Inf1B
Conditionals and Loops

Perdita Stevens
adapting earlier versions by Ewan Klein, Volker Seeker, et al.

School of Informatics

\(^1\)Thanks to Sedgewick\&Wayne for much of this content
Conditional Statements
Control Flow

Control flow:

- A sequence of statements that are actually executed in a program

Diagram:

1. Statement 1
2. Statement 2
3. Statement 3
4. Statement 4
Control Flow

Control flow:

- A sequence of statements that are actually executed in a program
- **Conditionals** and **loops** enable us to choreograph control flow
If Statement

If / conditional statement:

- Evaluate a boolean expression $E$.
- If value of $E$ is true, execute some statements.
- If value of $E$ is false, execute some other statements — this is the else part of a conditional statement.

```
if (boolean expression) {
    statement T;
}
else {
    statement F;
}
```
If Statement

If / conditional statement:

- Evaluate a boolean expression $E$.
- If value of $E$ is true, execute some statements.
- If value of $E$ is false, execute some other statements — this is the else part of a conditional statement.

```java
if (boolean expression) {
    statement T;
} else {
    statement F;
}
```
can be any sequence of statements

```java
if (x > y) {
    int t = x;
    x = y;
    y = t;
}
```
If Statement

If / conditional statement:

- Evaluate a boolean expression $E$.
- If value of $E$ is `true`, execute some statements.
- If value of $E$ is `false`, execute some other statements — this is the else part of a conditional statement.

```java
if (boolean expression) {
    statement T;
}
else {
    statement F;
}
```
can be any sequence of statements

```java
if (x > y) {
    int t = x;
    x = y;
    y = t;
}
```

`boolean expression`

`sequence of statements`
If Statement

If / conditional statement — sometimes called branching structures:
**If Statement**

**If / conditional statement:**

- Evaluate a boolean expression.
- If true, execute some statements.
- If false, execute some other statements.

```plaintext
if (x < 0) x = -x;
if (x > y) max = x;
else max = y;
```
### If Statement: Examples

<table>
<thead>
<tr>
<th><strong>absolute value</strong></th>
<th>if ((x &lt; 0)) (x = -x;)</th>
</tr>
</thead>
</table>
| **put \(x\) and \(y\) into ascending order (swap)** | if \((x > y)\) {
  
  int temp = \(x\);
  
  \(x = y;\)
  
  \(y = temp;\)
  
} |
| **maximum of \(x\) and \(y\)** | if \((x > y)\) max = \(x\);

else \(max = y;\) |
| **error check for division operation** | if \((\text{den} == 0)\) {

  System.out.println("Division by zero");

} else {

  System.out.println("Quotient = " + num / \(\text{den}\));

} |
Loops (While)
While Loop

The *while* loop is a structure for expressing repetition.

- Evaluate a boolean expression.
- If true, execute some statements.
- Repeat.

```plaintext
while (boolean expression) {
    statement 1;
    statement 2;
}
```
The while loop is a structure for expressing repetition.

- Evaluate a boolean expression.
- If true, execute some statements.
- Repeat.

```java
while (boolean expression) {
    statement 1;
    statement 2;
}
```
While Loop: Powers of Two

Print powers of 2 that are \( \leq 2^n \) for some \( n \).

- Increment loop counter \( i \) by 1, from 0 to \( n \).
- Double \( val \) each time.

```java
int i = 0;
int val = 1;
while (i <= n) {
    System.out.println(i + " " + val);
    i = i + 1;
    val = 2 * val;
}
```

Output

- \( i = 0 \)
- \( 0 \)
- \( i = 1 \)
- \( 1 \)
- \( i = 2 \)
- \( 2 \)
- \( i = 3 \)
- \( 4 \)
- \( i = 4 \)
- \( 8 \)
- \( i = 5 \)
- \( 16 \)
- \( i = 6 \)
- \( 32 \)
- \( i = 7 \)
- \( 64 \)
- \( i = 8 \)
- \( 128 \)

Start Again
Print powers of 2 that are $\leq 2^n$ for some $n$. Set $n = 6$.

- Increment loop counter $i$ by 1, from 0 to $n$.
- Double $val$ each time.

```java
int i = 0;
int val = 1;
while (i <= n) {
    System.out.println(i + " " + val);
    i = i + 1;
    val = 2 * val;
}
```
While Loop: Powers of Two

Print powers of 2 that are \( \leq 2^n \) for some \( n \). Set \( n = 6 \).

- Increment loop counter \( i \) by 1, from 0 to \( n \).
- Double \( \text{val} \) each time.

```
int i = 0;
int val = 1;
while (i <= n) {
    System.out.println(i + " " + val);
    i = i + 1;
    val = 2 * val;
}
```

<table>
<thead>
<tr>
<th>i</th>
<th>val</th>
<th>i ( \leq n )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>true</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>true</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>true</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>true</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>true</td>
</tr>
<tr>
<td>5</td>
<td>32</td>
<td>true</td>
</tr>
<tr>
<td>6</td>
<td>64</td>
<td>true</td>
</tr>
<tr>
<td>7</td>
<td>128</td>
<td>false</td>
</tr>
</tbody>
</table>

Start Again
While Loop: Powers of Two

Print powers of 2 that are \( \leq 2^n \) for some \( n \). Set \( n = 6 \).

- Increment loop counter \( i \) by 1, from 0 to \( n \).
- Double \( \text{val} \) each time.

```java
int i = 0;
int val = 1;
while (i <= n) {
    System.out.println(i + " " + val);
    i = i + 1;
    val = 2 * val;
}
```

<table>
<thead>
<tr>
<th>i</th>
<th>val</th>
<th>i ≤ n</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>true</td>
<td>0 1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>true</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>true</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>true</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>true</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>32</td>
<td>true</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>64</td>
<td>true</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>128</td>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

Start Again
While Loop: Powers of Two

Print powers of 2 that are $\leq 2^n$ for some $n$. Set $n = 6$.

- Increment loop counter $i$ by 1, from 0 to $n$.
- Double $val$ each time.

```
int i = 0;
int val = 1;
while (i <= n) {
    System.out.println(i + " " + val);
    i = i + 1;
    val = 2 * val;
}
```

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>i</td>
<td>val</td>
<td>i ≤ n</td>
<td>Output</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>true</td>
<td>0 1</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
While Loop: Powers of Two

Print powers of 2 that are $\leq 2^n$ for some $n$. Set $n = 6$.

