

# Debugging

Help with finding bugs in your code

---

Produced      Dr. Siobhán Drohan  
by:            Ms. Mairéad Meagher  
                  Siobhan Roche



# Topic List

---

1. What are **bugs**?



2. What are **debuggers**?



3. How do I use them?

# What are bugs?



A software **bug** is an error, flaw, failure or fault in a **computer** program or system that causes it to produce an incorrect or unexpected result, or to behave in unintended ways.



**Software bug - Wikipedia, the free encyclopedia**  
[https://en.wikipedia.org/wiki/Software\\_bug](https://en.wikipedia.org/wiki/Software_bug)

# Bugs can be frustrating to find/fix

---



# Topic List

---

1. What are **bugs**?



2. What are **debuggers**?



3. How do I use them?

# Help is at hand...debuggers!

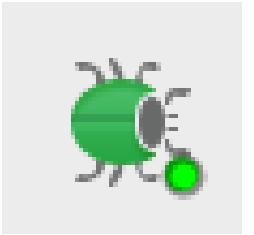
---

A debugger can be **used to fix bugs**

...hence the name debugger!

# Debugger

---



- A **debugger** is a software tool that
  - helps in examining how an application executes
  - allows programmers execute an application one statement at a time.
    - **Step, step into, step out.**
  - typically provides functions
    - to stop and start a program at selected points in the source code(**breakpoints**)
    - to examine the values of variables (**watch, trace**)

# Debugger



- Debuggers are especially useful when your program contains **logical errors**.
  - i.e. errors that the compiler will not pickup but that lead to incorrect results
  - e.g. if your syntax is correct but the logic of your problem solution is faulty.
- Using the debugger, you can **trace** how each of the calculations and changes made to fields/variables happen and hopefully **figure out where the error is occurring**.



# Topic List

---

1. What are **bugs**?



2. What are **debuggers**?



3. How do I use them?

# Debugger

---



- Most IDEs come with a debugger; **IntelliJ** has one.
- We are going to use the **IntelliJ Debugger** to **step** through the debugging of a small program
  - The program iterates over a primitive array of *int* and prints out the largest number in the array.

```
public class Driver {  
  
    public static void main(String args[])  
    {  
        int list[] = {2,5,3,4};  
        int largestNumber = Largest.findLargest(list);  
        System.out.println("Largest number is: " + largestNumber);  
    }  
}
```

Given this code...

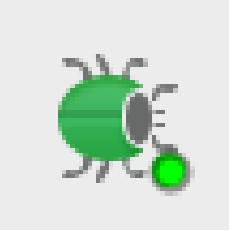
We are expecting this output:

