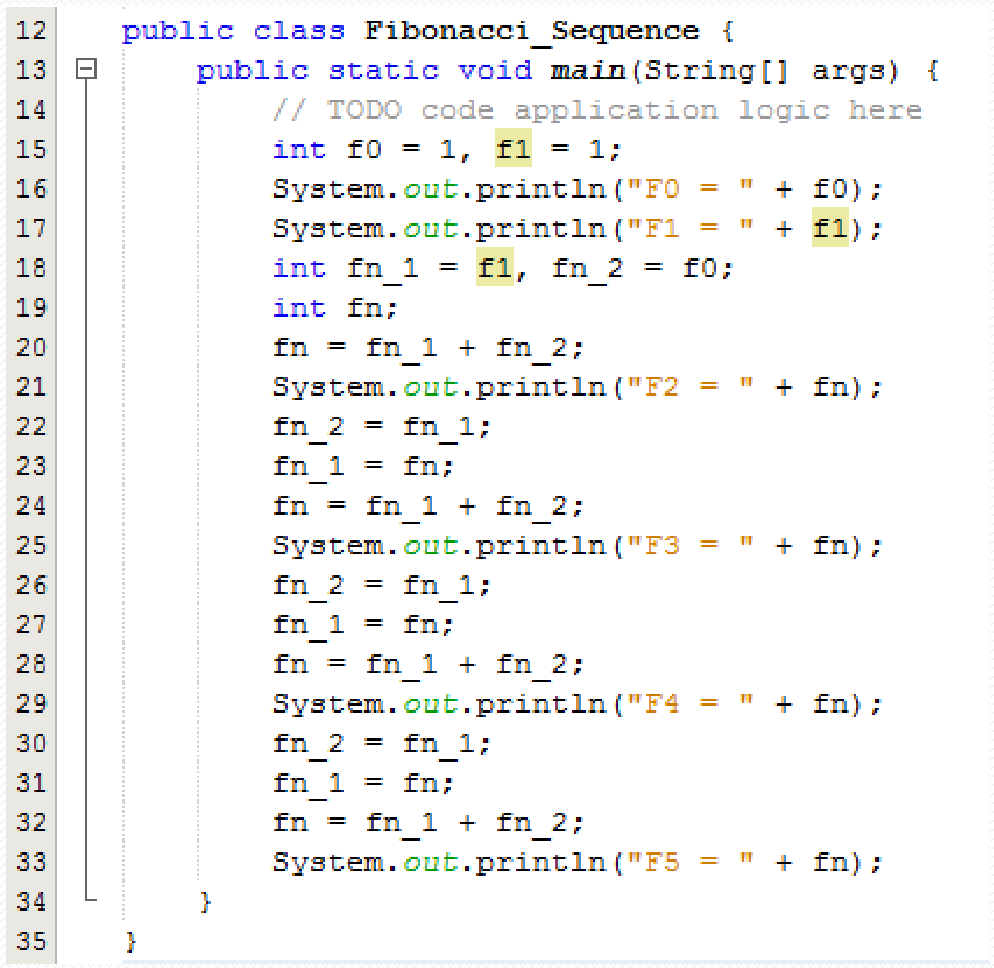
$F(2) = F(0)+F(1)$

.

.

.

