
CSE 142
Introduction to Iteration

2/10/2002 (c) University of Washington CSE 2001-2 M-1

Introduction

- **Topics**
 - Doing some action repeatedly
 - Loop statements – while
- **Reading**
 - Dugan notes: ch. 15
 - Niño & Hosch : first part of ch. 12

2/10/2002 (c) University of Washington CSE 2001-2 M-2

Repeating Actions

- Think of the following algorithms:
 - Drive until you see a fork in the road.
 - Give a cookie to each of your instructors.
 - Bake the roast until it has an internal temperature of 220 degrees.
 - While there are still donuts in the box, eat one.
 - Lather, rinse, repeat.
- They all call for us to repeat some action for a period of time. Repeating is also called *iterating*.
- In Java, *loop statements* provide us with a way to execute a block of statements repeatedly.

2/10/2002 (c) University of Washington CSE 2001-2 M-3

Computing an Average (1)

- Suppose we want to calculate the average of a set of numbers
- We need to know:
 - Total of all the numbers
 - How many numbers there were
- But when we write our program we won't know how much data there is!
- Solution: keep asking for new numbers...

2/10/2002 (c) University of Washington CSE 2001-2 M-4

Computing an Average (2)

- An algorithm:
 1. Ask the user if they have more data to enter.
 2. If they have more, then ask for the amount, add it to the running total, increase the count of how much data we've read, and go back to step 1.
 3. Otherwise, we're done. Compute the average.

2/10/2002 (c) University of Washington CSE 2001-2 M-5

A Flow Chart

- We can draw a picture of the steps in this algorithm:

```
graph TD; A{1. Do you have some data?} -- yes --> B[2. What is the data?]; B --> C[Add data to total & increase count]; C --> A; A -- no --> D[3. Compute the average (total/count)];
```

- This kind of picture is called a *flow chart*.

2/10/2002 (c) University of Washington CSE 2001-2 M-6

A Rephrasing

- Another way to say this algorithm:

While the user has data to enter,
Ask the user for the amount.
Add it to the rain gauge running total.
Add 1 to the count of data read
Then go back to the top.
Otherwise, we're done. Compute the average.

- Detail: need to set the running total and count to 0 before we start

2/10/2002 (c) University of Washington CSE 2001-2 M-7

The Java Version

- We can translate this directly to Java code:

```
Input input = new Input();
double total = 0.0;           // sum of numbers read so far
int inputCount = 0;          // count of numbers read so far

while ( input.readBoolean("Do you have data to enter?") == true ) {
    double amount = input.readDouble("What is the amount?");
    total = total + amount;
    inputCount = inputCount + 1;
}

double average = total/inputCount;
```

- A while statement in Java is a way to express a loop

2/10/2002 (c) University of Washington CSE 2001-2 M-8

The While Loop Pattern

- New Java statement to repeatedly execute other statements:

```
while (<test-expression> ) {
    <body-statements>
}
```

- Meaning: repeatedly do the following
 - Evaluate <test-expression> (a boolean expression)
 - If <test-expression> is true, execute <body-statements>, and then restart from the top, reevaluating <test-expression> etc.
 - If <test-expression> is false, then quit without executing body

2/10/2002 (c) University of Washington CSE 2001-2 M-9

A Problem: Computing Interest

- Problem: how many years will it take, at 10% interest per year, for your money to double?

- What's the algorithm, in English?
- What's the flow chart?
- What's the Java code?

2/10/2002 (c) University of Washington CSE 2001-2 M-10

Solution

2/10/2002 (c) University of Washington CSE 2001-2 M-11

Solution

2/10/2002 (c) University of Washington CSE 2001-2 M-12