Largest number is: 5

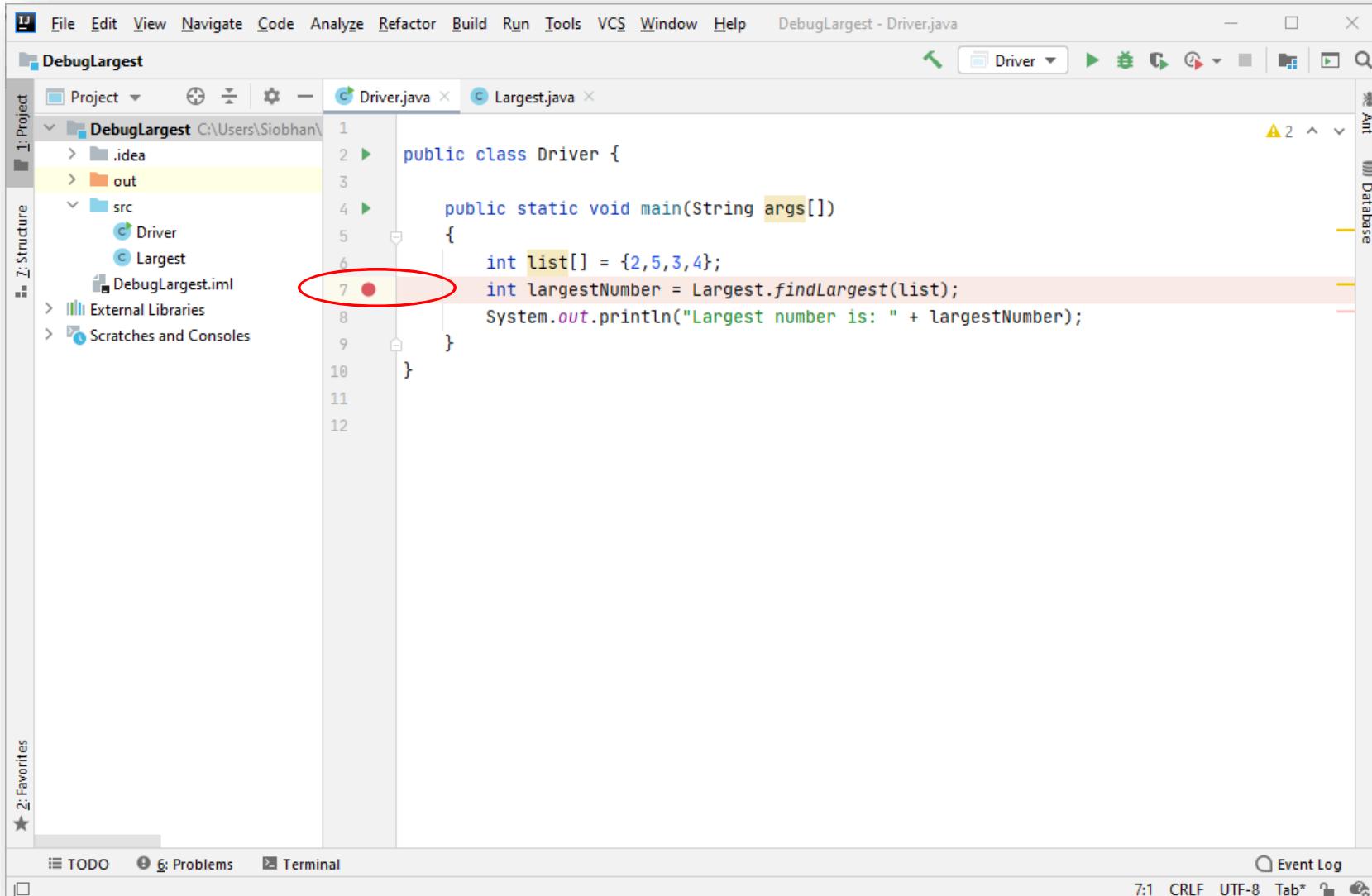
But we get:

Largest number is: 2147483647

```
public class Largest {  
  
    public static int findLargest (int[] list) {  
        int index = 0;  
        int max = Integer.MAX_VALUE;  
  
        for (index = 0; index < list.length; index++) {  
            if (list[index] > max) {  
                max = list[index];  
            }  
        }  
  
        return max;  
    }  
}
```



