### Terminology

**Debugging.** Identifying and fixing errors in computer code.

A **compile-time error** is an error that occurs when compiling the program.

```java
public class BadHelloWorld {
    public state void main(String[]) {
        print "Hello, World"
    }
}
```

A **run-time error** is an error that occurs while the program is running.

```java
public class DivideByZero {
    public static void main(String[] args) {
        int x = 10;
        int y = 0;
        int z = x / y;
    }
}
```

integer division with denominator = 0
DEBUGGING

› bug hunt
› live coding
In-class polling: QuizSocket

Step 1. http://www.quizsocket.com
Step 2. Type ABCDEFG for Quiz ID and click Join Quiz.
Step 3. Click desired option for each question.
Bug 1

What is the value of the variable x at the end of the computation?

```
int numerator = 3;
int denominator = 5;
double x = (double) (numerator/denominator);
```

A. 0.0  
B. 0.59999999999999997779553950749686919152736663818359375  
C. 0.6  
D. compile-time error  
E. run-time error
Bug 2

What is the values of \texttt{max} at the end of the computation?

```java
int a = 2;
int b = 4;
int c = -4;
int max = Math.max(a, b, c);
```

A. \(-4\)
B. \(2\)
C. \(4\)
D. \textit{compile-time error}\nE. \textit{run-time error}\n
Bug 3

What does the following code fragment print?

double humidity = 40.0;
if (40 <= humidity <= 60.0)
    System.out.println("confortable");

A. confortable
B. 40.0
C. nothing
D. compile-time error
E. run-time error
What does the following code fragment print?

```java
int x = 0;
if (x == 0) System.out.println("zero");
else System.out.println("not zero");
```

A. zero  
B. not zero  
C. nothing  
D. compile-time error  
E. run-time error
What does the following code fragment print?

```java
int x = -123;
boolean isPositive = (x > 0);
if (isPositive = true) System.out.println("positive");
else System.out.println("not positive");
```

A. "positive"
B. "not positive"
C. nothing
D. compile-time error
E. run-time error
What is the values of `result` at the end of the computation?

```java
int l = 0;
int 0 = 1;
int l00 = 2;
int result = l00 + 100 + 0*0 + 1*1 + 1001;
```

A. 1024
B. 1104
C. 1201
D. 1202
E. `compile-time error`
What does the following code fragment print?

```java
int default = 18;
int x = 24;
boolean resetToDefault = true;
if (resetToDefault)
    x = default;
System.out.println(x);
```

A. zero
B. not zero
C. nothing
D. compile-time error
E. run-time error
Why is it called "debugging"?

Admiral Grace Hopper's operational logbook for the Harvard Mark II computer.
Bug 7a

What does the following code fragment print?

```java
int x = -123;
if (x > 0) {
    System.out.println("positive");
}
```

A. "positive"
B. "not positive"
C. nothing
D. compile-time error
E. run-time error
What does the following code fragment print?

```java
int x = 99;
while (x > 0) {
    x = x / 2;
}
System.out.println(x);
```

A. 0  
B. 1  
C. 99  
D. nothing  
E. compile-time error
Bug 8

What is the value of the variable `force` at the end of the computation?

double G = 1.0;       // in some other universe
double mass1 = 1.0;
double mass2 = 2.0;
double r = 5.0;
double force = G * mass1 * mass2 / r * r;

A. 0.0
B. 0.04
C. 0.08
D. 2.0
E. NaN
Bug 9

Which is the value of the variable \( x \) at the end of the computation?

\[
\text{double } x = 180.0; \\
\text{Math.toRadians}(x);
\]

A. 0.0  
B. 1.5707963267948966  
C. 3.141592653589793  
D. 180.0  
E. 10313.2
“If debugging is the process of removing bugs, then programming must be the process of putting them in.” — Edsger Dijkstra

“Everyone knows that debugging is twice as hard as writing a program in the first place. So if you’re as clever as you can be when you write it, how will you ever debug it?” — Brian Kernighan
DEBUGGING

- bug hunt
- live coding
Debugging a complete program

**Goal.** Write a program that reads an integer command-line argument $n$ and prints $n! = 1 \times 2 \times 3 \times \ldots \times n$. 