- Increment loop counter $i$ by 1, from 0 to $n$.
- Double $val$ each time.

```java
int i = 0;
int val = 1;
while (i <= n) {
    System.out.println(i + " " + val);
    i = i + 1;
    val = 2 * val;
}
```

<table>
<thead>
<tr>
<th>i</th>
<th>val</th>
<th>$i \leq n$</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>true</td>
<td>0 1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>true</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>true</td>
<td>8</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>true</td>
<td>16</td>
</tr>
<tr>
<td>5</td>
<td>32</td>
<td>true</td>
<td>32</td>
</tr>
<tr>
<td>6</td>
<td>64</td>
<td>true</td>
<td>64</td>
</tr>
<tr>
<td>7</td>
<td>128</td>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>
```

Start Again
While Loop: Powers of Two

Print powers of 2 that are \( \leq 2^n \) for some \( n \). Set \( n = 6 \).

- Increment loop counter \( i \) by 1, from 0 to \( n \).
- Double \( \texttt{val} \) each time.

```java
int i = 0;
int val = 1;
while (i <= n) {
    System.out.println(i + " " + val);
    i = i + 1;
    val = 2 * val;
}
```

<table>
<thead>
<tr>
<th>i</th>
<th>val</th>
<th>i ( \leq n )</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>true</td>
<td>0 1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>true</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>true</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>true</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>true</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>32</td>
<td>true</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>64</td>
<td>true</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>128</td>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

Start Again
While Loop: Powers of Two

Print powers of 2 that are \( \leq 2^n \) for some \( n \). Set \( n = 6 \).

- Increment loop counter \( i \) by 1, from 0 to \( n \).
- Double \( \text{val} \) each time.

```java
int i = 0;
int val = 1;
while (i <= n) {
    System.out.println(i + " " + val);
    i = i + 1;
    val = 2 * val;
}
```

<table>
<thead>
<tr>
<th>i</th>
<th>val</th>
<th>i ( \leq ) n</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>true</td>
<td>0 1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>true</td>
<td>1 2</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>true</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>true</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>true</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>32</td>
<td>true</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>64</td>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

- Start Again
Print powers of 2 that are $\leq 2^n$ for some $n$. Set $n = 6$.

- Increment loop counter $i$ by 1, from 0 to $n$.
- Double $val$ each time.

```java
int i = 0;
int val = 1;
while (i <= n) {
    System.out.println(i + " " + val);
    i = i + 1;
    val = 2 * val;
}
```

<table>
<thead>
<tr>
<th>$i$</th>
<th>$val$</th>
<th>$i \leq n$</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>true</td>
<td>0 1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>true</td>
<td>1 2</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
While Loop: Powers of Two

Print powers of 2 that are $\leq 2^n$ for some $n$. Set $n = 6$.

- Increment loop counter $i$ by 1, from 0 to $n$.
- Double $val$ each time.

```java
int i = 0;
int val = 1;
while (i <= n) {
    System.out.println(i + " " + val);
    i = i + 1;
    val = 2 * val;
}
```

<table>
<thead>
<tr>
<th>i</th>
<th>val</th>
<th>$i \leq n$</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>true</td>
<td>0 1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>true</td>
<td>1 2</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Start Again
While Loop: Powers of Two

Print powers of 2 that are $\leq 2^n$ for some $n$. Set $n = 6$.

- Increment loop counter $i$ by 1, from 0 to $n$.
- Double $val$ each time.

```java
int i = 0;
int val = 1;
while (i <= n) {
    System.out.println(i + " " + val);
    i = i + 1;
    val = 2 * val;
}
```

<table>
<thead>
<tr>
<th>i</th>
<th>val</th>
<th>$i \leq n$</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>true</td>
<td>0 1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>true</td>
<td>1 2</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>true</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Start Again
While Loop: Powers of Two

Print powers of 2 that are $\leq 2^n$ for some $n$. Set $n = 6$.

- Increment loop counter $i$ by 1, from 0 to $n$.
- Double $val$ each time.

```java
int i = 0;
int val = 1;
while (i <= n) {
    System.out.println(i + " " + val);
    i = i + 1;
    val = 2 * val;
}
```

<table>
<thead>
<tr>
<th>i</th>
<th>val</th>
<th>i $\leq$ n</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>true</td>
<td>0 1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>true</td>
<td>1 2</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>true</td>
<td>2 4</td>
</tr>
</tbody>
</table>
While Loop: Powers of Two

Print powers of 2 that are $\leq 2^n$ for some $n$. Set $n = 6$.

- Increment loop counter $i$ by 1, from 0 to $n$.
- Double $val$ each time.

```java
int i = 0;
int val = 1;
while (i <= n) {
    System.out.println(i + " " + val);
    i = i + 1;
    val = 2 * val;
}
```

<table>
<thead>
<tr>
<th>i</th>
<th>val</th>
<th>i ≤ n</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>true</td>
<td>0 1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>true</td>
<td>1 2</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>true</td>
<td>2 4</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Start Again
While Loop: Powers of Two

Print powers of 2 that are \( \leq 2^n \) for some \( n \). Set \( n = 6 \).

- Increment loop counter \( i \) by 1, from 0 to \( n \).
- Double \( \text{val} \) each time.

```java
int i = 0;
int val = 1;
while (i <= n) {
    System.out.println(i + " " + val);
    i = i + 1;
    val = 2 * val;
}
```

<table>
<thead>
<tr>
<th>i</th>
<th>val</th>
<th>i ( \leq ) n</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>true</td>
<td>0 1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>true</td>
<td>1 2</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>true</td>
<td>2 4</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

Start Again
While Loop: Powers of Two

Print powers of 2 that are $\leq 2^n$ for some $n$. Set $n = 6$.

- Increment loop counter $i$ by 1, from 0 to $n$.
- Double $val$ each time.

```java
int i = 0;
int val = 1;
while (i <= n) {
    System.out.println(i + " " + val);
    i = i + 1;
    val = 2 * val;
}
```

<table>
<thead>
<tr>
<th>i</th>
<th>val</th>
<th>i $\leq$ n</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>true</td>
<td>0 1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>true</td>
<td>1 2</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>true</td>
<td>2 4</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>
While Loop: Powers of Two

Print powers of 2 that are $\leq 2^n$ for some $n$. Set $n = 6$.

- Increment loop counter $i$ by 1, from 0 to $n$.
- Double `val` each time.

```java
int i = 0;
int val = 1;
while (i <= n) {
    System.out.println(i + " " + val);
    i = i + 1;
    val = 2 * val;
}
```

<table>
<thead>
<tr>
<th>i</th>
<th>val</th>
<th>i ≤ n</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>true</td>
<td>0 1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>true</td>
<td>1 2</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>true</td>
<td>2 4</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>true</td>
<td>3 8</td>
</tr>
</tbody>
</table>

Start Again
Print powers of 2 that are $\leq 2^n$ for some $n$. Set $n = 6$.

- Increment loop counter $i$ by 1, from 0 to $n$.
- Double $val$ each time.