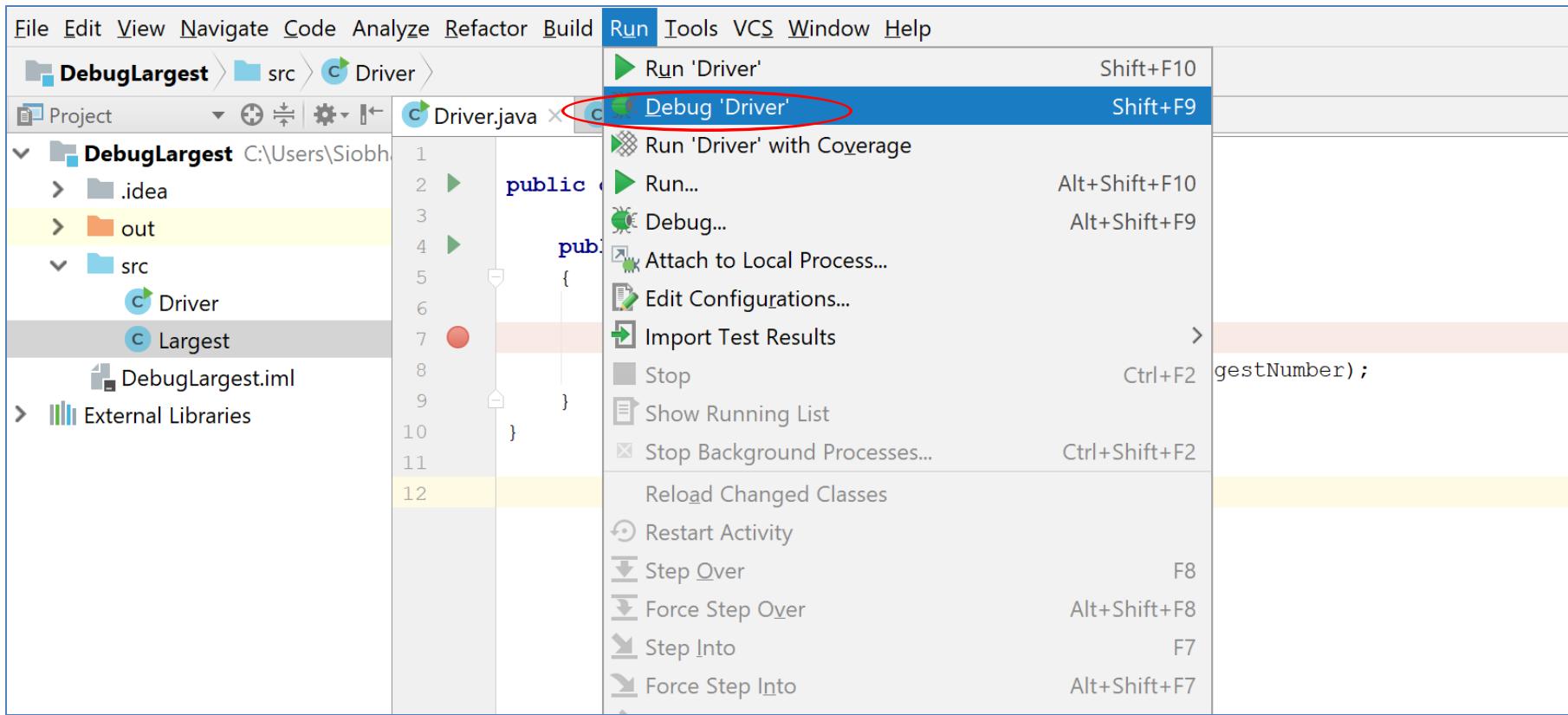
Let's debug the code  
in [IntelliJ](#)  
to help us find the error...



```
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help DebugLargest - Driver.java
DebugLargest
Project Driver.java Largest.java
1 public class Driver {
2     public static void main(String args[])
3     {
4         int list[] = {2,5,3,4};
5         int largestNumber = Largest.findLargest(list);
6         System.out.println("Largest number is: " + largestNumber);
7     }
8 }
9
10
11
12
```

