ISE 582: Information Technology for Industrial Engineers
University of Southern California
Department of Industrial and Systems Engineering

Lecture 5
JAVA Cup 4: Nuts and Bolts

Handouts
• Lecture 5 slides
• Assignment 4
• READ Winston & Narasimhan:
  – Chapters 21-26 (pp 117 - 147)

Announcements
• HW2 is due today
• Pick up HW1 at ISE office on Monday
• HW1: some code not working because of file permissions
• Homework presentation
  – Use folders
  – Don’t be afraid of white space
• Lab availability sufficient?
• Assignment4: conversion Celcius / Farenheit

http://www-classes.usc.edu/engr/ise/582
JAVA Cup 4

- Boolean operators and methods
- Conditional statements
- Combining Boolean expressions
- Iteration Statements
- Recursion
- Multi-way Conditions

Booleans

- ==
- !=
- >
- <
- >=
- <=
- !
- instanceof
- equals

EXAMPLE:

```java
public class Demonstrate {
    public static void main (String argv[]) {
        System.out.println(28==2);
        Gringotts js = new Gringotts();
        MuggleBank js2 = new MuggleBank();
        System.out.println(js instanceof Gringotts);
        System.out.println(js instanceof Account);
        System.out.println(js2.equals(js));
    }
}
```

Binary Predicates

Predicates Example 2

```java
public class Demonstrate1 {
    public static void main (String argv[]) {
        JamesBondMovie jm = new JamesBondMovie();
        Movie m = new Movie(1,1,1);
        System.out.println(jm instanceof Attraction);
        System.out.println(jm instanceof Movie);
        System.out.println(jm instanceof JamesBondMovie);
        System.out.println(m instanceof Attraction);
        System.out.println(m instanceof Movie);
        System.out.println(m instanceof JamesBondMovie);
    }
}
```

NEW
Predicates Example 3

```java
public class Demonstrate {
    public static void main (String argv[]) {
        Movie m1 = new Movie(3,4,5);
        Movie m2 = new Movie(3,4,5);
        Movie m3 = new Movie(4,5,6);
        System.out.println(m1.equals(m2));
        System.out.println(m2.equals(m3));
        m3=m2=m1;
        System.out.println(m1.equals(m2));
        System.out.println(m2.equals(m3));
    }
}
```

Conditional Statements

- The usual if-else statements
- Each “else” belongs to nearest unmatched “if”
- Embedding allowed

```
if (Boolean expression)
    {   …   ; }
else
    {   …   ; }
```

Conditional Statement Example 1

```java
public class Demonstrate {
    public static void main (String argv[]) {
        int s = 5;
        if (s == 1) {
            System.out.println("HP has " + s + " sickle");
        } else {
            System.out.println("HP has " + s + " sickles");
        }
    }
}
```
**Conditional Statement Exercise**

*Quidditch* is the premier sport of the wizarding world. It is played on broomsticks. Everyone follows Quidditch. The World Cup matches attract hundreds of thousands of fans from all over the world.

Design a class called *Quidditch*, where each instance contains information about the competing teams and their scores. Write a method to display the state of the game (i.e. who is winning).
The Conditional Operator

- Boolean expression ? if-true expression : if-false expression

```java
public class Demonstrate {
    public static void main (String argv[]) {
        int s = 5;
        System.out.print( "HP has " + s);
        System.out.println( s == 1 ? " sickle." : " sickles.");
    }
}
```

Combining Boolean Expressions

- && = the AND operator
- || = the OR operator
- Evaluations from Left to Right

EXAMPLES

Iteration Statements

- while ( Boolean expression ) { … ; }
- for ( entry expression; Boolean expression; continuation expression ) { … ; }
- for ( counter initialization expression; counter testing expression; counter reassignment expression ) { … ; }
Examples

```java
public static int powerOf2 (int n) {
    int result = 1;
    while (n != 0) { result = 2 * result; n = n - 1; }
    return result;
}
```

```java
public static int powerOf2 (int n) {
    int result = 1;
    for (int i = n; i != 0; i = i-1) { result = 2 * result; }
    return result;
}
```

Augmented Assignment Operator

• Short-cut
  
  ```java
  <variable> = <variable> <operator> <expr>
  ```

  ```java
  <variable> <operator><expr> e.g. x*=2
  ```

• Increment / Decrement short-cuts
  
  – Prefix (->x, ++x ): hands over new value
  – Suffix (x-, x++ ): hands over original value

QUIZ: what is the difference between
(++)x + x and (x++) + x?

More For-Loop Examples

```java
public static int powerOf2 (int n) {
    int result = 1;
    for ( int i = n; i-- != 0; ) { result = 2 * result; }
    return result;
}
```

```java
public static int powerOf2 (int n) {
    int result = 1;
    for ( int i = n; i != 0; --i; result *= 2; )
}
```
Breaking the Loop!

```java
public static int powerOf2 (int n) {
    int result = 1;
    for (int i = n; i > 0; i = i - 1) {
        if (i == 0) {
            break;
        }
        result = 2 * result;
    }
    return result;
}
```

Recursion
- Method occurs in its own definition
- New storage is allocated with each call

```java
public static int recursivePowerOf2 (int n) {
    if (n == 0) {return 1;}
    else { return 2 * recursivePowerOf2 (n-1); }
}
```

**BASE part**

**RECURSIVE part**

**DIRECT RECURRENCE**
```java
public static int rabbits (int n) {
    if (n == 0 || n == 1) {return 1;}
    else { return rabbits(n-1) + rabbits(n-2); }
}
```

**INDIRECT RECURRENCE**
```java
public static int rabbits (int n) {
    if (n == 0 || n == 1) {return 1;}
    else { return previous(n) + penultimate(n); }
}
```
```java
public static int previous (int n) { return rabbits(n-1); }
```
```java
public static int penultimate (int n) { return rabbits(n-2); }
```
Some like it Recursive

• Some prefer recursions
  – Inherent elegance
• Some prefer iterative definitions
  – Can you guess why?

Multiway Conditional Statements

• Keywords: switch, case, default
• Statements terminated by break or return, otherwise statement falls through to next set of statements.

```java
public static int rabbits(int n) {
    switch (n) {
        case 0: return 1;
        case 1: return 1;
        default: return rabbits(n-1) + rabbits(n-2);
    }
}
```

EXERCISE

• You have been hired by Gringotts to write a program that counts the number of possible ways to make change (in coins) for a given amount of money. Ultimately, you want a method that takes two arguments:
  • number of sickles, # of types of coins allowed.
  • E.g. countChange(175, 3) would ask how many ways there are make change for a 175 knuts, using sickles, knuts and galleons.
• Design an algorithm and Implement it.