```java
int i = 0;
int val = 1;
while (i <= n) {
    System.out.println(i + " " + val);
    i = i + 1;
    val = 2 * val;
}
```

---

<table>
<thead>
<tr>
<th>i</th>
<th>val</th>
<th>$i \leq n$</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>true</td>
<td>0 1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>true</td>
<td>1 2</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>true</td>
<td>2 4</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>true</td>
<td>3 8</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>
While Loop: Powers of Two

Print powers of 2 that are $\leq 2^n$ for some $n$. Set $n = 6$.

- Increment loop counter $i$ by 1, from 0 to $n$.
- Double $val$ each time.

```java
int i = 0;
int val = 1;
while (i <= n) {
    System.out.println(i + " " + val);
    i = i + 1;
    val = 2 * val;
}
```

<table>
<thead>
<tr>
<th>i</th>
<th>val</th>
<th>i &lt;= n</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>true</td>
<td>0 1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>true</td>
<td>1 2</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>true</td>
<td>2 4</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>true</td>
<td>3 8</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>

Start Again
Print powers of 2 that are $\leq 2^n$ for some $n$. Set $n = 6$.

- Increment loop counter $i$ by 1, from 0 to $n$.
- Double $val$ each time.

```java
int i = 0;
int val = 1;
while (i <= n) {
    System.out.println(i + " " + val);
    i = i + 1;
    val = 2 * val;
}
```

<table>
<thead>
<tr>
<th>i</th>
<th>val</th>
<th>i \leq n</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>true</td>
<td>0 1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>true</td>
<td>1 2</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>true</td>
<td>2 4</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>true</td>
<td>3 8</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>

Start Again
While Loop: Powers of Two

Print powers of 2 that are $\leq 2^n$ for some $n$. Set $n = 6$.

- Increment loop counter $i$ by 1, from 0 to $n$.
- Double $\text{val}$ each time.

```java
int i = 0;
int val = 1;
while (i <= n) {
    System.out.println(i + " " + val);
    i = i + 1;
    val = 2 * val;
}
```

<table>
<thead>
<tr>
<th>$i$</th>
<th>$\text{val}$</th>
<th>$i \leq n$</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>true</td>
<td>0 1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>true</td>
<td>1 2</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>true</td>
<td>2 4</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>true</td>
<td>3 8</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>true</td>
<td>4 16</td>
</tr>
</tbody>
</table>

Start Again
Print powers of 2 that are \( \leq 2^n \) for some \( n \). Set \( n = 6 \).

- Increment loop counter \( i \) by 1, from 0 to \( n \).
- Double \( \text{val} \) each time.

```java
int i = 0;
int val = 1;
while (i <= n) {
    System.out.println(i + " " + val);
    i = i + 1;
    val = 2 * val;
}
```

<table>
<thead>
<tr>
<th>i</th>
<th>val</th>
<th>i ( \leq n )</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>true</td>
<td>0 1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>true</td>
<td>1 2</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>true</td>
<td>2 4</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>true</td>
<td>3 8</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>true</td>
<td>4 16</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
While Loop: Powers of Two

Print powers of 2 that are \( \leq 2^n \) for some \( n \). Set \( n = 6 \).

- Increment loop counter \( i \) by 1, from 0 to \( n \).
- Double \( \text{val} \) each time.

```java
int i = 0;
int val = 1;
while (i <= n) {
    System.out.println(i + " " + val);
    i = i + 1;
    val = 2 * val;
}
```

<table>
<thead>
<tr>
<th>i</th>
<th>val</th>
<th>i ( \leq ) n</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>true</td>
<td>0 1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>true</td>
<td>1 2</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>true</td>
<td>2 4</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>true</td>
<td>3 8</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>true</td>
<td>4 16</td>
</tr>
<tr>
<td>5</td>
<td>32</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Start Again
While Loop: Powers of Two

Print powers of 2 that are $\leq 2^n$ for some $n$. Set $n = 6$.

- Increment loop counter $i$ by 1, from 0 to $n$.
- Double $val$ each time.

```java
int i = 0;
int val = 1;
while (i <= n) {
    System.out.println(i + " " + val);
    i = i + 1;
    val = 2 * val;
}
```

<table>
<thead>
<tr>
<th>i</th>
<th>val</th>
<th>i $\leq$ n</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>true</td>
<td>0 1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>true</td>
<td>1 2</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>true</td>
<td>2 4</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>true</td>
<td>3 8</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>true</td>
<td>4 16</td>
</tr>
<tr>
<td>5</td>
<td>32</td>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>

Start Again
While Loop: Powers of Two

Print powers of 2 that are $\leq 2^n$ for some $n$. Set $n = 6$.

- Increment loop counter $i$ by 1, from 0 to $n$.
- Double $val$ each time.

```java
int i = 0;
int val = 1;
while (i <= n) {
    System.out.println(i + " " + val);
    i = i + 1;
    val = 2 * val;
}
```

<table>
<thead>
<tr>
<th>i</th>
<th>val</th>
<th>i $\leq$ n</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>true</td>
<td>0 1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>true</td>
<td>1 2</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>true</td>
<td>2 4</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>true</td>
<td>3 8</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>true</td>
<td>4 16</td>
</tr>
<tr>
<td>5</td>
<td>32</td>
<td>true</td>
<td>5 32</td>
</tr>
</tbody>
</table>

Start Again
While Loop: Powers of Two

Print powers of 2 that are \( \leq 2^n \) for some \( n \). Set \( n = 6 \).

- Increment loop counter \( i \) by 1, from 0 to \( n \).
- Double \( \text{val} \) each time.

```java
int i = 0;
int val = 1;
while (i <= n) {
    System.out.println(i + " " + val);
    i = i + 1;
    val = 2 * val;
}
```

<table>
<thead>
<tr>
<th>i</th>
<th>val</th>
<th>i ( \leq n )</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>true</td>
<td>0 1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>true</td>
<td>1 2</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>true</td>
<td>2 4</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>true</td>
<td>3 8</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>true</td>
<td>4 16</td>
</tr>
<tr>
<td>5</td>
<td>32</td>
<td>true</td>
<td>5 32</td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
While Loop: Powers of Two

Print powers of 2 that are $\leq 2^n$ for some $n$. Set $n = 6$.
- Increment loop counter $i$ by 1, from 0 to $n$.
- Double $\text{val}$ each time.