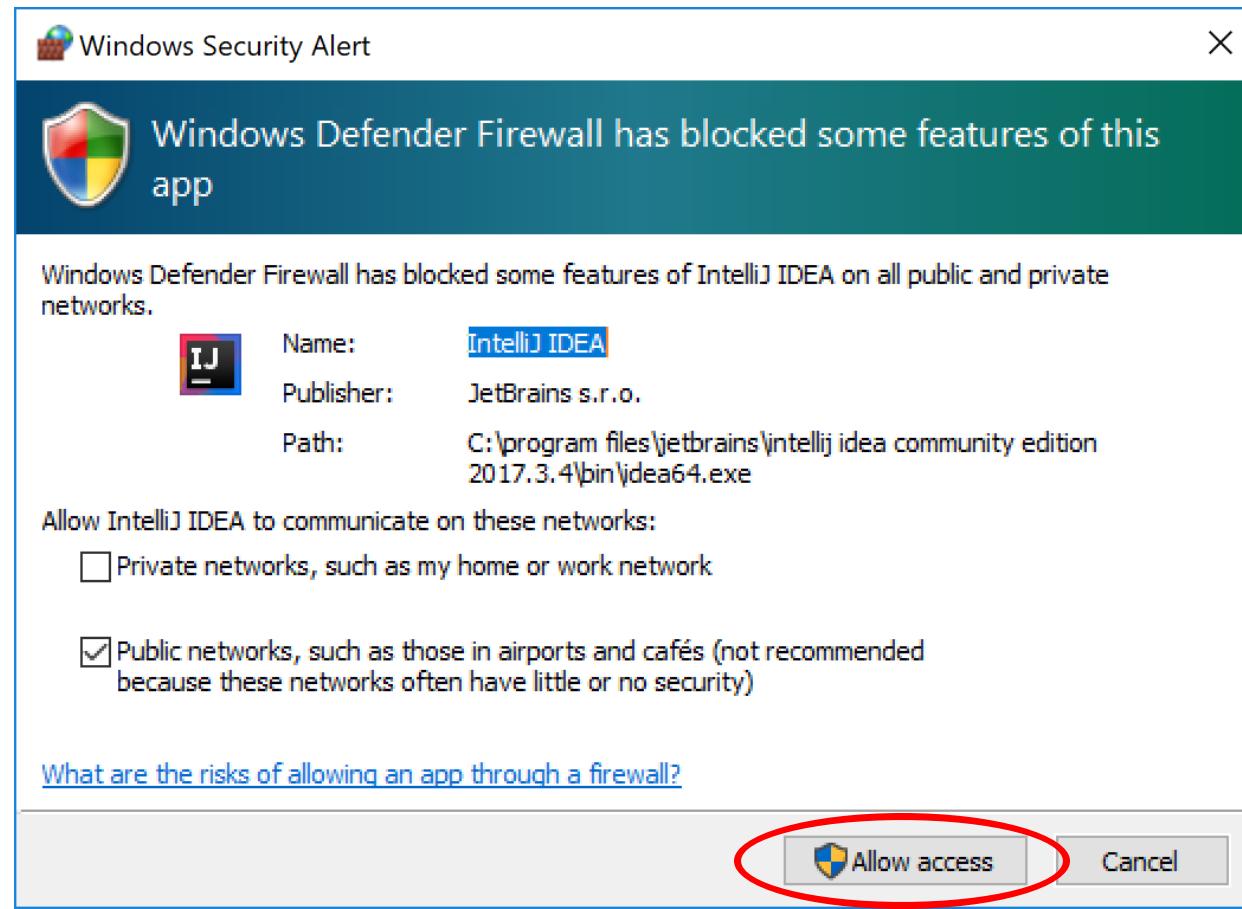
1

Click in the grey margin beside line 7. This will set up a **breakpoint** on this line.



2

From the Run menu, select **Debug 'Driver'**.



3

If this window appears, click on “Allow access”.

The screenshot shows the IntelliJ IDEA interface with the following details:

- File Menu:** File, Edit, View, Navigate, Code, Analyze, Refactor, Build, Run, Tools, VCS, Window, Help.
- Title Bar:** DebugLargest - Driver.java
- Toolbar:** Includes icons for Save, Run, Stop, Refresh, and Search.
- Project Structure:** Shows the project structure with files: .idea, out, src (containing Driver.java and Largest.java), and DebugLargest.iml.
- Code Editor:** Displays the Driver.java code. Line 7 is highlighted with a red circle and a break point icon, indicating the current execution point. The code is as follows:

```
public class Driver {    public static void main(String args[]) {        int list[] = {2,5,3,4};        int largestNumber = Largest.findLargest(list);        System.out.println("Largest number is: " + largestNumber);    }}
```
- Debug Toolbars:** Includes buttons for Step Into, Step Over, Step Out, and Breakpoint.
- Variables View:** Shows the current variable state:
  - args = {String[0]@808}
  - list = {int[4]@809}
    - 0 = 2
    - 1 = 5
    - 2 = 3
    - 3 = 4
- Bottom Navigation:** Includes tabs for Debug, TODO, Problems, Build, Terminal, Event Log, and status message: Build completed successfully in 3 s 20 ms (moments ago).

4

You are now in Debug mode...the program has stopped just before executing line 7.

The screenshot shows the IntelliJ IDEA IDE interface. The code editor displays `Driver.java` with the following code:

```
public class Driver {    public static void main(String args[]) {        int list[] = {2,5,3,4};        int largestNumber = Largest.findLargest(list);        System.out.println("Largest number is: " + largestNumber);    }}
```

The `Largest.java` file is also visible in the editor. The project structure on the left shows a `DebugLargest` project with `.idea`, `out`, and `src` directories. The `src` directory contains `Driver` and `Largest` files, along with an `DebugLargest.iml` file.

