

---

## CSE 142

### Conditional Statements & Boolean Expressions

1/10/2003

(c) 2001-3, University of Washington

G-1

---

## Outline for Today

- Conditional statements – *if*
- Boolean expressions
  - Comparisons (<, <=, >, >=, !=, ==)
  - Boolean operators (and, or, not - &&, ||, !)
- Class invariants

1/10/2003

(c) 2001-3, University of Washington

G-2

---

### *withdraw* Method for BankAccount

```
/** Withdraw the requested amount from this BankAccount */
public void withdraw(double amount) {
    balance = balance - amount;
}
```

- Critique: is this good/bad/incomplete?

1/10/2003

(c) 2001-3, University of Washington

G-3

---

### Class Invariants

- In many cases, the state of an object must obey some rules to have a sensible value
  - For a BankAccount, some rules might be:
    - that balance >= 0.0 always
    - that the account must have a non-empty name
- These rules are examples of invariants
- Things that must always be true, if the program is operating correctly
- Invariants concerning the state of objects are called class invariants

1/10/2003

(c) 2001-3, University of Washington

G-4

## More About Invariants

- Invariants are *not* syntax rules of Java
- Advice:
  - Write down invariants as comments
  - When you implement methods, double check that you never violate the invariants
- Very powerful bug prevention technique
- Java 1.4 has a special statement to help check invariants
  - *assert* (to be discussed at another time)

1/10/2003

(c) 2001-3, University of Washington

G-5

## A Better *withdraw* Method

- Specification

```
/** Withdraw requested amount from this BankAccount provided that the
 * balance is at least as large as the amount requested. Otherwise do nothing */
public void withdraw(double amount) {
}
```
- Comment in the spec. changes, but not the Java (yet)
- We want to say (in Java) something like

```
"if the amount is less than or equal to the balance, withdraw the amount"
```
- Java solution: *if* statement
  - Or "conditional" statement

1/10/2003

(c) 2001-3, University of Washington

G-6

## *withdraw* Method Implementation

```
/** Withdraw requested amount from this BankAccount provided that the
 * balance is at least as large as the amount requested. Otherwise do nothing */
public void withdraw(double amount) {
    if (amount <= balance) {
        balance = balance - amount;
    }
}
```

1/10/2003

(c) 2001-3, University of Washington

G-7

## *If* Statement Syntax

- Syntax

```
if ( condition ) {
    list of statements
}
or
if ( condition ) {
    list1 of statements
} else {
    list2 of statements
}
```
- *condition* must be a Boolean expression – one that is either true or false
- *list of statements* may contain any Java statements, including *if*(!)

1/10/2003

(c) 2001-3, University of Washington

G-8

## If Statement: Meaning of Each Form

```
if ( condition ) {  
    list of statements  
}  
or  
if ( condition ) {  
    list1 of statements  
} else {  
    list2 of statements  
}
```

- Meaning of first form
  - Evaluate condition
  - If the condition is true, execute the list of statements
  - If it is false, do nothing (skip statements)
- Meaning of second form
  - Evaluate condition
  - If the condition is true, execute the first list of statements and skip the second one
  - If the condition is false, skip the first list of statements and execute the second one

1/10/2003

(c) 2001-3, University of Washington

G-9

## Better *withdraw* Method

- Instead of silently doing nothing if amount is too large, return a Boolean result to indicate if the withdraw succeeded.
- Note that this is a change in the specification!

```
/* Withdraw requested amount from this BankAccount and return true, provided  
 * that the balance is at least as large as the amount requested. Otherwise  
 * return false */  
public boolean withdraw(double amount) {  
    ...  
}
```

1/10/2003

(c) 2001-3, University of Washington

G-10

## Better *withdraw* Method: Implementation

- Instead of silently doing nothing if amount is too large, return a Boolean result to indicate if the withdraw succeeded.
- Note that this is a change in the specification!

```
/* Withdraw requested amount from this BankAccount and return true, provided  
 * that the balance is at least as large as the amount requested. Otherwise  
 * return false */  
public boolean withdraw(double amount) {  
    if (amount <= balance) {  
        balance = balance - amount;  
        return true;  
    } else {  
        return false;  
    }  
}
```

1/10/2003

(c) 2001-3, University of Washington

G-11

## Boolean Expressions

- Boolean constants
  - true
  - false
- Simple relations on numbers also give boolean values
  - > >= < <= != ==
  - All are binary operators
  - Note use of == for equality comparison (not!!! single =)
  - Examples
    - x > y
    - x\*2.5 - 17.0 <= 0.0
    - balance >= amount

1/10/2003

(c) 2001-3, University of Washington

G-12

## Boolean Operators

- Make complex boolean expressions from simpler boolean expressions
- `&&` means "and"
  - true if *both* expressions are true, false otherwise  
`x > 10 && x <= 100`
    - Can only compare two things at a time; can't do `10 < x <= 100`
- `||` means "or"
  - true if *either* expression is true, false only if both are false  
`x > y || x <= 0`
- `!` means "not"
  - true if expression is false  
`!(x < y)` // means same thing as `x >= y`

1/10/2003

(c) 2001-3, University of Washington

G-13

## Practice With Boolean Expressions

- Suppose `x` is 10 and `y` is two. What is the value of each expression?
  - `x < 9`
  - `x == y - 8`
  - `x >= 0`
  - `y == 0 || x != 3`
  - `y != x && x > y`
  - `!(x < y)`

1/10/2003

(c) 2001-3, University of Washington

G-14

## Exercise

- Recall that the statement  
`System.out.println("Hi there!");`  
will write a message (in this case, "Hi there!")
- Exercise 1: assume that we have a double variable called `temperature` holding the outside temperature. Write the message "Too Hot!" if the temperature is above 80.
- Exercise 2: use the variable `temperature` as above, but this time write "Too Hot!" if the temperature is above 80, "Too Cold!" if it is below 60, and "Just Right" if it is in between.

1/10/2003

(c) 2001-3, University of Washington

G-15

## Solution to Exercise 1

1/10/2003

(c) 2001-3, University of Washington

G-16

## Solution to Exercise 2

---

1/10/2003

(c) 2001-3, University of Washington

G-17

## Summary

---

- Invariants
- Conditional execution – *if* statement
- Boolean expressions
  - Comparisons
  - Operators – and, or, not

1/10/2003

(c) 2001-3, University of Washington

G-18