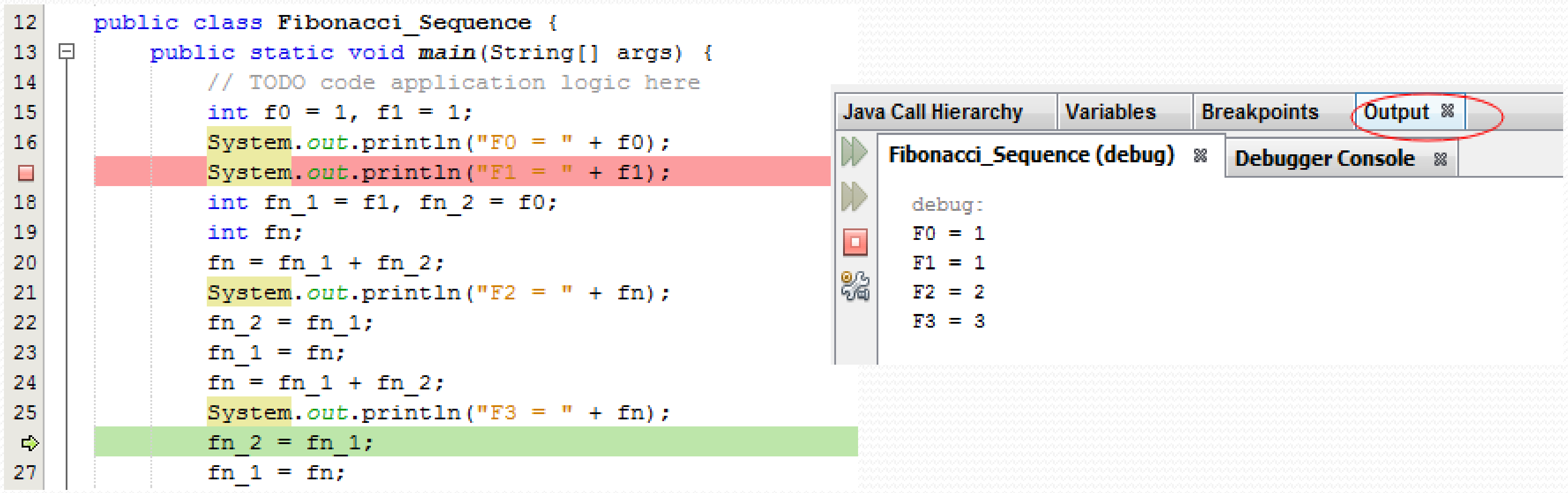
$F(n) = F(n-1) + F(n-2)$



```
public class Fibonacci_Sequence {
    public static void main(String[] args) {
        // TODO code application logic here
        int f0 = 1, f1 = 1;
        System.out.println("F0 = " + f0);
        System.out.println("F1 = " + f1);
        int fn_1 = f1, fn_2 = f0;
        int fn;
        fn = fn_1 + fn_2;
        System.out.println("F2 = " + fn);
        fn_2 = fn_1;
        fn_1 = fn;
        fn = fn_1 + fn_2;
        System.out.println("F3 = " + fn);
        fn_2 = fn_1;
        fn_1 = fn;
        fn = fn_1 + fn_2;
        System.out.println("F4 = " + fn);
        fn_2 = fn_1;
        fn_1 = fn;
        fn = fn_1 + fn_2;
        System.out.println("F5 = " + fn);
    }
}
```

Debug Exercise 1

- Insert print statements
- Set the breakpoint
- Click 'step into' execute one line
    - The green arrow on the left shows the next line to be executed

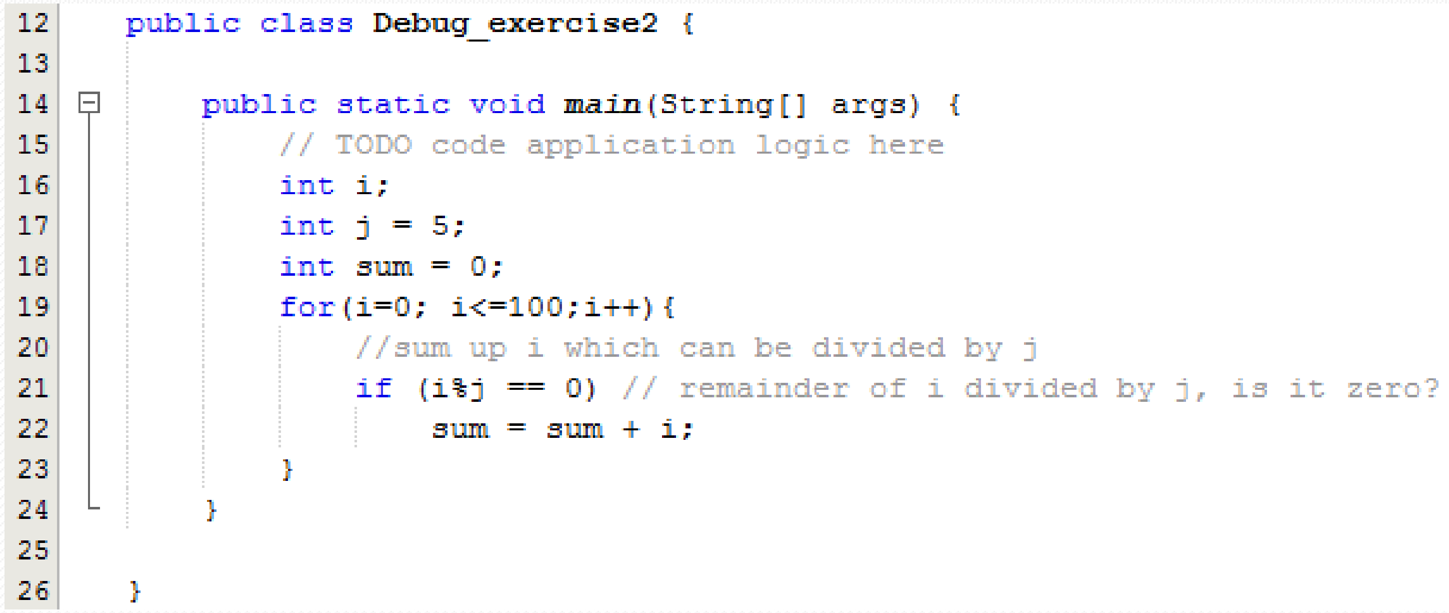


```
public class Fibonacci_Sequence {
    public static void main(String[] args) {
        // TODO code application logic here
        int f0 = 1, f1 = 1;
        System.out.println("F0 = " + f0);
        System.out.println("F1 = " + f1);
        int fn_1 = f1, fn_2 = f0;
        int fn;
        fn = fn_1 + fn_2;
        System.out.println("F2 = " + fn);
        fn_2 = fn_1;
        fn_1 = fn;
        fn = fn_1 + fn_2;
        System.out.println("F3 = " + fn);
        fn_2 = fn_1;
        fn_1 = fn;
```

```
debug:
F0 = 1
F1 = 1
F2 = 2
F3 = 3
```