The debugger tool window at the bottom is open, showing the `Debugger` tab selected. A red circle highlights the `Step Into` button (a downward-pointing arrow) in the toolbar above the debugger pane. The `Frames` section shows the current frame is `main:7, Driver`. The `Variables` section shows the `list` variable with elements `0 = 2`, `1 = 5`, `2 = 3`, and `3 = 4`.

5

'Step Into' the `findLargest` method...

The screenshot shows the IntelliJ IDEA IDE interface. The top menu bar includes File, Edit, View, Navigate, Code, Analyze, Refactor, Build, Run, Tools, VCS, Window, Help, and DebugLargest - Largest.java. The title bar indicates the project is "DebugLargest" and the file is "Largest". The left sidebar shows the Project structure with "DebugLargest" selected, containing ".idea", "out", "src" (with "Driver" and "Largest" files), "DebugLargest.iml", "External Libraries", and "Scratches and Consoles". The main editor window displays the "Largest.java" code:

```
public class Largest {  
    public static int findLargest (int[] list) {  list: {2, 5, 3, 4}  
        int index = 0;  
        int max = Integer.MAX_VALUE;  
  
        for (index = 0; index < list.length; index++) {  
            if (list[index] > max) {  
                max = list[index];  
            }  
        }  
  
        return max;  
}
```

The "Driver" tab is selected in the bottom-left of the editor. The bottom half of the screen shows the "Debug" tool window. The "Frames" tab is active, showing the current stack trace:  
"main:1, Driver" (highlighted)  
"findLargest:5, Largest"  
The "Variables" tab shows the state of variables:  
list = {int[4]@809}  
0 = 2  
1 = 5  
2 = 3  
3 = 4  
list.length = 4

6

Now that we are in the **findLargest** method, we want to execute each line while monitoring the value of **Max**. This should help us locate the error...

The screenshot shows the IntelliJ IDEA IDE interface. In the top navigation bar, the project is named "DebugLargest". The code editor displays the "Largest.java" file, which contains a static method "findLargest" that iterates through an array to find the maximum value. A red circle highlights the line of code "int max = Integer.MAX\_VALUE;". The bottom half of the screen shows the "Debug" tool window, which includes tabs for "Debugger" and "Console". The "Debugger" tab is active, showing the current stack frames. The frame "findLargest:6, Largest" is selected, and its local variables are listed: "list = {int[4]@809}" with elements 0=2, 1=5, 2=3, 3=4, and "index = 0". A red circle highlights the "Step Into" button in the toolbar of the debugger tool window. The status bar at the bottom indicates a successful build.

```
public class Largest {  
    public static int findLargest (int[] list) {  list: {2, 5, 3, 4}  
        int index = 0;  index: 0  
        int max = Integer.MAX_VALUE;  
  
        for (index = 0; index < list.length; index++) {  
            if (list[index] > max) {  
                max = list[index];  
            }  
        }  
  
        return max;  
}
```

Debug: Driver

Debugger

Variables

list = {int[4]@809}

- 0 = 2
- 1 = 5
- 2 = 3
- 3 = 4

index = 0

list[index] = 2

list.length = 4

Event Log

Build completed successfully in 3 s 20 ms (5 minutes ago)

7

'Step Into' again → execution now stopped on line 6

The screenshot shows the IntelliJ IDEA IDE interface. The top menu bar includes File, Edit, View, Navigate, Code, Analyze, Refactor, Build, Run, Tools, VCS, Window, Help, and DebugLargest - Largest.java. The title bar indicates the current project is 'DebugLargest' and the file is 'Largest'. The left sidebar shows the Project structure with 'DebugLargest' selected, containing '.idea', 'out', 'src' (with 'Driver' and 'Largest' files), 'External Libraries', and 'Scratches and Consoles'. The main editor window displays the 'Largest.java' code:

```
public class Largest {  
    public static int findLargest (int[] list) {    list: {2, 5, 3, 4}  
        int index = 0;    index: 0  
        int max = Integer.MAX_VALUE;    max: 2147483647  
        for (index = 0; index < list.length; index++) {    index: 0    list: {2, 5, 3, 4}  
            if (list[index] > max) {  
                max = list[index];  
            }  
        }  
  
        return max;  
}
```

