COE318 Lecture Notes Week 1 (Week of Sept 1, 2014)

Topics

- Course management
- Programming Languages
- Java usage
- Java and C
- The boolean data type in Java
- The overloaded + operator
- First glimpse at the flavour of object-oriented programming
- Hello World!

Programming language types

The popularity of various programming languages as of August 2012 is shown below (taken from http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html)

Aug 2014	Aug 2013	Change	Programming Language	Ratings	Change
1	2	▲	C	16.401%	+0.43%
2	1	▼	Java	14.984%	-0.99%
3	4	▲	Objective-C	9.552%	+1.47%
4	3	▼	C++	4.695%	-4.68%
5	7	▲	Basic	3.635%	-0.24%
6	6		C#	3.409%	-2.71%
7	8	▲	Python	3.121%	-0.48%
8	5	▼	PHP	2.864%	-3.83%
9	11	▲	Perl	2.218%	+0.18%
10	9	▼	JavaScript	2.172%	+0.08%

Notes:

- 4 of the 5 most used languages are *Object-oriented*. (Only C is not; it is a *procedural* language.)
- The rapid rise in recent years of Objective-C is probably due to its being the language used to develop iPad or iPhone apps.
- Knowing any one of the 4 object-oriented languages makes learning any of the others much easier.
- C# is the most like Java, then Objective-C while C++ is the most different.

Java Uses

Java is used:

- As a general-purpose programming language. (This is how we use Java in this course.)
- The programming language for devices such as cell phones (Android, Blackberry and Nokia), Android tablets, Blu-ray players and some TVs.
- Widely used at the server end of Internet applications.
- As the primary programming language in well-known projects such as:
  - The Mars rover program at JPL.
  - The Large Hadron Collider at CERN.
  - *Watson* – the IBM computer that defeated the best human players on the TV show *Jeopardy!*.

A First look at Java syntax

- Consider the following code fragment:

```
int c = 10;
int sum = 0;
while (c != 0) {
 s += c;
 c--;
}
```

- It looks like C. But it is also valid Java (or Objective-C, C# or C++)!
- Most of the syntax you learned in C is the same in Java (or any of the other C-like object-oriented languages.)
- All of these languages are *strongly typed* --- you have to declare the type of a variable before using it.
- Control structures like *if*, *while*, and *for* work the same way.
- And although Java does not have *functions*, it does have *methods* which look syntactically just like functions.
- The **big** difference between C and Java is not syntax and a few new keywords; rather, it is how to exploit the object-oriented paradigm. And that is what this course is about!
- Before looking at this, however, let's examine two differences between C and Java.

The `boolean` data type in Java

- There is no boolean data type in C; but there is one in Java.
- In C, the `int` data type is used to represent *true* and *false*. The value 0 (zero) represents *false*;

any other value (123, -6578, 1, ...) represents *true.*

- When the conditional in an *if* or *while* statement is something that is true or false (such as comparing with ==, !=, >=, etc.) there is no difference between the Java and C code.
- For example, the following function (C) or method (Java) are valid in both C and Java:

```
int biggest(int x, int y) {
 if(x > y) {
      return x;
 }
 return y;
}
```

- In Java, there is a `boolean` data type with only two possible values: `true` and `false` (which are keywords in Java). The differences between Java and C are apparent with the standard way to write an infinite loop. The Java version works only in Java and the C version only in C.
- The C version of an infinite loop:

```
while(1) {
 //do something forever
}
```

- The Java version of an infinite loop:

```
while(true) {
 //do something forever
}
```

The overloaded + operator in Java

- The expression `a + b` where both a and b are numbers (ints, shorts, floats, doubles, etc.) evaluates the sum (addition) of the two numbers. In this case, the + operator is the same in both Java and C.
- However, in Java you can also use the + between two Strings and the resulting evaluation is a new String that is the concatenation (joining) of the two original Strings (which, however, remain unchanged.)
- Thus in Java, `"ab" + "cde"` is evaluated as the String `"abcde"`.
- Similarly, `"2" + "3"` is evaluated as the String `"23"`.
- But what does + mean when when placed between a number and a String?
- In this case, the number is converted into a String and the two Strings are concatenated.
- Thus `2 + "4"` evaluates to the String `"24"`.
- What about `2 + 3 + "4"`?
- The expression is evaluated from left to right; when `2 + 3` is encountered, it evaluates to the number 5. The expression is now reduced to 5 `+ "4"` which evaluates to `"54"`.

- Similarly, `"2" + 3 + 4` evaluates as `"234"`.
- When expressions are enclosed in parenthesis, they are evaluataed first. Thus `"2" + (3 + 4)` evaluates as `"27"`.
- A very common idiom is converting anything into a String by concatenating it with an empty String. Thus `"" + 2` yields the String `"2"`.

A First Glimpse at Object-oriented Programming

- Suppose we wish to model various animals such as dogs, cats, tigers, etc.
- In Java we can write something like:

```
//Declare variables felix and rover as type Animal
Animal felix, rover;
//Instantiate felix as a Cat object
felix = new Cat();
//Instantiate rover as a Dog object
rover = new Dog();
//Ask rover to speak
rover.speak();//He will say "Woof! Woof!"
//Ask felix to speak
felix.speak(); //He will say "Meow meow..."
```

- To do the same thing in a procedural language like C might require if-statements.
- The pseudo-code might look something like:

```
if it's a dog then
    say Woof! Woof!
else if it's a cat then
    say "Meow meow..."
```

- A real cat, of course, does not say to itself "Humm, I'm not a dog so I won't say woof but I am a cat so I'll say meow".
- Similarly, a Cat object simply "knows" that it's a cat and says moew.
- The formal aspects of object-oriented programming used here are called *inheritance* and *polymorphism*. These concepts will be explained and explored as the course progresses.
- One hallmark of object-oriented languages is that there tend to be fewer if-statements than there are in procedural ones. And getting if-statements wrong is a common cause of programming bugs.