```java
int i = 0;
int val = 1;
while (i <= n) {
    System.out.println(i + " " + val);
    i = i + 1;
    val = 2 * val;
}
```

<table>
<thead>
<tr>
<th>i</th>
<th>val</th>
<th>$i \leq n$</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>true</td>
<td>0 1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>true</td>
<td>1 2</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>true</td>
<td>2 4</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>true</td>
<td>3 8</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>true</td>
<td>4 16</td>
</tr>
<tr>
<td>5</td>
<td>32</td>
<td>true</td>
<td>5 32</td>
</tr>
<tr>
<td>6</td>
<td>64</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
While Loop: Powers of Two

Print powers of 2 that are \( \leq 2^n \) for some \( n \). Set \( n = 6 \).

- Increment loop counter \( i \) by 1, from 0 to \( n \).
- Double \( \text{val} \) each time.

```java
int i = 0;
int val = 1;
while (i <= n) {
    System.out.println(i + " " + val);
    i = i + 1;
    val = 2 * val;
}
```

<table>
<thead>
<tr>
<th>i</th>
<th>val</th>
<th>i ( \leq n )</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>true</td>
<td>0 1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>true</td>
<td>1 2</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>true</td>
<td>2 4</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>true</td>
<td>3 8</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>true</td>
<td>4 16</td>
</tr>
<tr>
<td>5</td>
<td>32</td>
<td>true</td>
<td>5 32</td>
</tr>
<tr>
<td>6</td>
<td>64</td>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>
```

Start Again
While Loop: Powers of Two

Print powers of 2 that are $\leq 2^n$ for some $n$. Set $n = 6$.

- Increment loop counter $i$ by 1, from 0 to $n$.
- Double $val$ each time.

```java
int i = 0;
int val = 1;
while (i <= n) {
    System.out.println(i + " " + val);
    i = i + 1;
    val = 2 * val;
}
```

<table>
<thead>
<tr>
<th>i</th>
<th>val</th>
<th>i ≤ n</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>true</td>
<td>0 1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>true</td>
<td>1 2</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>true</td>
<td>2 4</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>true</td>
<td>3 8</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>true</td>
<td>4 16</td>
</tr>
<tr>
<td>5</td>
<td>32</td>
<td>true</td>
<td>5 32</td>
</tr>
<tr>
<td>6</td>
<td>64</td>
<td>true</td>
<td>6 64</td>
</tr>
</tbody>
</table>

Start Again
While Loop: Powers of Two

Print powers of 2 that are $\leq 2^n$ for some $n$. Set $n = 6$.

- Increment loop counter $i$ by 1, from 0 to $n$.
- Double $val$ each time.

```java
int i = 0;
int val = 1;
while (i <= n) {
    System.out.println(i + " " + val);
    i = i + 1;
    val = 2 * val;
}
```

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>i</td>
<td>val</td>
<td>$i \leq n$</td>
<td>Output</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>true</td>
<td>0 1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>true</td>
<td>1 2</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>true</td>
<td>2 4</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>true</td>
<td>3 8</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>true</td>
<td>4 16</td>
</tr>
<tr>
<td>5</td>
<td>32</td>
<td>true</td>
<td>5 32</td>
</tr>
<tr>
<td>6</td>
<td>64</td>
<td>true</td>
<td>6 64</td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
While Loop: Powers of Two

Print powers of 2 that are $\leq 2^n$ for some $n$. Set $n = 6$.

- Increment loop counter $i$ by 1, from 0 to $n$.
- Double $\texttt{val}$ each time.

```java
int i = 0;
int val = 1;
while (i <= n) {
    System.out.println(i + " " + val);
    i = i + 1;
    val = 2 * val;
}
```

<table>
<thead>
<tr>
<th>$i$</th>
<th>$\text{val}$</th>
<th>$i \leq n$</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>true</td>
<td>0 1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>true</td>
<td>1 2</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>true</td>
<td>2 4</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>true</td>
<td>3 8</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>true</td>
<td>4 16</td>
</tr>
<tr>
<td>5</td>
<td>32</td>
<td>true</td>
<td>5 32</td>
</tr>
<tr>
<td>6</td>
<td>64</td>
<td>true</td>
<td>6 64</td>
</tr>
<tr>
<td>7</td>
<td>128</td>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

Start Again
While Loop: Powers of Two

Print powers of 2 that are \( \leq 2^n \) for some \( n \). Set \( n = 6 \).

- Increment loop counter \( i \) by 1, from 0 to \( n \).
- Double \( \text{val} \) each time.

```java
int i = 0;
int val = 1;
while (i <= n) {
    System.out.println(i + " " + val);
    i = i + 1;
    val = 2 * val;
}
```

<table>
<thead>
<tr>
<th>i</th>
<th>val</th>
<th>i ≤ n</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>true</td>
<td>0 1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>true</td>
<td>1 2</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>true</td>
<td>2 4</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>true</td>
<td>3 8</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>true</td>
<td>4 16</td>
</tr>
<tr>
<td>5</td>
<td>32</td>
<td>true</td>
<td>5 32</td>
</tr>
<tr>
<td>6</td>
<td>64</td>
<td>true</td>
<td>6 64</td>
</tr>
<tr>
<td>7</td>
<td>128</td>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

Start Again
```java
public class PowersOfTwo {
    public static void main(String[] args) {
        int n = Integer.parseInt(args[0]);
        int i = 0;
        int val = 1;
        while (i <= n) {
            System.out.println(i + " " + val);
            i = i + 1;
            val = 2 * val;
        }
    }
}
```

```
% java PowersOfTwo 3
0 1
1 2
2 4
3 8
```
Q: Is anything wrong with the following version of PowersOfTwo?

```java
int i = 0;
int val = 1;
while (i <= n)
    System.out.println(i + " " + val);
    i = i + 1;
    val = 2 * val;
```
Q: Is anything wrong with the following version of PowersOfTwo?

```java
int i = 0;
int val = 1;
while (i <= n)
    System.out.println(i + " " + val);
i = i + 1;
val = 2 * val;
```

A: Need curly braces around statements in while loop. Otherwise, only the first of the statements is executed before returning to while condition; enters an infinite loop, printing 0 1 for ever.

(How to stop an infinite loop? At the Linux command-line, hit Control-c.)
The Increment Operator

```java
int i = 0;
int val = 1;
while (i <= n) {
    System.out.println(i + " " + val);
    i = i + 1;
    val = 2 * val;
}
```
The Increment Operator

```java
int i = 0;
int val = 1;
while (i <= n) {
    System.out.println(i + " " + val);
    i = i + 1;
    val = 2 * val;
}
```

- standard assignment: \(i = i + 1;\)
- semantically equivalent shorthand: \(i++;\)
The Increment Operator

```java
int i = 0;
int val = 1;
while (i <= n) {
    System.out.println(i + " " + val);
    i = i + 1;
    val = 2 * val;
}
```

- standard assignment: `i = i + 1;`
- semantically equivalent shorthand: `i++;`

```java
int i = 0;
int val = 1;
while (i <= n) {
    System.out.println(i + " " + val);
    i++;
    val = 2 * val;
}
```
Loops (For)
The for loop is another common structure for repeating things.