A red circle highlights the line number 8 in the code editor. The bottom half of the screen shows the 'Debug' tool window. The 'Debugger' tab is active. The 'Frames' section shows the current stack frame: 'main:7, Driver' calling 'findLargest:8, Largest'. The 'Variables' section shows the local variables:

Variable	Type	Value
list	int[4]@809	0 = 2 1 = 5 2 = 3 3 = 4
index	int	0
max	int	2147483647
list[index]	int	2
list.length	int	4

The 'Step Into' button in the 'Debugger' toolbar is highlighted with a red circle.

8

Step Into → execution now stopped on line 8...note the value of **max**.

The screenshot shows the IntelliJ IDEA IDE interface. The code editor displays the `Largest.java` file with the following code:

```
public class Largest {  
    public static int findLargest (int[] list) {  
        list: {2, 5, 3, 4}  
        int index = 0; index: 0  
        int max = Integer.MAX_VALUE; max: 2147483647  
  
        for (index = 0; index < list.length; index++) {  
            if (list[index] > max) {  
                list: {2, 5, 3, 4} index: 0 max: 2147483647  
                max = list[index];  
            }  
        }  
  
        return max;  
    }  
}
```

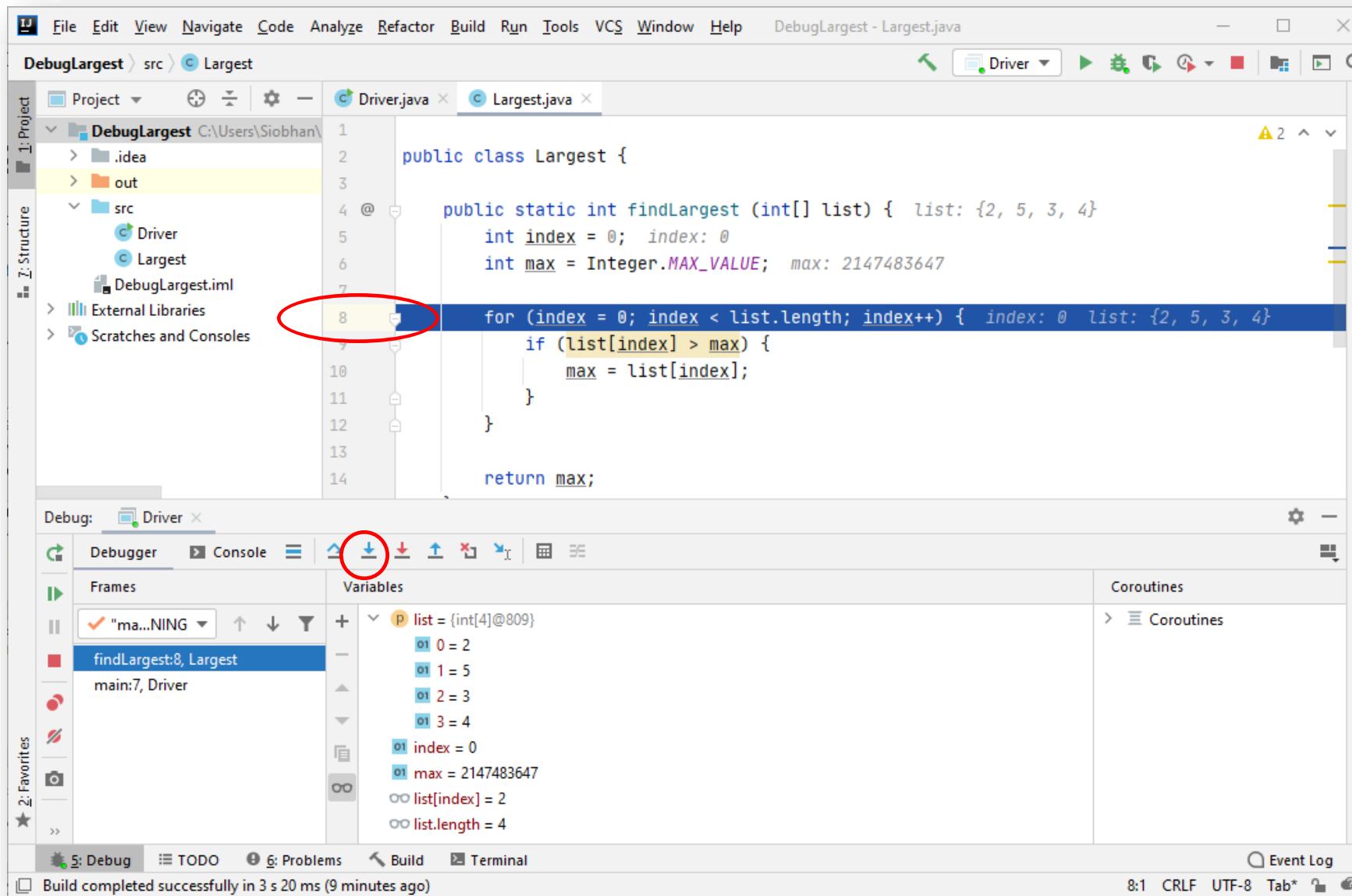
A red circle highlights the line number 9, which contains the assignment statement `max = list[index];`. The debugger tool window at the bottom shows the current stack frame: `findLargest:9, Largest`. The variables pane shows the state of local variables:

