CMSC 131
Object-Oriented Programming I

Loops (while, do while)

Dept of Computer Science
University of Maryland College Park

This material is based on material provided by Ben Bederson, Bonnie Dorr, Fawzi Emad, David Mount, Jan Plane

Overview

- while loops
- do while
- scope

Loops in Java

- So far our programs execute every program statement at most once
- Often, we want to perform operations more than once
  - "Sum all numbers from 1 to 10"
  - "Repeatedly prompt user for input"
- Loops allow statements to be executed multiple times
  Loop types in Java:
  - while
  - do-while
  - for
- Called "iteration"

while and do-while Loops

- **while** and **do-while** loops contain
  - A statement, called the body
  - A boolean condition
  - Idea → the body is executed one more time as long as the condition is true

- **while-loop** → The condition is tested before each body execution

```
while ( condition ) {
    body
}
```

- **do-while-loop** → The condition is tested after each body execution (notice the semicolon after the right parentheses)

```
do {
    body
} while ( condition );
```

while and do-while Loops

- **Main difference:** do-while loop bodies always executed at least once because it is “bottom tested” rather than “top tested”
- No need for { } if only one statement is executed (you should always use them regardless the number of statements ☺)
- What is the iteration variable?
- **Examples:** SimpleWhile.java, SimpleDoWhile.java, AskAge.java

Types of loops

- Indefinite iteration (infinite loop)
  - Usually tests something that is coming from outside the loop structure (e.g. input)
  - Needs to eventually change from **true** to **false**
- Counted iteration
  - Something that is controlled inside the loop
  - To start at some value and count up or down until some set ending point

Infinite Loops

- Loops can run forever if condition never becomes false
- Be careful when programming loops!
  - Add statements for termination into loop body first
  - Often these statements are at end of the body
  - How to stop a loop in Eclipse?
    - Select the red square button near the console tab
- **Example:** FastArithmetic.java
- Another source of infinite loop
  - Semicolon after loop expression in while loop

Infinite Loops

- Loops can run forever if condition never becomes false
- Be careful when programming loops!
  - Add statements for termination into loop body first
  - Often these statements are at end of body
  - e.g.

```
while(i <= 10) {
    System.out.println(i);

    i = i + 1;
}
```

Variables, Blocks and Scoping

- Variables can be declared anywhere in a Java program
- When are the declarations active?
  - After they are executed
  - *Only inside the block in which they are declared*
- Scope rules formalize which variable declaration are active when
  - Global variables → scope is entire program
  - Local variables → scope is a block

Trace Tables

- Mechanism to keep track of values in a program
- Allows you to understand the program behavior
- You create a column for each variable in your program
- You update values as you execute the program
- Let's create a trace table for one of our examples

Combination of Statements

- You can have any combination of conditional and iteration statements
  - Conditionals inside of loops
  - Conditionals inside conditionals
  - Loops inside conditionals
  - Loops inside loops

JOptionPane

- JOptionPane
  - Provides dialog boxes
  - We need “import javax.swing.*”;
- showInputDialog
  - For input
  - Returns a string
  - For numerical operations we need
    - Integer.parseInt → Conversion to int
    - Double.parseDouble → Conversion to double
- showMessageDialog
  - For output
- **Example:** DaysCalculator.java