Hello Word in Java

- We are now ready for our first program: Hello World.
- This will also introduce you to the Netbeans IDE (Integrated Development Environment) used in this course.
- It is **highly recommended** that you install Java and Netbeans on your own computer and do the tutorial for the HelloWorld program found at http://netbeans.org/kb/docs/java/quickstart.html.
- The program is shown below:

```
package coe318.week1;

/**
 *
 * @author kclowes
 */
public class Hello {

    /**
     * Prints "Hello world".
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        System.out.println("Hello world");
    }
}
```

- This looks a lot worse than the equivalent program in C!
- And, alas, a full explanation of each of the elements in the program will have to wait a few weeks!
- But don't despair. It will all make sense in the end.
- The essential features can be briefly described as follows:
    - The source code describes a *class* called `Hello`.
    - By convention, class names are capitalized and should be given some sort of name that describes it.
    - The keyword *public* makes the class Hello and the *static method* `main` available to anyone.
    - Comments can be delimited by `/* comment */` as in C or with `//` which makes the rest of the line a comment.
    - The keyword *static* indicates that what is being declared belongs to the Hello class and exists even if no objects of type Hello exist. (You'll have to wait a bit for that to be

understandable; for the moment just accept that the main method must be declared as static.)

- The keyword *void* indicates that the method `main` does not return anything. (Yes, this is another minor difference between C and Java. In C you may recall that the main function returns an integer.)
- The argument to the main method is an array of Strings. Arrays in Java are similar to arrays in C but there are significant differences.
- The line `System.out.println("Hello world");` produces the actual output.
- Here `System` is a class name, `out` is static PrintStream object belonging to the System class and `println` is a method that can be applied to any PrintStream object.

Classes and Objects

- A *class* defines the general properties, characteristics and behavior of an *object* of that type.
- For example, a the class (or category) "car" defines something with 4 wheels, a body, brakes, steering wheel, etc. An *object* of class "car" could be a blue Prius belonging to Alice Smith.

```
package coe518.week1;

public class Person {
    private String name;
    private boolean isMale;
    private int birthYear;

    public Person(String n, boolean male, int by) {
        name = n;
        isMale = male;
        birthYear = by;
    }



    public int getAge(int year) {
        return year - birthYear;
    }

    public String toString() {
        return name + "(" + (isMale ? "M" : "F")
                + ")" + " Born: " + birthYear;
    }
}
```

```
package coe518.week1;

public class Main {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        Person alice = new Person("Alice", false, 1994);
        Person bob = new Person("Robert", true, 1995);
        System.out.println("" + alice);
        System.out.println(bob);
    }

}
```

Questions

The answers are given in the next section. It is ***strongly advised*** *that you attempt the questions before peeking at the answers!*

1. Write an infinite while loop that is valid in both C and Java.
2. What value is evaluated for each of the following expressions? If the expression is not valid Java, indicate *syntax error.*
   a) `1 + 2 + "x"`
   b) `(1 + 2) + "x"`
   c) `(2 > 1 ? "a" : "b") + 3`
   d) `(2 > 1) + "5"`
   e) `"" + (2 < 1) + 23`
   f) `"" + 3 + 45 + (1 + 2) + 5`
3. The following is valid code for a Java method or a C function:

```
int foo(int n) {
 int i, s = 0;
 for(i = 1; i <= n; i++) {
      s = s + i*i;
 }
```

```
  return s;
 }
```

a) What value is returned when `foo(2)` is invoked?

b) What value is returned when `foo(3)` is invoked?

c) Give a better name than foo to the function.

4. Write a Java method (same as a C function) `int f(int n, int i)` that returns the sum of *n* integers each raised to the *i*th power. For example, `f(4, 3)` would return $1^3+2^3+3^3+4^3=1+8+27+64=100$

Answers

1. Just make the conditional something that is always true. For example:

```
while(2 == 2) {
 //do something forever
 }
```

2.

```
a)     1 + 2 + "x" --> "3x"
b)     (1 + 2) + "x"  --> "3x"
c)     (2 > 1 ? "a" : "b") + 3 --> "a3"
d)     (2 > 1) + "5"  --> "true5"
e)     "" + (2 < 1) + 23 --> "false23"
f)     "" + 3 + 45 + (1 + 2) + 5 --> "34535"
```

3.

a) 5

b) 14

c) `sumOfSquares(int n)`

4.

```
int f(int n, int i) {
  int s = 0;
     for(int j = 1; j <= n ; j++) {
         int p = 1;
         for(int k = 0; k < i; k++) {
             p *= j;
         }
         s += p;
     }
  return s;
 }
```

Online resources

The following resources are available at no charge on the Internet. They all work under Windows, MacOS and Linux.

- You can download the Java software development kit (SDK) for Windows, MacOS or Linux from http://www.oracle.com/technetwork/java/javase/downloads/index.html
- An excellent tutorial is available for on-line viewing or download at http://docs.oracle.com/javase/tutorial/index.html
- The Netbeans IDE used in this course can be downloaded from http://netbeans.org/downloads/index.html
- Android users (cell phone or tablet) can program apps in Java directly on their device with the free apps AIDE or the Spartacus Rex Terminal IDE that you can get for free at the Play Store. (You can also develop apps more conveniently on your computer (any OS) and download the app to the device later.)