<table>
<thead>
<tr>
<th>$n$</th>
<th>$n!$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>4</td>
<td>24</td>
</tr>
<tr>
<td>5</td>
<td>120</td>
</tr>
<tr>
<td>6</td>
<td>720</td>
</tr>
<tr>
<td>\vdots</td>
<td>\vdots</td>
</tr>
</tbody>
</table>

_factorial function_
Goal. Write a program that reads an integer command-line argument $n$ and prints $n! = 1 \times 2 \times 3 \times \ldots \times n$. 

```java
public class Factorial {
    public static void main(String[] args) {
        int n = Integer.parseInt(args[0]);
        for (int i = 0; i < n; i++)
            product = product * i;
    }

    System.out.println(product);
}
```

error: <identifier> expected
Bug 1 (undeclared variable)

**Goal.** Write a program that reads an integer command-line argument $n$ and prints $n! = 1 \times 2 \times 3 \times \ldots \times n$.

```java
public class Factorial {
    public static void main(String[] args) {
        int n = Integer.parseInt(args[0]);
        for (int i = 0; i < n; i++)
            product = product * i;  
        System.out.println(product);
    }
}
```

```plaintext
error: cannot find symbol
  product = product * i;
  ^
symbol:   variable product
...
2 errors
```
Goal. Write a program that reads an integer command-line argument \( n \) and prints \( n! = 1 \times 2 \times 3 \times \ldots \times n \).
Bug 3 (curly braces missing, misleading indentation)

Goal. Write a program that reads an integer command-line argument $n$ and prints $n! = 1 \times 2 \times 3 \times \ldots \times n$.

```java
public class Factorial {
    public static void main(String[] args) {
        int n = Integer.parseInt(args[0]);
        int product = 1;
        for (int i = 1; i <= n; i++) {
            product *= i;
        }
        System.out.println(product);
    }
}
```

error: cannot find symbol
  product = product * i;
     ^
symbol:   variable i
1 error
Bug 4 (initialization in wrong place)

Goal. Write a program that reads an integer command-line argument $n$ and prints $n! = 1 \times 2 \times 3 \times \ldots \times n$.

```java
public class Factorial {
    public static void main(String[] args) {
        int n = Integer.parseInt(args[0]);
        int product;
        for (int i = 0; i < n; i++) {
            product = 0;
            product = product * i;
        }
        System.out.println(product);
    }
}
```

error: variable product might not have been initialized
System.out.println(product);
^
1 error
Bug 5 (forgot to type command-line arguments)

Goal. Write a program that reads an integer command-line argument $n$ and prints $n! = 1 \times 2 \times 3 \times \ldots \times n$.

```java
public class Factorial {
    public static void main(String[] args) {
        int n = Integer.parseInt(args[0]);
        int product = 1;
        for (int i = 0; i < n; i++) {
            product = product * i;
        }
        System.out.println(product);
    }
}
```

% java Factorial
Exception in thread "main"
java.lang.ArrayIndexOutOfBoundsException: 0
    at Factorial.main(Factorial.java:3)
Bug 6 (off-by-one error)

**Goal.** Write a program that reads an integer command-line argument \( n \) and prints \( n! = 1 \times 2 \times 3 \times \ldots \times n \).

```java
public class Factorial {
    public static void main(String[] args) {
        int n = Integer.parseInt(args[0]);
        int product = 1;
        for (int i = 0; i < n; i++) {
            product = product * i;
        }
        System.out.println(product);
    }
}
```

```
% java Factorial 3
0
%
% java Factorial 5
0
```
Hooray!

**Goal.** Write a program that reads an integer command-line argument $n$ and prints $n! = 1 \times 2 \times 3 \times \ldots \times n$.

```java
public class Factorial {
    public static void main(String[] args) {
        int n = Integer.parseInt(args[0]);
        int product = 1;
        for (int i = 1; i <= n; i++) {
            product = product * i;
        }
        System.out.println(product);
    }
}
```

% java Factorial 3
6

% java Factorial 5
120