Computer Science 121 - Lecture 8

Announcements

Ch 4 OWL assignment due Saturday, 11 PM

Ch 5 embedded problems due morning of 10/7

Programming Assignment 3 due Monday at 11 PM - for hand in, no comments in “import” zone.

TA Office hours: see website, CourseAdministration. Note: Office hours today end at 5. So this week:

Thur 2:30-5; Fri 9-5.

Remember to use the learning resource center, see details, schedule at umass.edu/lrc

SI review session Thursday 8:45-10 (1349 main library);
Sun 8:15-9:30 (1085 main library)
integer division

5/4 = 1
5/9 = 0
9 / 5 = 1 (but note: 9 / 5.0 = 1.8)

remainder - mod - %

5 % 3 = 2 (remainder, after dividing 5 by 3)
5 % 9 = 5 (remainder, after dividing 5 by 9)
9 % 5 = 4 (remainder, after dividing 9 by 5)
Conditional, Looping Statements in Java

Conditional 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 have

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

```java
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

```java
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!!
The for loop - a control line, a body

```java
for(int j = 1; j <= 5; j++){
    System.out.println(j + " " + j/3 + " " + j%3);
}
```

Ans: 4/3 = 1 5%3 = 2

1 0 1
2 0 2
3 1 0
4 1 1
5 1 2
```java
for(int j = 3; j < 12; j = j + 4){
    System.out.println(j);
}

for(int j = 30; j > 20; j = j - 3){
    System.out.println(j);
}
```
General form:

for(initialize; test; increment){
    bunch of statements to be executed
}

String s = "blah-blah";
for(int j = 0; j < s.length(); j++){
    System.out.print(j + " ");
    System.out.println(s.charAt(j));
}
for(int j = s.length() - 1; j >= 0; j--){
    System.out.print(j + " ");
    System.out.println(s.charAt(j));
}
Boolean connectives

&& (and)     || (or)     ! (not)

if ((n < 5) || (n > 10)) System.out.println("hi");
if((n == 7) && (k > 12)) System.out.println("bye");

boolean b = true;
for(int j = 0; j < 5; j = j + 1){
    b = !b;
    System.out.println(b);
}
The loop as scoreboard - a more complex generate & test.

How many numbers between 1 and 100 are divisible by 3 or 7?

```java
int counter = 0; // initialized outside of loop
for(int i = 1; i <= 100; i = i+1){
    if ((i % 7) == 0 || (i % 3) == 0)
        counter = counter + 1; // incremented inside loop
}
System.out.println(counter); // report outside of loop
```
Add up the numbers from 1 to 100 - accumulator pattern

```java
int counter = 0;
for(int i = 1; i <= 100; i = i+1){
    counter = counter + i;
}
System.out.println(counter);
```
import java.util.Scanner;

public class AddUp{
    public static void main(String[] args){
        Scanner s = new Scanner(System.in);
        System.out.println("Enter start,stop nums");
        int first = s.nextInt();
        int last = s.nextInt();
        int sum = 0;
        for (int n = first; n <= last; n=n+1){
            sum = sum + n;
        }
        System.out.println("sum from " + first + " to " + last);
        System.out.println(": " + sum);
    }
}
Some syntax:

```
for(init; test; increment)
    stmt;
```

Or

```
for(....; ....;....){
    stmt1;
    stmt2;
    ...
    stmtn;
}
```
Some syntax:

if (boolean)

    stmt;

Or

if(boolean){

    stmt1;
    stmt2;
    ...
    stmtn;

}
Some syntax:

```plaintext
if (boolean)
    stmt;
else stmt;.....  or  else {smt1...stmn}
or
if(boolean){
    stmt1;
    stmt2;
    stmtn;
}
else stmt;.....  or  else {smt1...stmn}
```
Write a complete program that reads in a String, then reports if there are strictly more a’s (or A’s) than b’s (or B’s).

“abbaAAaaAabbbB” -> true (7/6)
“monkeybusiness” -> false (0/1)
“junkie” -> false (0/0)
import java.util.*;

class AB{
    public static void main(String[] args){
        Scanner sca = new Scanner(System.in);
        System.out.println("enter a phrase");
        String s = sca.nextLine();
        s = s.toLowerCase();
        int ctrA = 0; int ctrB = 0; char c;
        for(int j = 0; j < s.length(); j++){
            c = s.charAt(j);
            if (c == 'a') ctrA++;
            if (c == 'b') ctrB++;
        }
        System.out.println("more a's?? " + (ctrA > ctrB));
    }
}

import java.util.*;

public class AB{
    public static void main(String[] args){
        Scanner sca = new Scanner(System.in);
        System.out.println("enter a phrase");
        String s = sca.nextLine();
        s = s.toLowerCase();
        int ctr = 0; char c;
        for(int j = 0; j < s.length(); j++){
            c = s.charAt(j);
            if (c == 'a') ctr++;
            if (c == 'b') ctr--;
        }
        System.out.println("more a's?? " + (ctr > 0));
    }
}
The English character set (letters, digits, etc) are laid out in a row, every letter has a fixed position.

