### Terminology

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

- A **compile-time error** is an error that prevents the program from compiling.
- A **run-time exception** is an error that causes the program to abort prematurely.
- A **logic error** is an error that causes the program to produce an incorrect result.

<table>
<thead>
<tr>
<th>Error Type</th>
<th>valid program?</th>
<th>completes execution?</th>
<th>correct output?</th>
<th>example</th>
</tr>
</thead>
<tbody>
<tr>
<td>compile-time error</td>
<td>✗</td>
<td></td>
<td>✗</td>
<td>variable used before it is declared</td>
</tr>
<tr>
<td>run-time exception</td>
<td>✔</td>
<td>✗</td>
<td>✗</td>
<td>integer division by zero</td>
</tr>
<tr>
<td>logic error</td>
<td>✔</td>
<td>✔</td>
<td>✗</td>
<td>integer division (when floating-point division needed)</td>
</tr>
<tr>
<td>goal</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>Programming Exam 1 (we hope!)</td>
</tr>
</tbody>
</table>
DEBUGGING

- bug hunt
- live coding
What’s the bug?
What is the value of the variable $x$ at the end of the computation?

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

A. 0.0
B. 0.599999999999997779553950749686919152736663818359375
C. 0.6
D. *compile-time error*
E. *run-time error*
What is the values of max at the end of the computation?

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

A. -4
B. 2
C. 4
D. compile-time error
E. run-time error
Bug 3

What does the following code fragment print?

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

A. "comfortable"
B. 40.0
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 \texttt{result} at the end of the computation?

\begin{code}
\begin{verbatim}
int l = 0;
int 0 = 1;
int 100 = 2;
int result = 100 + 100 + 0*0 + 1*1 + 1001;
\end{verbatim}
\end{code}

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

```java
int default = 123;
int x = 456;
boolean resetToDefault = true;
if (resetToDefault)
    x = default;
System.out.println(x);
```

A. 123
B. 456
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 8

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

```java
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
Which is the value of the variable $x$ at the end of the computation?

```java
double x = 180.0;
Math.toRadians(x);
```

A. 0.0  
B. 1.5707963267948966  
C. 3.141592653589793  
D. 180.0  
E. 10313.2
“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 a command-line argument $n$ and prints $n! = 1 \times 2 \times 3 \times \cdots \times n$.

<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>...</td>
<td>...</td>
</tr>
</tbody>
</table>

factorial function

factorial function
**Goal.** Develop a program, fixing compile-time, run-time, and logic errors.