The Use of Curly Braces

- The curly braces are **required** if there are **multiple statements** on a branch
- The curly braces are **optional** if there is one **single statement** on a branch
- It is **highly recommended** to **always** use braces, even when not necessary

```
if (<condition>){
    <if controlled statement(s)>;
} else {
    <else controlled statement(s)>;
}
```

Valid, and recommended

```
if (<condition>)
    <if controlled SINGLE statement>;
else
    <else controlled SINGLE statement>;
```

Valid, but NOT recommended

Example – Valid

```java
public class SingleLineIf {
    public static void main(String[] args) {

        int a = 1;

        if(a > 0){
            System.out.println("1: a > 0");
        } else {
            System.out.println("1: a <= 0");
        }

        if(a > 0)
            System.out.println("2: a > 0");
        else
            System.out.println("2: a <= 0");
    }
}
```

```
$ javac SingleLineIf.java
$ java SingleLineIf
1: a > 0
2: a > 0
```

Example – Invalid

```java
public class SingleLineIf {
    public static void main(String[] args) {

        int a = 1;

        if(a > 0) {
            System.out.println("a > 0");}
            System.out.println("a is " + a);
        else
            System.out.println("a <= 0");
    }
}
```

```
$ javac SingleLineIf.java
SingleLineIf.java:9: error: 'else' without 'if'
        else
        ^
1 error
```

```java
public class SingleLineIf {
    public static void main(String[] args) {

        int a = 1;

        if(a > 0)
            System.out.println("a > 0");
        else {
            System.out.println("a <= 0");}
            System.out.println("a is " + a);
    }
}
```

```
$ javac SingleLineIf.java
$ java SingleLineIf
a > 0
a is 1
```

Q: What's the exact output of the following code??

```
int a = 1, b = 2, c = 3, d = 4;
int e = 5, f = 6, g = 7, h = 8;

if (a > b)
    if (c > d)
        e = f;
else
   g = h;

System.out.println(a);
System.out.println(b);
System.out.println(c);
System.out.println(d);
System.out.println(e);
System.out.println(f);
System.out.println(g);
System.out.println(h);
```

```
1
2
3
4
5
6
7
8
```

Q: Why the value of g is not updated?

“Dangling else” Problem

- Every `else`-part is paired with the **nearest unmatched** `if`-part
- Computers ignore the indentation

Original code:

```
if (a > b)
    if (c > d)
        e = f;
else
   g = h;
```

Computers read the code as:

```
if (a > b) {
    if (c > d) {
        e = f;
    } else {
        g = h;
    }
}
```