Computer Science 121 - Lecture 6

Announcements:

Ch 4 Embedded problems due?

Programming Assignment 2 due Friday at 5 PM

Programming Assignment 3 up, due next Friday

Third OWL assignment up, due Monday, 10 AM

Fourth OWL assignment due (next) Thursday, 10 AM

Midterm posted (W evening 10/14)

TA Office hours:

M 1-4; TU 3:45-5; W 4-6 (4-5:15 until 10/10); TH 2-5; F 12:15 - 3:15; LGRT 223
Files and Programming Assignment Two

All in directory Users/moll/CS121
Conditional, Looping Statements in Java

Conditional statements and looping statements are **flow of control** constructions.

At a primitive level, Java programs are made up of statements, and it often makes sense to

1) have statements repeat in a systematic way; and
2) have statements execute conditionally.
Conditionals first - Consider:

if (n % 2 == 0) System.out.println("n is even");

Lots going on here: statement says: “if the remainder after dividing n by 2 is equal to (==) 0, then report that n is an even number

Master this difference:

(n % 2 == 0)   #1

n = 0;          # 2
if (n % 2 != 0) System.out.println("n is odd");
else System.out.println("n is even");

An important point: (n % 2 == 0) is a boolean expression (returns a boolean value) --
a boolean must go into the test slot of an if stmt!

Nothing else will do!!
if (n % 2 == 0) System.out.println("n is even");

if (n % 2 == 0) {
    System.out.println(n);
    System.out.println("is even");
}

block
Shorthand operators..

`j++` ;  ->  `j = j + 1;`

`j--` ;  ->  `j = j - 1;`

`sum += j;`  ->  `sum = sum + j;`
The for loop - repetitive control in java

```java
for(int j = 0; j < 3; j++){
    System.out.println(j);
}
```

Prints:
0
1
2
The for loop

```java
for(int j = 0; j < 100; j++){
    System.out.println("I will not talk in class");
}

for(int j = 3; j < 12; j = j + 2){ // 3,5,7,9,11
    System.out.println(j);
}

for(int j = 30; j > 20; j = j - 2){
    System.out.println(j);
}

Prints 30,28,26,24,22
```
for(int j = 3; (j < 12); j = j*j){ // 3,9
    System.out.println(j);
}

for(int j = 3; (j != 12); j = j + 2){ // ??
    System.out.println(j);
}
General form: must be boolean

for(initialize; test; increment){
  stmt1;
  stmt2;
  ...
  stmtk;
}

0 or more stmts to be executed...
The cube sum problem - a math puzzle

Which 3 digit numbers equal the sum of the cubes of their digits?

Examples:

\[241 = 2^3 + 4^3 + 1^3 = 8 + 64 + 1 = 73 \neq 241 \text{(no)}\]

\[153 = 1^3 + 5^3 + 3^3 = 1 + 125 + 27 = 153! \text{(yes!)}\]

Are there others??
public class CubeTester{
    public static void main(String[] args){
        SumOfCubes c = new SumOfCubes();
        for(int j = 100; j < 1000; j++)
            if (c.cubeSum(j) == j){
                System.out.println(j);
            }
    }
}

What is cubeSum?
What class does it belong to?
What’s its return type?
class SumOfCubes{

public int cubeSum(int k){
    int sum, cur, ones, tens, hundreds;
    cur = k;
    ones = cur % 10; // one's place
    cur = cur / 10;
    tens = cur % 10;
    cur = cur / 10;
    hundreds = cur;
    return (cube(ones) + cube(tens) + cube(hundreds));
}

public int cube(int k){ return (k*k*k); }
}

--- attributes? constructor?
k = 365;

cur = k; // cur = 365
ones = cur % 10; // ones = 5
cur = cur / 10; // cur = 36
tens = cur % 10; // tens = 6
cur = cur / 10; // cur = 3
hundreds = cur; // hundreds = 3
Looping Patterns

Generate and test

List nums between 220 & 320 divisible by 37
for(int j = 220; j <= 320; j++)
    if((j % 37) == 0) System.out.println(j);
Accumulate

add nums between 220 & 320 not divisible by 23
int sum = 0;
for(int j = 220; j <= 320; j++)
    if((j % 23) != 0) sum = sum + j;
System.out.println(sum);
Filter

Read 20 nums, sum ones greater than 10
import java.util.Scanner;
public class SumWork {
    public static void main(String[] args) {
        Scanner r = new Scanner(System.in);
        int sum = 0;
        int num = 0;
        for(int j = 0; (j < 10); j++){
            num = r.nextInt();
            if (num > 10) sum += num;
        }
        System.out.println(sum);
    }
}

