Bring a few blank copies of the survey and ask if there are students

who haven't completed the survey yet.

--------------------------------------------------------------------------

Announce these reminders for students:

1) Office hours are underway -- check the schedule and come see us!

2) We have posted "study questions" on the class webpage.

The authentication is

login: 131

password: student

3) Remind them that although laptops are banned from my lectures, they should

plan to bring them to the lab sessions -- soon they will be working on

programming exercises in the lab.

-------------------------------------------------------------------------------

THE MATERIAL FOR TODAY'S DISCUSSSION IS REVIEWING THINGS THAT I HAVE

ALREADY EXPLAINED DURING THE LECTURE. IT IS NOT MANDATORY THAT YOU COVER

ALL OF THIS MATERIAL, JUST DO THE THINGS YOU FEEL ARE MOST IMPORTANT AND

BE SURE TO USE ALL OF THE CLASS TIME!

-------------------------------------------------------------------------------

Review the formula for computing the number of combinations one could

represent with n bits.

(2^n).

Ask them to consider the opposite question:

If I want to represent, say, 1 million different combinations, how many

bits would I need (log base 2 of 1 million, or 20 bits. 2^n = 1 million,

solve for n).

Related review questions:

Remind them that the old standard for text (ASCII)

was one byte for each character. How many different characters

could be represented (256).

The new standard (Unicode) uses two bytes for each character.

How many combinations are possible now? (2^16 = 65536).

Suppose I want to store ONLY codes for capital letters A through

Z. How many bits would I need? (5 -- that gives 32 possible

combinations, which is more than 26.) So, for example, 'A'

could be stored as 00000, 'B' could be 00001, etc.

Some languages use 4 bytes to store integers. How many combo's

does this give us? (2^32 = About 4 billion). If half of the

integers to be stored are positive and half are negative, what

range of values can be stored this way (about minus 2 billion

to plus 2 billion).

Some languages use 8 bytes to store integers. Do the same

analysis.

-------------------------------------------------------------------

Please remind them that the Java compiler

translates their source code into "Java byte code", which is a

universal (platform independent) proprietary format kind of like

assembly language.

Also remind them that when you run a Java program, the JVM acts

as an interpreter, translating the byte-code into native machine

code as the program runs.

-------------------------------------------------------------------

Please review the following additional topics:

1) primitive types - what are they, how many bytes does each take,

what's the purpose of each group (integers: byte, short, int,

long; foating point: float, double; character; boolean)

[The amount of memory required by type boolean is complicated,

so don't bother talking about that.]

2) print/println distinction

3) For each example below, I suggest writing the code on the board

ask them if it is valid if it is a statement or what type it is

if it is an expression.

Talk about why it is invalid (if it is) then type it into Eclipse

(using the projection from the desktop machine at the front of the

room) and allow eclipse to tell you what is wrong. Then fix it.

If you are doing it all interactively in eclipse, try to make sure

they see the error before eclipse tells them that there is an error.

Notice we have only talked about local variables and a main method -

please don't confuse them by writing additional methods or declaring

variables outside the main metbod.

Some common mistakes for discussion:

---------------

int x, y = l0; //Assuming this sets both to ten.

---------------

x = 3;

int x; // declaring it after it is used

---------------

int x = 5;

int y = 7;

int x = y\*2; //redeclaring a variable

---------------

int x = 5;

System.out.println("2 + x"); //using string literals incorrectly

// change this to System.out.println(2+x);

// and to System.out.println(2 + "x");

// and to System.out.println("2" + x);

----------------

String x = "3.0";

String y = "2.5";

System.out.println(x + y);

// explain why this is string concatenation

-----------------

int x = 2, y = 5;

System.out.println("Sum = " + x + y);

// Why isn't this 7? (+ goes from left-to-right, so the first plus

// concatenates "Sum = " with the String "5", then that result

// is concatenated with the String "2".

-----------------

string x;

Int y;

----------------

int x = 5;

x = x \* 2;

System.out.println(x + "doubled is" + x);

//Talk about:

// why it doesn't give you the sentence 5 doubled is 10

// why the spaces before the d and after the s would make it better

// why this type of assignment (same variable on right and left) is ok

----------------------------------------------------------------------

In whatever time remains, have them practice some base conversion

problems. The most important cases are from base 10 to base 2; from

base 2 to base 10; from base 10 to base 16 and from base 16 to base 10.

USE THE WHOLE 50 MINUTE TIME!!