- Execute initialization statement.
- Evaluate a boolean expression.
- If true, execute some statements.
- Then execute the increment statement.
- Repeat.
For Loop

The `for` loop is another common structure for repeating things.

- Execute initialization statement.
- Evaluate a boolean expression.
- If true, execute some statements.
- Then execute the increment statement.
- Repeat.

```java
for (init; boolean expression; increment) {
    statement 1;
    statement 2;
}
```
The for loop is another common structure for repeating things.

- Execute initialization statement.
- Evaluate a boolean expression.
- If true, execute some statements.
- Then execute the increment statement.
- Repeat.

```java
for (init; boolean expression; increment) {
    statement 1;
    statement 2;
}
```
Anatomy of a For Loop

- **int val = 1;**
- **for (int i = 0; i <= N; i++ ) {**
  - **System.out.println(i + " " + val);**
  - **val = 2 * val;**
- **}**

**loop body**

**loop continuation condition**

**loop control variable**

**increment loop variable**

**declare and initialize a loop control variable**

**initialize another variable in a separate statement**
For Loop: Powers of Two

Print the first $n$ powers of 2. Set $n = 6$.

- Double `val` each time.

```java
int val = 1;
for (int i = 0; i <= n; i++) {
    System.out.println(i + " " + val);
    val = 2 * val;
}
```

Output:

```
0 1
1 2
2 4
3 8
4 16
5 32
6 64
7 128
```

Start Again
For Loop: Powers of Two

Print the first $n$ powers of 2. Set $n = 6$.

- Double $\text{val}$ each time.

```
int val = 1;
for (int i = 0; i <= n; i++) {
    System.out.println(i + " " + val);
    val = 2 * val;
}
```

<table>
<thead>
<tr>
<th>i</th>
<th>val</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
</tr>
<tr>
<td>5</td>
<td>32</td>
</tr>
<tr>
<td>6</td>
<td>64</td>
</tr>
</tbody>
</table>

Start Again
For Loop: Powers of Two

Print the first \( n \) powers of 2. Set \( n = 6 \).

- Double \( \text{val} \) each time.

```java
int val = 1;
for (int i = 0; i <= n; i++) {
    System.out.println(i + " " + val);
    val = 2 * val;
}
```

<table>
<thead>
<tr>
<th>val</th>
<th>i</th>
<th>i ( \leq ) n</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>true</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>64</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>128</td>
<td>7</td>
<td>false</td>
</tr>
</tbody>
</table>

Start Again
For Loop: Powers of Two

Print the first $n$ powers of 2. Set $n = 6$.

- Double $val$ each time.

```java
int val = 1;
for (int i = 0; i <= n; i++) {
    System.out.println(i + " " + val);
    val = 2 * val;
}
```

<table>
<thead>
<tr>
<th>val</th>
<th>i</th>
<th>i ≤ n</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>true</td>
<td>0 1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>true</td>
<td>1 2</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>true</td>
<td>2 4</td>
</tr>
<tr>
<td>8</td>
<td>3</td>
<td>true</td>
<td>3 8</td>
</tr>
<tr>
<td>16</td>
<td>4</td>
<td>true</td>
<td>4 16</td>
</tr>
<tr>
<td>32</td>
<td>5</td>
<td>true</td>
<td>5 32</td>
</tr>
<tr>
<td>64</td>
<td>6</td>
<td>true</td>
<td>6 64</td>
</tr>
<tr>
<td>128</td>
<td>7</td>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>
```

Start Again
For Loop: Powers of Two

Print the first $n$ powers of 2. Set $n = 6$.

- Double `val` each time.

```java
int val = 1;
for (int i = 0; i <= n; i++) {
    System.out.println(i + " " + val);
    val = 2 * val;
}
```

<table>
<thead>
<tr>
<th>val</th>
<th>i</th>
<th>i ≤ n</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>true</td>
<td>0 1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Print the first $n$ powers of 2. Set $n = 6$.

- Double `val` each time.

```java
int val = 1;
for (int i = 0; i <= n; i++) {
    System.out.println(i + " " + val);
    val = 2 * val;
}
```

<table>
<thead>
<tr>
<th>val</th>
<th>i</th>
<th>$i \leq n$</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>true</td>
<td>0 1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
For Loop: Powers of Two

Print the first $n$ powers of 2. Set $n = 6$.

- Double $val$ each time.

```java
int val = 1;
for (int i = 0; i <= n; i++) {
    System.out.println(i + " " + val);
    val = 2 * val;
}
```

<table>
<thead>
<tr>
<th>val</th>
<th>i</th>
<th>$i \leq n$</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>true</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>true</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>true</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>3</td>
<td>true</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>4</td>
<td>true</td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>5</td>
<td>true</td>
<td></td>
</tr>
<tr>
<td>64</td>
<td>6</td>
<td>true</td>
<td></td>
</tr>
<tr>
<td>128</td>
<td>7</td>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

Start Again
For Loop: Powers of Two

Print the first $n$ powers of 2. Set $n = 6$.

- Double `val` each time.

```java
int val = 1;
for (int i = 0; i <= n; i++) {
    System.out.println(i + " " + val);
    val = 2 * val;
}
```

<table>
<thead>
<tr>
<th>val</th>
<th>i</th>
<th>i ≤ n</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>true</td>
<td>0 1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>true</td>
<td>1 2</td>
</tr>
</tbody>
</table>

Start Again
For Loop: Powers of Two

Print the first $n$ powers of 2. Set $n = 6$.

- Double `val` each time.

```java
int val = 1;
for (int i = 0; i <= n; i++) {
    System.out.println(i + " " + val);
    val = 2 * val;
}
```

<table>
<thead>
<tr>
<th>val</th>
<th>i</th>
<th>i ≤ n</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>true</td>
<td>0 1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>true</td>
<td>1 2</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>true</td>
<td>1 2</td>
</tr>
</tbody>
</table>
For Loop: Powers of Two

Print the first $n$ powers of 2. Set $n = 6$.

- Double $val$ each time.

```java
int val = 1;
for (int i = 0; i <= n; i++) {
    System.out.println(i + " " + val);
    val = 2 * val;
}
```

<table>
<thead>
<tr>
<th>val</th>
<th>i</th>
<th>$i \leq n$</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>true</td>
<td>0 1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>true</td>
<td>1 2</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
For Loop: Powers of Two

Print the first \( n \) powers of 2. Set \( n = 6 \).

- Double \( \text{val} \) each time.