Debug Exercise 2

- Sum up the number of i(i ∈ [0,100]), if i is divisible by 5.



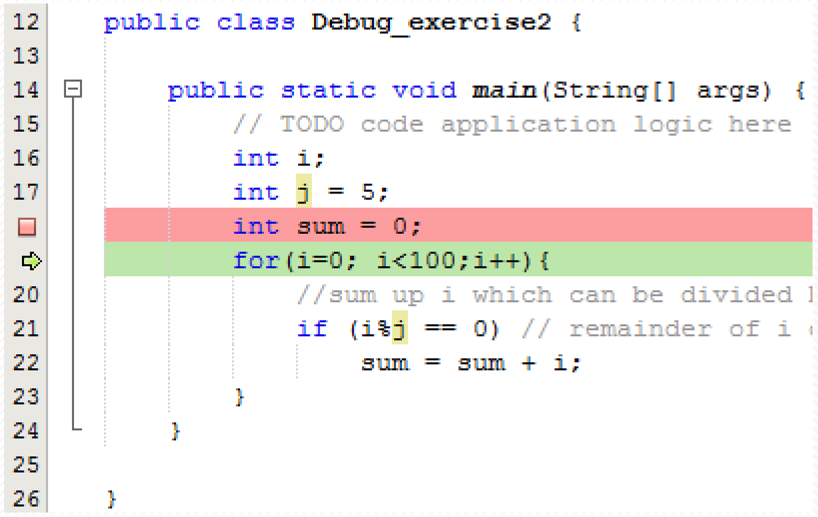
```
public class Debug_exercise2 {

    public static void main(String[] args) {
        // TODO code application logic here
        int i;
        int j = 5;
        int sum = 0;
        for(i=0; i<=100;i++){
            //sum up i which can be divided by j
            if (i%j == 0) // remainder of i divided by j, is it zero?
                sum = sum + i;
        }
    }

}
```

Debug Exercise 2

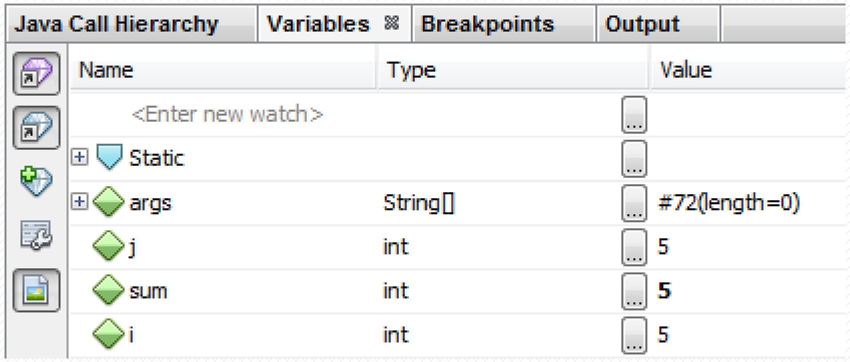
- Set breakpoint
- Click 'Step into' to execute each line
  - Inspect variables, etc.



```
public class Debug_exercise2 {

    public static void main(String[] args) {
        // TODO code application logic here
        int i;
        int j = 5;
        int sum = 0;
        for(i=0; i<100;i++){
            //sum up i which can be divided l
            if (i%j == 0) // remainder of i (
                sum = sum + i;
        }
    }

}
```



Name	Type	Value
<Enter new watch>		
Static		
args	String[]	#72(length=0)
j	int	5
sum	int	5
i	int	5

The end

Thank you!