

COMP 110/L Lecture 11

Kyle Dewey

Outline

- `@Test vs. assertEquals`
- **Boolean operations**
 - `&&`
 - `||`
 - `!`
- **Complex `if` conditions**

@Test **vs.**

assertEquals

@Test vs. assertEquals

- `@Test` defines a test
- `assertEquals` checks a condition
- Can have a `@Test` containing no `assertEquals`
 - Test always passes
- Can have multiple `assertEquals` per `@Test`
 - Test passes if all `assertEquals` are ok

Example:

`MultiAssert.java`

`MultiAssertTest.java`

Boolean Operations

Boolean Operations

You're already familiar with
operations returning `boolean`

Boolean Operations

You're already familiar with
operations returning `boolean`

`3 < 6`

Boolean Operations

You're already familiar with
operations returning `boolean`

`3 < 6`

`2 == 7`

Boolean Operations

You're already familiar with
operations returning `boolean`

`3 < 6`

`2 == 7`

`8 >= 8`

Bigger Expressions

Can chain `boolean` expressions with **AND** (`&&`).

Semantics: only `true` if both sides are `true`.

Bigger Expressions

Can chain `boolean` expressions with **AND** (`&&`).

Semantics: only `true` if both sides are `true`.

`3 > 1 && 1 < 5`

Bigger Expressions

Can chain `boolean` expressions with **AND** (`&&`).

Semantics: only `true` if both sides are `true`.

```
3 > 1 && 1 < 5
```

```
true
```

Bigger Expressions

Can chain `boolean` expressions with **AND** (`&&`).

Semantics: only `true` if both sides are `true`.

```
3 > 1 && 1 < 5
```

```
true
```

```
1 > 3 && 1 < 5
```

Bigger Expressions

Can chain `boolean` expressions with **AND** (`&&`).

Semantics: only `true` if both sides are `true`.

```
3 > 1 && 1 < 5
```

```
true
```

```
1 > 3 && 1 < 5
```

```
false
```

Bigger Expressions

Can chain `boolean` expressions with **AND** (`&&`).

Semantics: only `true` if both sides are `true`.

```
3 > 1 && 1 < 5
```

```
true
```

```
1 > 3 && 1 < 5
```

```
false
```

```
3 > 1 && 5 < 1
```


Bigger Expressions

Can chain `boolean` expressions with **AND** (`&&`).

Semantics: only `true` if both sides are `true`.

```
3 > 1 && 1 < 5
```

```
true
```

```
1 > 3 && 1 < 5
```

```
false
```

```
3 > 1 && 5 < 1
```

```
false
```

Example:
`And.java`

Boolean Or

boolean expressions can also be combined with OR (| |)

Semantics: true if either side is true.

Boolean Or

boolean expressions can also be combined with OR (||)

Semantics: true if either side is true.

`3 > 1 || 5 < 1`

Boolean Or

boolean expressions can also be combined with OR (||)

Semantics: true if either side is true.

```
3 > 1 || 5 < 1  
true
```

Boolean Or

boolean expressions can also be combined with OR (||)

Semantics: true if either side is true.

```
3 > 1 || 5 < 1  
true
```

```
2 < 1 || 8 < 9
```

Boolean Or

boolean expressions can also be combined with OR (||)

Semantics: true if either side is true.

```
3 > 1 || 5 < 1  
true
```

```
2 < 1 || 8 < 9  
true
```

Boolean Or

boolean expressions can also be combined with OR (||)

Semantics: true if either side is true.

```
3 > 1 || 5 < 1  
true
```

```
2 < 1 || 8 < 9  
true
```

```
2 < 1 || 9 < 8
```


Boolean Or

boolean expressions can also be combined with OR (||)

Semantics: true if either side is true.

```
3 > 1 || 5 < 1  
true
```

```
2 < 1 || 8 < 9  
true
```

```
2 < 1 || 9 < 8  
false
```

Example:
Or .java

Boolean Not

Can negate a boolean expression with not (!).

Semantics: `!true == false` and `!false == true`.

Boolean Not

Can negate a boolean expression with not (!).

Semantics: `!true == false` and `!false == true`.

`!(1 < 2)`

Boolean Not

Can negate a boolean expression with not (!).

Semantics: `!true == false` and `!false == true`.

```
!(1 < 2)
false
```

Boolean Not

Can negate a boolean expression with not (!).

Semantics: `!true == false` and `!false == true`.

`!(1 < 2)`
`false`

`!(1 > 7)`

Boolean Not

Can negate a boolean expression with not (!).

Semantics: `!true == false` and `!false == true`.

`!(1 < 2)`
`false`

`!(1 > 7)`
`true`

Boolean Not

Can negate a boolean expression with not (!).

Semantics: `!true == false` and `!false == true`.

```
!(1 < 2)  
false
```

```
!(1 > 7)  
true
```

```
!(1 < 2 && 1 > 3)
```


Boolean Not

Can negate a boolean expression with not (!).

Semantics: `!true == false` and `!false == true`.

```
!(1 < 2)
false
```

```
!(1 > 7)
true
```

```
!(1 < 2 && 1 > 3)
true
```

Example:

Not .java

Putting it Together:

`ComplexConditional.java`

Testing with Boolean Operations

Uses of `&&` and `||` usually mean
more tests are appropriate

Testing with Boolean Operations

Uses of `&&` and `||` usually mean more tests are appropriate

```
if (x == 1 || x == 5) {  
    return 7;  
} else if (x > 7 && x <= 20) {  
    return 8;  
} else {  
    return 55;  
}
```

Testing with Boolean Operations

Uses of `&&` and `||` usually mean more tests are appropriate

```
    Test: x = 1
if (x == 1 || x == 5) {
    return 7;
} else if (x > 7 && x <= 20) {
    return 8;
} else {
    return 55;
}
```

Testing with Boolean Operations

Uses of `&&` and `||` usually mean more tests are appropriate

```
    Test: x = 1      Test: x = 5  
if (x == 1 || x == 5) {  
    return 7;  
} else if (x > 7 && x <= 20) {  
    return 8;  
} else {  
    return 55;  
}
```

Testing with Boolean Operations

Uses of `&&` and `||` usually mean more tests are appropriate

```
    Test: x = 1           Test: x = 5  
if (x == 1 || x == 5) {  
    return 7;  
} else if (x > 7 Test: x = 8 && x <= 20) {  
    return 8;  
} else {  
    return 55;  
}
```


Testing with Boolean Operations

Uses of `&&` and `||` usually mean more tests are appropriate

```
    Test: x = 1          Test: x = 5  
if (x == 1 || x == 5) {  
    return 7;  
} else if (x > 7 Test: x = 8 && x <= 20) {  
    return 8;  
} else {  
    return 55; Test: x = 21  
}
```

Putting it Together:

`ComplexConditionalTest.java`