Logical Operators

Deborah A. Trytten

CS 1323 & 1324

And (&&)

- Usually same as colloquial English
  - I’m going to the store and the movies
- Show truth table for &&
- Example: Write a conditional that prints out an int if it is between 10 and 20, inclusive

Think, Pair, Share

- Write a conditional that prints out an int if it is divisible by both 6 and 8 using the && operator

iClicker Question

Given the following variables:

```
int x = 3;
int y = 7;
```

What is the truth value of the expression below?

```
(x < 5) && (y < 2)
```

a) True

b) False

Announcements and Reminders (Sept 13)

- Upcoming assignments:
  - TC 4 and TC 5 due Friday (today)

- Midterm 1 next Monday
  - Review old midterms posted on Canvas
  - Bring a pencil and eraser
  - No calculator

- Homework 2 solution posted on Canvas

Or | |

- Does not work the same as colloquial English
  - English is contextual and Java is not

- I’m going to the store or I’m going to the movies
  - Usual implication is one or the other but not both
  - This kind of or is called an exclusive or
    - Symbol is ^ in Java (rarely used in practice)

- Show truth table for | |

- Example: Print out a String if it starts with a vowel (a, e, i, o, u)

iClicker Question

Given the following variables:

```
int x = 3;
int y = 7;
```

What is the truth value of the expression below?

```
(x < 5) || (y < 2)
```

a) True

b) False

Not (!)

- Generally same as colloquial English
- Show truth table
- Example: Print out a String if it starts with a consonant
  - Vowel is a, e, i, o, u
  - Where is the not?
- Sometimes much easier to look at the opposite of what you want

iClicker Question

Given the following variables:

```
int x = 3;
int y = 7;
```

What is the truth value of the expression below?

```
(!(x < 5)) || (y < 2)
```

a) True

b) False

ShortCut

- Both && and | | are shortcut operators
  - This means they stop as soon as they can determine the result
    - Good for efficiency
    - Sometimes necessary

- && stops when…

- | | stops when…

Precedence

- Goal: minimize parentheses in common statements

- `x >= 3 && x <= 10`
  - Should && have higher or lower precedence than relational operators?

- `x == 4 || x == 5`
  - Should || have higher or lower precedence than equality operators?

- What about && and ||
  - Recall multiplicative operators have higher precedence than additive

Precedence Table

1. Parentheses
2. + (Unary) – (Unary) !
3. * / %
4. + -
5. > < <= >=
6. == !=
7. &&
8. | |
9. =

0 is false
1 is true
&& is like multiplication
| | is sort of like addition

&&, | |, ! Interact Oddly

- Example: Write a conditional that allows the user to repeat data entry until they enter an integer in the range of 20 to 30 (inclusive)

- In range: `(x >= 20) && (x <= 30)`

- Out of range?
  - Examine a number line

De Morgan's Laws

- A regular distributive law is below (algebra)

```
a*(b + c) is a*b + a*c
```

- Rules:
  - `!(P && Q) is !P || !Q`
  - `!(P || Q) is !P && !Q`

- Can think of this as some kind of weird distributive law

iClicker Question

- To find numbers that are divisible by 6, we could look for ones that are divisible by both 2 and 3:

```
number%2 == 0 && number%3 == 0
```

- To find a number that is **not** divisible by 6, which of the expressions below is correct?

a) `number%2 == 0 || number%3 == 0`

b) `number%2 != 0 && number%3 != 0`

c) `number%2 != 0 || number%3 != 0`

d) `!(number%2 == 0 || number%3 == 0)`