Common Pitfalls and Errors when Using Loops

Hui Chen

Department of Computer & Information Science

Brooklyn College

Objectives

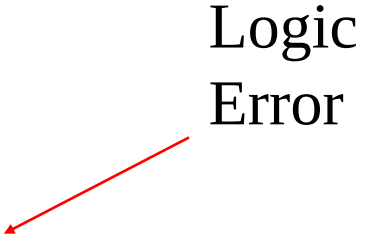
- To learn the techniques for minimizing numerical errors (§5.10).
- To discover the similarities and differences of three types loop statements (§5.8)

Outline

- Discussed
  - Concept of loops
  - While loops vs. do-while loops vs. for loops
  - Nested while loops
- Pitfalls and Errors
- Which loop statement to use?

Pitfalls and Errors. for Loop and ";"

- Adding a semicolon at the end of the for clause before the loop body is a common mistake, as shown below




```
for (int i=0; i<10; i++);
{
  System.out.println("i is " + i);
}
```

Pitfalls and Errors. while Loop and ";"

- Similarly, the following loop is also wrong (unless it is intended).

```
int i=0;
while (i < 10);
{
  System.out.println("i is " + i);
  i++;
}
```

Pitfalls and Errors. How about do-while Loop?

- In the case of the do loop, the following semicolon is needed to end the loop.

```
int i=0;
do {
  System.out.println("i is " + i);
  i++;
} while (i<10);
```

Pitfalls and Errors. Numeric Errors

- We often use loops to process floating point values.
  - Using floating point numbers in the loop continuation condition may cause numeric errors
  - Adding floating point numbers from biggest to smallest is less accurate than adding from smallest to biggest

Computing the Sum of Floats

- Observe

```
float sum = 0.;
for (float i=0.01f; i <= 1.0f; i=i+0.01f) {
  sum += i;
}
System.out.println(“The sum is “ + sum);
```

Computing the Sum of Doubles

- Observe

```
double sum = 0.;
for (double i=0.01; i <= 1.0; i=i+0.01) {
  sum += i;
}
System.out.println("The sum is " + sum);
```

Using Integer Loop Variable

- To compute the sum of float point numbers (float or double), we use an integer loop variable

But, Compare the following two solutions

```
double currentValue = 0.01;
for (int count = 0; count < 100; count ++) {
  sum += currentValue;
  currentValue += 0.01;
}
```

```
double currentValue = 1.0;
for (int count = 0; count < 100; count ++) {
  sum += currentValue;
  currentValue -= 0.01;
}
```

Which one gives us answer with smaller error?

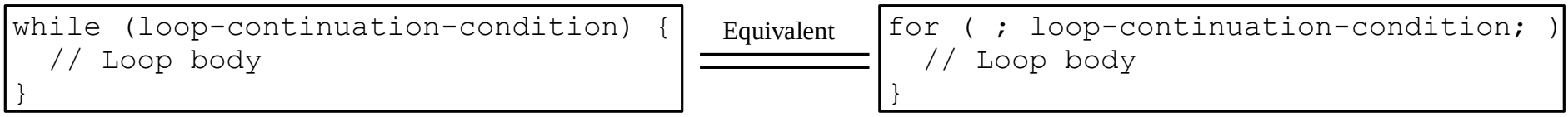
Questions?

Which Loop to Use?

- The three forms of loop statements, while, do-while, and for, are expressively equivalent
- You can write a loop in any of these three forms.

Converting while Loop to for Loop

- For example, a while loop in (a) in the following figure can always be converted into the following for loop in (b)



(a) (b)

Converting for Loop to while Loop

- A for loop in (a) in the following figure can generally be converted into the following while loop in (b) except in certain special cases (see Review Questions of the chapter)

```
for (initial-action;
     loop-continuation-condition;
     action-after-each-iteration) {
  // Loop body;
}
```

(a)

Equivalent

```
initial-action;
while (loop-continuation-condition) {
  // Loop body;
  action-after-each-iteration;
}
```

(b)

Question. Which one to use?

Recommendations

- Use the one that is most intuitive and comfortable for you
- In general, a for loop may be used if the number of repetitions is known, as, for example, when you need to print a message 100 times.
- A while loop may be used if the number of repetitions is not known, as in the case of reading the numbers until the input is 0.
- A do-while loop can be used to replace a while loop if the loop body has to be executed before testing the continuation condition

Questions?