Variable	Type	Value
list	int[4]	{2, 5, 3, 4}
index	int	0
max	int	2147483647
list[index]	int	2
list.length	int	4

The status bar at the bottom indicates: "Build completed successfully in 3 s 20 ms (8 minutes ago)".

9

Step Into → execution now stopped on line 9...



10

**Step Into** → execution now stopped back on line 8...  
can you see the problem?

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help DebugLargest - Largest.java

DebugLargest > src > Largest

Driver.java Largest

1: Project 1: Structure

DebugLargest C:\Users\Siobhan\

.idea  
out  
src  
Driver  
Largest  
DebugLargest.iml  
External Libraries  
Scratches and Consoles

```
public class Largest {  
    public static int findLargest (int[] list) {  list: {2, 5, 3, 4}  
        int index = 0;  index: 0  
        int max = Integer.MAX_VALUE;  max: 2147483647  
        for (index = 0; index < list.length; index++) {  index: 0  list: {2, 5, 3, 4}  
            if (list[index] > max) {  
                max = list[index];  
            }  
        }  
  
        return max;  
}
```

Debug: Driver

Debugger Console

Frames Variables Coroutines

"main:8, Largest"

list = {int[4]@809}  
0 = 2  
1 = 5  
2 = 3  
3 = 4  
index = 0  
max = 2147483647  
list[index] = 2  
list.length = 4

Coroutines

5: Debug TODO 6: Problems Build Terminal

Build completed successfully in 3 s 20 ms (9 minutes ago)

Event Log

Max is set to the maximum possible value an int can be.

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help DebugLargest - Largest.java

DebugLargest > src > Largest

Project Structure

Driver.java x Largest.java x

```
1 public class Largest {  
2  
3     public static int findLargest (int[] list) {    list: {2, 5, 3, 4}  
4         int index = 0;  index: 0  
5         int max = Integer.MAX_VALUE;  max: 2147483647  
6  
7         for (index = 0; index < list.length; index++) {  index: 0  list: {2, 5, 3, 4}  
8             if (list[index] > max) {  
9                 max = list[index];  
10            }  
11        }  
12  
13    }  
14  
15    return max;  
16}
```

Debug: Driver x

Debugger Console

Frames

- "main:7, Driver"
- findLargest:8, Largest

Variables

- list = {int[4]}@809
  - o1 0 = 2
  - o1 1 = 5
  - o1 2 = 3
  - o1 3 = 4
- index = 0
- max = 2147483647
- list[index] = 2
- list.length = 4

Coroutines

No value in the list will be greater than this max value

Event Log

Build completed successfully in 3 s 20 ms (9 minutes ago)

This is the **bug**...  
we should have set  
the max value  
to the first value in the list.

```
public class Largest {  
    public static int findLargest (int[] list) {  
        int index = 0; index: 0  
        int max = Integer.MAX_VALUE; max: 2147483647  
        for (index = 0; index < list.length; index++) { index:  
            if (list[index] > max) {  
                max = list[index];  
            }  
        }  
  
        return max;  
    }  
}
```