Note: sum += num    ->    sum = sum + num;
Innocent until proven guilty

*Given a String s, does s contain the char 'a'?*
import java.util.Scanner;
public class CharWork {
    public static void main(String[] args) {
        Scanner r = new Scanner(System.in);
        String s = r.next();
        boolean aPresent = false;
        for(int j = 0; (j < s.length()); j++) {
            if (s.charAt(j) == 'a') aPresent = true;
        }
        System.out.println(aPresent);
    }
}

Is there an int value between to arbitrary int values, a and b, that is divisible by 47?
We’re interested in creating a class that makes forming patterns of rows of stars or other symbols easy - patterns like these:

***********
***********
***********
***********
***********
***********
***********

or

*
**
***
****
*****
******
public class Rows{
    private char sym;
    private int width;
    final char BLANK = ' '; // a constant!

    public Rows(char s, int w){
        sym = s;
        width = w;
    }

    public char getSym(){
        return sym;
    }

    public int getWidth(){
        return width;
    }
}
public void makeRow()
{
    for(int j = 0; j < width; j++)
        System.out.print(sym);
}

public void varyRow(int k){
    for(int j = 0; j < k; j++)
        System.out.print(sym);
}

public void spacedRow(){
    for(int j = 0; j < width; j++)
        if (j % 2 == 0) System.out.print(sym);
        else System.out.print(BLANK);
}

public void newLine(){System.out.println(); }
}
public class RowTester{
    public static void main(String[] args){
        Rows r = new Rows("*",5);
        for(int j = 0; j < 5; j++){
            // makes 5 rows of 5
            r.makeRow();
            r.newLine();
        }
        for(int j = 0; j < 5; j++){
            // makes a triangle
            r.varyRow(2+j);
            r.newLine();
        }
        for(int j = 0; j < 5; j++){
            // makes spaced rows
            r.spacedRow();
            r.newLine();
        }
    }
}
The Math class and static methods
• Not all methods are invoked by objects.
• The Math class, for example
• Doesn’t make sense to attach a funcion like sqrt or max to an object
• All Math functions are static, Math.sqrt() etc.

• One unusual function in the class:
  Math.random()
Gives a random value \( r, 0 \leq r < 1 \)
What does random mean? Basically: generating many in a row will reveal no pattern...
Area of quarter circle = $\pi/4$

Length = 1.0 - the unit square

Area of circle = $\pi \times 1.0 \times 1.0 = \pi = 3.14159$
public class RandomPi{
    public static void main(String[] args){
        Scanner s = new Scanner(System.in);
        int trials = 0; int inside = 0;
        System.out.println("Enter number of trials");
        trials = s.nextInt();
        System.out.println("Trials:" + trials);
        double x,y;
        for(int j = 0; j < trials; j++){
            x = Math.random(); y = Math.random();
            if ((x*x + y*y) < 1) inside++;
        }
        System.out.println("pi guess: " +
            (double)(4*inside)/trials);
    }
}
Enter number of trials
Trials: 100000
pi guess: 3.14116
-----------------------------------
Enter number of trials
Trials: 10000000
pi guess: 3.1415424
Binary conversion - b is a binary string

    int pow = 1;
    int total = 0;
    for(int j = b.length()-1; j >= 0; j--){
        if (b.charAt(j) == '1') total += pow;
        pow = 2*pow;
    }

At the end, total holds decimal integer representation of b
Compound interest

Suppose you have $10,000 in the bank, at 12%. If interest is compounded yearly, you will earn $1200 at the end of the year - for a total of $11,200.00

If interest is compounded every six months, you will earn $600 on July first, and then, at the end of the year, you will earn

1) Another $600, +

2) 6% on the mid-year interest payment, $36

So for the whole year, you will earn $1236 (better!)

If interest is compounded monthly, you will earn $100 at the end of the first month, but $101 at the end of the second month, and so forth - at the end of the year, you’ll have

$11,268.25 (still better!)

(best possible : 11,274.96)
10000 \times (1 + .12) \rightarrow \text{yearly}

10000 \times (1 + .06)(1 + .06) \rightarrow \text{six-monthly}

---first 6 mos ----|--

10000 \times (1 + .03)(1 + .03)(1 + .03)(1 + .03) \rightarrow \text{quarterly}

This is:

10000 \times (1 + .12/4)^4

10000 \times (1 + .01)(1 + .01) \ldots \times (1 + .01) \rightarrow \text{monthly}

---first month ----|--11 more factors-----|--

This is:

10000 \times (1 + .12/12)^{12}
Decimal Formatting for Money

```java
import java.util.*; import java.text.DecimalFormat;

public class MoneyDemo{

    public static void main(String[] args){
        DecimalFormat d =
            new DecimalFormat("$#,###.00");

        Scanner s = new Scanner(System.in);
        System.out.println("enter amount");

        double amt = s.nextDouble();
        System.out.println(d.format(amt));
    }
}

enter amount (12345.67)
$12,345.67
```