```java
int val = 1;
for (int i = 0; i <= n; i++) {
    System.out.println(i + " " + val);
    val = 2 * val;
}
```

<table>
<thead>
<tr>
<th>val</th>
<th>i</th>
<th>( i \leq n )</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>true</td>
<td>0 1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>true</td>
<td>1 2</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>

Start Again
For Loop: Powers of Two

Print the first \( n \) powers of 2. Set \( n = 6 \).

- Double \( \text{val} \) each time.

```java
int val = 1;
for (int i = 0; i <= n; i++) {
    System.out.println(i + " " + val);
    val = 2 * val;
}
```

<table>
<thead>
<tr>
<th>val</th>
<th>i</th>
<th>( i \leq n )</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>true</td>
<td>0 1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>true</td>
<td>1 2</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>true</td>
<td>2 4</td>
</tr>
</tbody>
</table>
Print the first $n$ powers of 2. Set $n = 6$.

- Double `val` each time.

```java
int val = 1;
for (int i = 0; i <= n; i++) {
    System.out.println(i + " " + val);
    val = 2 * val;
}
```

<table>
<thead>
<tr>
<th>val</th>
<th>i</th>
<th>i ≤ n</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>true</td>
<td>0 1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>true</td>
<td>1 2</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>true</td>
<td>2 4</td>
</tr>
<tr>
<td>8</td>
<td>3</td>
<td>false</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>true</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>true</td>
<td></td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>true</td>
<td></td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

Start Again
Print the first $n$ powers of 2. Set $n = 6$.

- Double `val` each time.

```java
int val = 1;
for (int i = 0; i <= n; i++) {
    System.out.println(i + " " + val);
    val = 2 * val;
}
```

<table>
<thead>
<tr>
<th>val</th>
<th>i</th>
<th>i $\leq$ n</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>true</td>
<td>0 1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>true</td>
<td>1 2</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>true</td>
<td>2 4</td>
</tr>
<tr>
<td>8</td>
<td>3</td>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>
For Loop: Powers of Two

Print the first $n$ powers of 2. Set $n = 6$.

- Double $val$ each time.

```java
int val = 1;
for (int i = 0; i <= n; i++) {
    System.out.println(i + " " + val);
    val = 2 * val;
}
```

<table>
<thead>
<tr>
<th>val</th>
<th>i</th>
<th>i ≤ n</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>true</td>
<td>0 1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>true</td>
<td>1 2</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>true</td>
<td>2 4</td>
</tr>
<tr>
<td>8</td>
<td>3</td>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>

Start Again
Print the first $n$ powers of 2. Set $n = 6$.

- Double $val$ each time.

```java
int val = 1;
for (int i = 0; i <= n; i++) {
    System.out.println(i + " " + val);
    val = 2 * val;
}
```

Output:

<table>
<thead>
<tr>
<th>val</th>
<th>i</th>
<th>$i \leq n$</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>true</td>
<td>0 1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>true</td>
<td>1 2</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>true</td>
<td>2 4</td>
</tr>
<tr>
<td>8</td>
<td>3</td>
<td>true</td>
<td>3 8</td>
</tr>
</tbody>
</table>
For Loop: Powers of Two

Print the first $n$ powers of 2. Set $n = 6$.

- Double $val$ each time.

```java
int val = 1;
for (int i = 0; i <= n; i++) {
    System.out.println(i + " " + val);
    val = 2 * val;
}
```

<table>
<thead>
<tr>
<th>val</th>
<th>i</th>
<th>$i \leq n$</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>true</td>
<td>0 1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>true</td>
<td>1 2</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>true</td>
<td>2 4</td>
</tr>
<tr>
<td>8</td>
<td>3</td>
<td>true</td>
<td>3 8</td>
</tr>
<tr>
<td>16</td>
<td>6</td>
<td>false</td>
<td>16</td>
</tr>
</tbody>
</table>
For Loop: Powers of Two

Print the first $n$ powers of 2. Set $n = 6$.

- Double `val` each time.

```java
int val = 1;
for (int i = 0; i <= n; i++) {
    System.out.println(i + " " + val);
    val = 2 * val;
}
```

<table>
<thead>
<tr>
<th>val</th>
<th>i</th>
<th>i ≤ n</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>true</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>true</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>true</td>
<td>2</td>
</tr>
<tr>
<td>8</td>
<td>3</td>
<td>true</td>
<td>3</td>
</tr>
<tr>
<td>16</td>
<td>4</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Print the first $n$ powers of 2. Set $n = 6$.

- Double `val` each time.

```java
int val = 1;
for (int i = 0; i <= n; i++) {
    System.out.println(i + " " + val);
    val = 2 * val;
}
```

<table>
<thead>
<tr>
<th>val</th>
<th>i</th>
<th>i ≤ n</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>true</td>
<td>0 1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>true</td>
<td>1 2</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>true</td>
<td>2 4</td>
</tr>
<tr>
<td>8</td>
<td>3</td>
<td>true</td>
<td>3 8</td>
</tr>
<tr>
<td>16</td>
<td>4</td>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>
For Loop: Powers of Two

Print the first $n$ powers of 2. Set $n = 6$.

- Double $val$ each time.

```
int val = 1;
for (int i = 0; i <= n; i++) {
    System.out.println(i + " " + val);
    val = 2 * val;
}
```

<table>
<thead>
<tr>
<th>val</th>
<th>i</th>
<th>$i \leq n$</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>true</td>
<td>0 1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>true</td>
<td>1 2</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>true</td>
<td>2 4</td>
</tr>
<tr>
<td>8</td>
<td>3</td>
<td>true</td>
<td>3 8</td>
</tr>
<tr>
<td>16</td>
<td>4</td>
<td>true</td>
<td>4 16</td>
</tr>
</tbody>
</table>
For Loop: Powers of Two

Print the first \( n \) powers of 2. Set \( n = 6 \).

- Double \( \text{val} \) each time.

```java
int val = 1;
for (int i = 0; i <= n; i++) {
    System.out.println(i + " " + val);
    val = 2 * val;
}
```

<table>
<thead>
<tr>
<th>val</th>
<th>i</th>
<th>i ( \leq ) n</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>true</td>
<td>0 1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>true</td>
<td>1 2</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>true</td>
<td>2 4</td>
</tr>
<tr>
<td>8</td>
<td>3</td>
<td>true</td>
<td>3 8</td>
</tr>
<tr>
<td>16</td>
<td>4</td>
<td>true</td>
<td>4 16</td>
</tr>
<tr>
<td>32</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Start Again
For Loop: Powers of Two

Print the first $n$ powers of 2. Set $n = 6$.

- Double $val$ each time.