Driver

Debug: Driver

Debugger

Frames

"main:8, Largest"

Variables

list = {int[4]@809}  
0 = 2  
1 = 5  
2 = 3  
3 = 4  
index = 0  
max = 2147483647  
list[index] = 2  
list.length = 4

Coroutines

Coroutines

Event Log

Build completed successfully in 3 s 20 ms (9 minutes ago)

# Fixing the bug

---

- Instead of the line of code:

**int max = Integer.MAX\_VALUE;**

- We need:

**int max = list[0];**



# Some IntelliJ debugger buttons...

The screenshot shows the IntelliJ IDEA debugger window with several buttons highlighted by red boxes:

- Left sidebar buttons:** Step Into (green arrow), Step Over (grey arrow), Step Out (red circle), Break (red square), and Stop (camera).
- Top toolbar buttons:** Debugger (selected), Console, and several other icons including a list, up, down, and search.

**Frames pane:** Shows the current stack frames:

- "main:7, Driver" (selected)
- "findLargest:8, Largest"
- "ma...NING" (with a dropdown arrow)

**Variables pane:** Shows the current variable values:

Variable	Value
list	{int[4]}@809
list[0]	2
list[1]	5
list[2]	3
list[3]	4
index	0
max	2147483647
list[index]	2
list.length	4

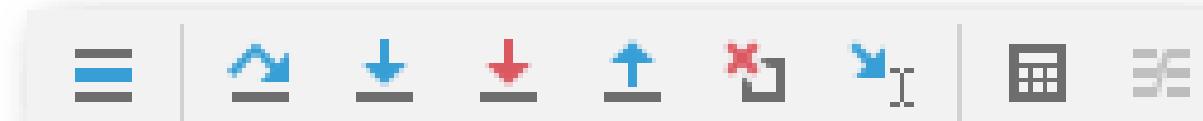
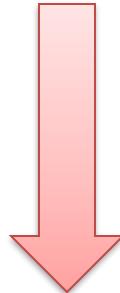
# Some Debugger buttons...



- **Step Over**



to step over the next method call (without entering it)  
at the currently executing line of code.



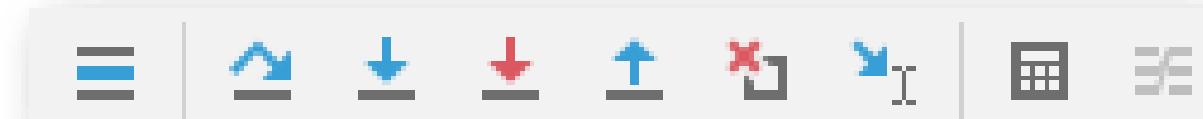
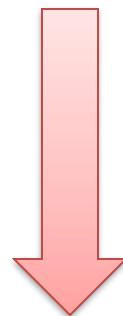
# Some Debugger buttons...



- **Step Into**



step into the next method call at the currently executing line of code.



# Some Debugger buttons...

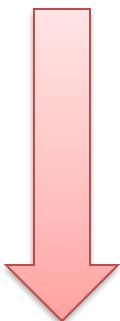


- **Step Out**



executes the remaining lines of a method in which the current execution point lies.

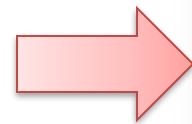
The next statement displayed is the statement following the method call.



# Some Debugger buttons...



- **Resume Program**



resume the execution of the currently suspended program; it will stop again at the next available breakpoint (if any).



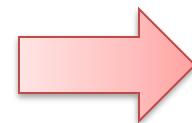
# Some Debugger buttons...



- **Terminate**



to terminate the launch associated  
with the selected debug target  
i.e. **stop** the program.



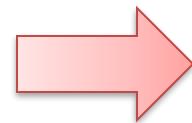
# Some Debugger buttons...



- **Show breakpoints**

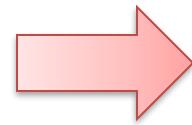


show all the breakpoints (in a pop up window) in the program.



# Some Debugger buttons...

---



- **Rerun program**



start the program again.





Now it's your turn!

- practice using the debugger on this code (there is a step in your labs to provide support).

Any  
Questions?

