

# COMP 110/L Lecture 11

Mahdi Ebrahimi

Slides adapted from Dr. 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
```

# Truth Table

Truth tables show the result of combining any two expression boolean expressions using the **AND** operator and the **OR** operator (or the **NOT** operator).  
**You should memorize/learn these values.**

condition 1 (e.g., X)	condition 2 (e.g., Y)	X AND Y ( X && Y )
false	false	false
false	true	false
true	false	false
true	true	true

**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`

# Truth Table

Truth tables show the result of combining any two expression boolean expressions using the **AND** operator and the **OR** operator (or the **NOT** operator). **You should memorize/learn these values.**

condition 1 (e.g., X)	condition 2 (e.g., Y)	X OR Y ( X    Y )
false	false	false
false	true	true
true	false	true
true	true	true

**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
```

# Truth Table

Truth tables show the result of combining any two expression boolean expressions using the **AND** operator and the **OR** operator (or the **NOT** operator).  
**You should memorize/learn these values.**

condition 1 (e.g., X)	NOT X ( !X )
false	true
true	false

**Example:**  
Not .java

# Truth Table

Truth tables show the result of combining any two expression boolean expressions using the **AND** operator and the **OR** operator (or the **NOT** operator).  
**You should memorize/learn these values.**

condition 1 (e.g., X)	condition 2 (e.g., Y)	NOT X ( !X )	X AND Y ( X && Y )	X OR Y ( X    Y )
false	false	true	false	false
false	true	true	false	true
true	false	false	false	true
true	true	false	true	true

# Putting it Together:

`ComplexConditional.java`

Operators	Description	Associativity
() .	Function Call Member Selection	Left to Right
!	Logical negation	Right to Left
* / %	Multiplication Division Modulo	Left to Right
+ -	Addition / Subtraction	Left to Right
< <= > >=	Relational Less Than / Less than Equal To Relational Greater / Greater than Equal To	Left to Right
== !=	Equality Inequality	Left to Right
&&	Logical AND	Left to Right
	Logical OR	Left to Right
=	Assignment Operators	Right to Left

# 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 && x <= 20) {  
    return 8;  
} else {  
    return 55; Test: x = 21  
}
```

**Putting it Together:**

`ComplexConditionalTest.java`