CS251L

REVIEW

2010.8.27 | Derek Trumbo | UNM

Announcements

- Temporary class web site at:
  - http://www.cs.unm.edu/~dtrumbo/cs251
  - Contains just materials for download
  - Might begin to post announcements as well
- Temporary mailing list set up at:
  - http://mail.cs.unm.edu/cgi-bin/mailman/listinfo
  - Click on "Cs251" and fill out form to subscribe

    (e-mail is sent that you then confirm)

Announcements

- Syllabus Posted
- HW 1 Posted
  - This article will help with the assignment: http://en.wikipedia.org/wiki/Epicycloid
  - More information will be posted as questions arise over the next week
  - Due Friday, Sept 3

Announcements

- Office Hours
  - Room FEC 321 (CS Department)
  - WF 12-1p (or by appointment)
- TA's:
  - Joe Collard (Tuesday 11a & 3p)
    - cs.unm.jcollard@gmail.com
    - http://cs.unm.edu/~jcollard/fall.2010/cs251/
    - OH: Tuesday 8 - 9a, 4 - 5p
  - Jesse Lockwood (Tuesday 12p)
    - skynet641@aol.com
    - OH: Tuesday 10:45-11:45a, Thursday 12-1p

Announcements

- Will occ*asionally* speak about the awesome developments taking place in software today
- Let me know if you're not understanding something – I want to know!!

Operator Extras

- Prefix increment/decrement operators vs. posfix increment/decrement operators

```
int a = 7;
int b = 7;

int x = ++a;  // Add one to var, return new val
int y = b++;  // Add one to var, return prev val
```

Variables a & b are *both* **8** at this point.

However, x is **8** and y is **7**. Value returned from postfix operators is the original value of variable.

Operator Extras

- Ternary operator – allows quick decision to be made with 3 operands (only operator with 3 operands)

```
condition ? trueval : falseval

String msg = enrolled ? "Yes" : "No";
int theCode = y > 10 ? highCode() : lowCode();
double price =
    (onSale && day != SUN) ? 9.99 : 14.99;
```

Operator Extras

- Ternary operator just a convenience – can always use an if statement to replace.

```
String msg = enrolled ? "Yes" : "No";

String msg;
if(enrolled) {
   msg = "Yes";
} else {
   msg = "No";
}
```

Decision Statements

- if-else statements

```
if(x > 10) {
    oneStatement();          // Unnecessary braces
}
```

vs.

```
if(x > 10)
    doSomething();
    doSomethingElse();       // Ugly bug
```

Iteration

- while loops
  - When you want some code to execute while some condition is true.
  - The loop body may never execute since loop is top-tested

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

Iteration

- Notice that for loops essentially are just while loops in disguise

```
int a = 0;
while(a < 100) {
    doSomething(a);
    a++;
}
```

Iteration

- Consider this loop used in a traffic simulation. Simulated driver doesn't stop for yellow lights, but regardless of the light color, will never enter an intersection if there's a car in it to avoid an accident

```
while(light == GR || light == YL && noCarsAhead){
    enterIntersection();
    exitIntersection();
    travelNextRoadSegment();
}
```

Iteration

- Wrong – remember operator precedence

```
while((light == GR || light == YL) &&
                                    noCarsAhead) {
    enterIntersection();
    exitIntersection();
    travelNextRoadSegment();
}
```

Iteration

- do-while loops
  - When you want some code to execute while some condition is true.
  - The loop body is guaranteed to execute at least once as this is a bottom-tested loop.

```
do {
    // loop body
} while(condition);  // Notice required ;
```

Arrays

- In programming, a named variable can only be referenced by its one single name, and you must type that name exactly to reference the value inside

```
String studentName = "Leeroy";
System.out.println("Student: " + studentName);
```

Arrays

- So if you wanted to have a variable that stores *extremely related* but unique information, you'd have to create a differently-named variable:

```
String studentName1 = "Leeroy";
String studentName2 = "Jenkins";
System.out.println("Student: " + studentName1);
System.out.println("Student: " + studentName2);
```

Arrays

- When storing lists of related data, creating uniquely named variables is not feasible
- Every programming language supports arrays in some fashion
- An array is a way to use the same variable name to reference an entire list of some data type, and also an integer value to specify an exact element in that array

Arrays

- The ability to use an integer to select the exact element is an extremely powerful construct
- Arrays and for loops are extremely symbiotic since for loops have that loop counter variable!
- An array of size 5 has valid elements

  array[0], array[1], array[2], array[3], array[4]
- Array indexing is zero-based!
- Attempting to access array[5] will raise an error!

Arrays

- Example using an array:

```
String[] studentNames = new String[2];
studentNames[0] = "Leeroy";
studentNames[1] = "Jenkins";
for(int s = 0; s < studentNames.length; s++) {
   System.out.println("SNm: " + studentNames[s]);
}
```

Arrays

- Example using an array:

```
String[] studentNames = new String[2];
studentNames[0] = "Leeroy";
studentNames[1] = "Jenkins";
for(int s = 0; s < studentNames.length; s++) {
   System.out.println("SNm: " + studentNames[s]);
}
```