```java
int val = 1;
for (int i = 0; i <= n; i++) {
    System.out.println(i + " " + val);
    val = 2 * val;
}
```

<table>
<thead>
<tr>
<th>val</th>
<th>i</th>
<th>$i \leq n$</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>true</td>
<td>0 1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>true</td>
<td>1 2</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>true</td>
<td>2 4</td>
</tr>
<tr>
<td>8</td>
<td>3</td>
<td>true</td>
<td>3 8</td>
</tr>
<tr>
<td>16</td>
<td>4</td>
<td>true</td>
<td>4 16</td>
</tr>
<tr>
<td>32</td>
<td>5</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Print the first $n$ powers of 2. Set $n = 6$.

- Double `val` each time.

```java
int val = 1;
for (int i = 0; i <= n; i++) {
    System.out.println(i + " " + val);
    val = 2 * val;
}
```

<table>
<thead>
<tr>
<th>val</th>
<th>i</th>
<th>i ≤ n</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>true</td>
<td>0 1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>true</td>
<td>1 2</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>true</td>
<td>2 4</td>
</tr>
<tr>
<td>8</td>
<td>3</td>
<td>true</td>
<td>3 8</td>
</tr>
<tr>
<td>16</td>
<td>4</td>
<td>true</td>
<td>4 16</td>
</tr>
<tr>
<td>32</td>
<td>5</td>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>

Start Again
For Loop: Powers of Two

Print the first $n$ powers of 2. Set $n = 6$.

- Double $val$ each time.

```java
int val = 1;
for (int i = 0; i <= n; i++) {
    System.out.println(i + " " + val);
    val = 2 * val;
}
```

<table>
<thead>
<tr>
<th>val</th>
<th>i</th>
<th>i ≤ n</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>true</td>
<td>0 1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>true</td>
<td>1 2</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>true</td>
<td>2 4</td>
</tr>
<tr>
<td>8</td>
<td>3</td>
<td>true</td>
<td>3 8</td>
</tr>
<tr>
<td>16</td>
<td>4</td>
<td>true</td>
<td>4 16</td>
</tr>
<tr>
<td>32</td>
<td>5</td>
<td>true</td>
<td>5 32</td>
</tr>
</tbody>
</table>

Start Again
Print the first $n$ powers of 2. Set $n = 6$.

- Double `val` each time.

```java
int val = 1;
for (int i = 0; i <= n; i++) {
    System.out.println(i + " " + val);
    val = 2 * val;
}
```

<table>
<thead>
<tr>
<th>val</th>
<th>i</th>
<th>i ≤ n</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>true</td>
<td>0 1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>true</td>
<td>1 2</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>true</td>
<td>2 4</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>true</td>
<td>3 8</td>
</tr>
<tr>
<td>8</td>
<td>4</td>
<td>true</td>
<td>4 16</td>
</tr>
<tr>
<td>16</td>
<td>5</td>
<td>true</td>
<td>5 32</td>
</tr>
<tr>
<td>32</td>
<td>6</td>
<td>false</td>
<td></td>
</tr>
<tr>
<td>64</td>
<td>7</td>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

Start Again
For Loop: Powers of Two

Print the first \( n \) powers of 2. Set \( n = 6 \).

- Double \( \text{val} \) each time.

```java
int val = 1;
for (int i = 0; i <= n; i++) {
    System.out.println(i + " " + val);
    val = 2 * val;
}
```

<table>
<thead>
<tr>
<th>val</th>
<th>i</th>
<th>( i \leq n )</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>true</td>
<td>0 1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>true</td>
<td>1 2</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>true</td>
<td>2 4</td>
</tr>
<tr>
<td>8</td>
<td>3</td>
<td>true</td>
<td>3 8</td>
</tr>
<tr>
<td>16</td>
<td>4</td>
<td>true</td>
<td>4 16</td>
</tr>
<tr>
<td>32</td>
<td>5</td>
<td>true</td>
<td>5 32</td>
</tr>
<tr>
<td>64</td>
<td>6</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Start Again
Print the first \( n \) powers of 2. Set \( n = 6 \).

- Double `val` each time.

```java
int val = 1;
for (int i = 0; i <= n; i++) {
    System.out.println(i + " " + val);
    val = 2 * val;
}
```

<table>
<thead>
<tr>
<th>val</th>
<th>i</th>
<th>( i \leq n )</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>true</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>true</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>true</td>
<td>2</td>
</tr>
<tr>
<td>8</td>
<td>3</td>
<td>true</td>
<td>3</td>
</tr>
<tr>
<td>16</td>
<td>4</td>
<td>true</td>
<td>4</td>
</tr>
<tr>
<td>32</td>
<td>5</td>
<td>true</td>
<td>5</td>
</tr>
<tr>
<td>64</td>
<td>6</td>
<td>true</td>
<td>64</td>
</tr>
</tbody>
</table>
```

Start Again
For Loop: Powers of Two

Print the first $n$ powers of 2. Set $n = 6$.
- Double $val$ each time.

```java
int val = 1;
for (int i = 0; i <= n; i++) {
    System.out.println(i + " " + val);
    val = 2 * val;
}
```

<table>
<thead>
<tr>
<th>val</th>
<th>i</th>
<th>i ≤ n</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>true</td>
<td>0 1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>true</td>
<td>1 2</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>true</td>
<td>2 4</td>
</tr>
<tr>
<td>8</td>
<td>3</td>
<td>true</td>
<td>3 8</td>
</tr>
<tr>
<td>16</td>
<td>4</td>
<td>true</td>
<td>4 16</td>
</tr>
<tr>
<td>32</td>
<td>5</td>
<td>true</td>
<td>5 32</td>
</tr>
<tr>
<td>64</td>
<td>6</td>
<td>true</td>
<td>6 64</td>
</tr>
</tbody>
</table>

Start Again
For Loop: Powers of Two

Print the first \( n \) powers of 2. Set \( n = 6 \).

- Double \( \text{val} \) each time.

```java
int val = 1;
for (int i = 0; i <= n; i++) {
    System.out.println(i + " " + val);
    val = 2 * val;
}
```

<table>
<thead>
<tr>
<th>val</th>
<th>i</th>
<th>( i \leq n )</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>true</td>
<td>0 1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>true</td>
<td>1 2</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>true</td>
<td>2 4</td>
</tr>
<tr>
<td>8</td>
<td>3</td>
<td>true</td>
<td>3 8</td>
</tr>
<tr>
<td>16</td>
<td>4</td>
<td>true</td>
<td>4 16</td>
</tr>
<tr>
<td>32</td>
<td>5</td>
<td>true</td>
<td>5 32</td>
</tr>
<tr>
<td>64</td>
<td>6</td>
<td>true</td>
<td>6 64</td>
</tr>
<tr>
<td>128</td>
<td>7</td>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

Start Again
Print the first $n$ powers of 2. Set $n = 6$.