Rule: arithmetic involving char data is always converted automatically to numerical data:

'A' + 30  \rightarrow  95

'A'\times'A' = 4225

You can cast to get back a character:

(char)95  \rightarrow  -  (a dash, or hyphen)

(char)4235 \rightarrow  'ₘ'  (Arabic? Hebrew?)
for(char ch = 'a'; ch < 'f'; ch = (char)(ch + 1)) {
    System.out.print(ch);
}

Comparing chars: chars come in fixed order, each char has a position in the order
What is ('a' + 1) (ans: 98) / ('a' < 50) is false
What is (char)98 (ans: 'b')
What is ('B' - 'b') (ans: 32) - huh??

If arithmetic operators appear in expressions involving chars, the chars are treats as ints!
for(char ch = 'A'; ch < 'z'; ch = (char)(ch + 1)){
    System.out.print(ch);
}

Output:

ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmnopqrstuvwxyz
Caesar Cipher
Used by Caesar himself to communicate with his generals.
A simple linear shift cipher - nowadays - useless.
So if the shift amount is 2

ZOO → BQQ

Shift of 5? 26? 31?
Key idea:

If letters fall at positions 0 - 25, then z is at 25, a is at 0, b at 1, e at 4, m at 12, o at 14.

If z is shifted 2, it goes to z -> a -> b
If z is shifted 5, z -> a -> b -> c -> d -> e

Use mod (%) to do the wrap:

(‘z’+2) % 26  -->  27 % 26  -->  1  =  ‘b’
(‘z’+5) % 26  -->  30 % 26  -->  4  =  ‘e’
(‘z’+31) % 26  -->  56 % 26  -->  4  =  ‘e’
public class CipherTest{
    public static void main(String[] args){
        Scanner sca = new Scanner(System.in);
        System.out.println("enter a phrase");
        String s = sca.nextLine();
        System.out.println("enter a shift amount");
        int k = sca.nextInt();
        Cipher c = new Cipher(k);
        System.out.println(s);
        System.out.println(c.encrypt(s));
    }
}
public class Cipher{
    private int shift;

    public Cipher(int s){
        shift = s;
    }

    public String encrypt(String plain){
        plain = plain.toUpperCase();
        String result = "";
        for(int j = 0; j < plain.length(); j++){
            result = result + letterShift(plain.charAt(j));
        }
        return result ;
    }
}
private char letterShift(char c){
    // expects c to be either a non-letter or upper case
    if (Character.isLetter(c)){
        int cPos = (c - 'A'); // a position from 0 to 25
        cPos = (cPos + shift) % 26; // deal with overflow
        return ((char)('A' + cPos)); // remake as caps
    }
    else return c;
}
}
Binary conversion - b a binary string

10011

16s 8s 4s 2s 1s - these are powers of 2

Decimal value: 16 + 2 + 1 = 19
int pow = 1;
int total = 0;
for(int j = b.length()-1; j >= 0; j--){
    if (b.charAt(j) == '1') total = total + pow;
    pow = 2* pow;
}

At the end, total holds decimal integer representation of b
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 function like sqrt or max to an object
• All Math fns are static, Math.sqrt() etc.
  Character.isLetter(..) is also a static call (very useful for program #3)
double r = Math.max(3.5, 7.1);

double r = Math.sqrt(2.0);

double r = Math.sin(.7);

double r = Math.min(3.5, 7.1);

double r = Math.pow(2, 5);  // 2^5 returned as double
An unusual function in the Math 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...
A preliminary problem:

Out of 100 random numbers, how many are less than 0.5?
More ambitious:

Create an application that considers some large number of random numbers and checks to see if the average of those numbers is 1/2 (0.5).
Random numbers to estimate pi
(aside: Math.PI)
If a circle has radius 1.0, what is its area?

Suppose the surface of the earth is 77% water, 23% land, and a meteor hitting earth is equally likely to hit anywhere.

If 100 hit earth in the last decade, how many would you expect to land on water?

If 41 / 1000 in last century landed in fresh water, what can you conclude?
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 (Math.sqrt((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