- Double $val$ each time.

```java
int val = 1;
for (int i = 0; i <= n; i++) {
    System.out.println(i + " " + val);
    val = 2 * val;
}
```

<table>
<thead>
<tr>
<th>val</th>
<th>i</th>
<th>$i \leq n$</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>true</td>
<td>0 1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>true</td>
<td>1 2</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>true</td>
<td>2 4</td>
</tr>
<tr>
<td>8</td>
<td>3</td>
<td>true</td>
<td>3 8</td>
</tr>
<tr>
<td>16</td>
<td>4</td>
<td>true</td>
<td>4 16</td>
</tr>
<tr>
<td>32</td>
<td>5</td>
<td>true</td>
<td>5 32</td>
</tr>
<tr>
<td>64</td>
<td>6</td>
<td>true</td>
<td>6 64</td>
</tr>
<tr>
<td>128</td>
<td>7</td>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

Start Again
Print the first \( n \) powers of 2. Set \( n = 6 \).

- Double \( \text{val} \) each time.

```java
int val = 1;
for (int i = 0; i <= n; i++) {
    System.out.println(i + " " + val);
    val = 2 * val;
}
```

<table>
<thead>
<tr>
<th>( \text{val} )</th>
<th>( i )</th>
<th>( i \leq n )</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>true</td>
<td>0 1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>true</td>
<td>1 2</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>true</td>
<td>2 4</td>
</tr>
<tr>
<td>8</td>
<td>3</td>
<td>true</td>
<td>3 8</td>
</tr>
<tr>
<td>16</td>
<td>4</td>
<td>true</td>
<td>4 16</td>
</tr>
<tr>
<td>32</td>
<td>5</td>
<td>true</td>
<td>5 32</td>
</tr>
<tr>
<td>64</td>
<td>6</td>
<td>true</td>
<td>6 64</td>
</tr>
<tr>
<td>128</td>
<td>7</td>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

Start Again
When to use While and when to use For?

Rule of thumb

- True
  - Know iteration count
    - Use For
    - Use While
- False
Let’s practise that

TOP HAT
Bailing Out Early
What if you need to leave a loop early?

Sometimes you don’t want to end the execution of a loop but instead break out early, e.g. for search algorithms.

`break` allows you to break out of a loop immediately
What if you need to leave a loop early?

Breaking out early

// find first number divisible by n
int start = 50;
int end = 5000;
int n = 344;
for(int i = start; i < end; i++) {
    if (i % n == 0) {
        System.out.println("Number found: " + i);
        break;
    }
    // some complex calculations
}
What if you need to leave a loop early?

At other times, you might want to skip a loop iteration for certain input and continue with the next one, e.g. when processing data and skipping invalid entries.

`continue` allows you to skip the remainder of the loop body and continue with the next iteration.
What if you need to leave a loop early?

Skipping iterations

```java
// skip numbers divisible by n
int start = 0;
int end = 100;
int n = 5;
for (int i = start; i < end; i++) {
    if (i % n == 0) {
        continue;
    }
}

// run some complex calculations
```
A word of caution

It is easy to end up writing complex, hard-to-read, error-prone code using `break` and `continue`.

Keep them for cases like the examples, where the whole loop body is irrelevant for certain values, or after a certain value.

Usually `nested conditionals` are better style, because they make it easier to reason about the circumstances under which a given line is reached...
Nested Conditionals
# Nested If Statements

How to classify Scottish weather:

<table>
<thead>
<tr>
<th>degrees C</th>
<th>verdict</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt; -5</td>
<td>wear a sweater</td>
</tr>
<tr>
<td>-5 to 0</td>
<td>nippy</td>
</tr>
<tr>
<td>1 to 10</td>
<td>normal</td>
</tr>
<tr>
<td>&gt; 10</td>
<td>roastin'</td>
</tr>
</tbody>
</table>

4 mutually exclusive alternatives
Nested If Statements

How to classify Scottish weather:

<table>
<thead>
<tr>
<th>degrees C</th>
<th>verdict</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt; -5</td>
<td>wear a sweater</td>
</tr>
<tr>
<td>-5 to 0</td>
<td>nippy</td>
</tr>
<tr>
<td>1 to 10</td>
<td>normal</td>
</tr>
<tr>
<td>&gt; 10</td>
<td>roastin'</td>
</tr>
</tbody>
</table>

4 mutually exclusive alternatives

String verdict;
if (temp < -5) verdict = "wear a sweater";
else {
    if (temp < 1) verdict = "nippy";
    else {
        if (temp < 11) verdict = "normal";
        else verdict = "roastin'";
    }
}
Nested If Statements

We don’t necessarily need all those braces.

```java
public class ScottishWeather {
    public static void main(String[] args) {
        String verdict;
        int temp = Integer.parseInt(args[0]);
        if (temp < -5) verdict = "wear a sweater";
        else if (temp < 1) verdict = "nippy";
        else if (temp < 11) verdict = "normal";
        else verdict = "roastin'");
        System.out.println("Verdict: " + verdict);
    }
}
```

**Output**

```
% java ScottishWeather -1
Verdict: nippy

% java ScottishWeather 1
Verdict: normal
```
Is there anything wrong with the logic of the following code?

```java
String verdict;
int temp = Integer.parseInt(args[0]);
if (temp < -5) verdict = "wear a sweater";
if (temp < 1)  verdict = "nippy";
if (temp < 11) verdict = "normal";
if (temp >= 11) verdict = "roastin'";
```

<table>
<thead>
<tr>
<th>degrees C</th>
<th>verdict</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt; -5</td>
<td>wear a sweater</td>
</tr>
<tr>
<td>-5 to 0</td>
<td>nippy</td>
</tr>
<tr>
<td>1 to 10</td>
<td>normal</td>
</tr>
<tr>
<td>&gt; 10</td>
<td>roastin'</td>
</tr>
</tbody>
</table>

4 mutually exclusive alternatives
## Summary

Control flow:

- Sequence of statements that are actually executed in a program run.
- Conditionals and loops: enable us to choreograph the control flow.

<table>
<thead>
<tr>
<th>Control Flow</th>
<th>Description</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>straight-line programs</td>
<td>all statements are executed in the order given</td>
<td></td>
</tr>
<tr>
<td>conditionals</td>
<td>certain statements are executed depending on the values of certain variables</td>
<td>if, if-else</td>
</tr>
<tr>
<td>loops</td>
<td>certain statements are executed repeatedly until certain conditions are met</td>
<td>while, for</td>
</tr>
</tbody>
</table>
Java Tutorial
pp68-86, i.e. Chapter 3 Language Basics from Expressions, Statements and Blocks to the end of the chapter.

Objects First
Appendix C.2 - C.3, Appendix D.1 - D.3