CSE8A Lecture3:

THE HARDEST CONCEPT OF THE QUARTER!!
(Variable assignment and memory models)

• TO DO:
  – Submit PSA1 if you haven’t yet. You WON’T be charged a slip day, deadline is extended 1 day!
  – Get an interview individually (B320)
  – Don’t forget to start soon with PSA2
PSA Interviews: Spread Out

• Some PSA’s you will need to complete a 5 minute interview in the open lab (B230) with a tutor
  – See the course website for scheduled hours. You have one day left. Go now!

• You should login and open the code from your assignment
  – If code is in your partner account, you need to email it to yourself and put it in your account.

• Then sign up on list on board for a 3-5 minute interview
  – We want to know if you know how your code works
  – We want to you develop professional communication skills for talking about code
Advice: READING!

• There is very important information in the textbook you are responsible for knowing.
  – I won’t talk about all of it in class, if you CAN learn it from the reading

• In class, you’ll need to have LEARNED from the reading in order to learn effectively from the clicker questions!
  – Which emphasize the book material that is hard to learn
Reading Quiz

CLICKERS OUT
the "equals" operators

SET equals isn't equal to IS equals
names data

$$\begin{aligned}
& \text{int } x = 41; \\
& \text{int } y = x + 1;
\end{aligned}$$

Aiden, Braden, Kaden…?
Ava, Abigail, Caylin…?

x and y are "variables"
Don’t confuse them with variables from math
In Java, variables store data
assignment, not equality!

= is an ACTIVE, DIRECTIONAL operator. It means:

“First calculate the value on the right hand side, and then put it into the box labeled with the name from the left hand side (replacing what was there, if necessary).”

It does not test for equality (that’s ==).

```
>> int x = 41;  "Put 41 into the box labeled x"
>> int y = x + 1; "Get the value out of x (41), and add 1 to it (42). Put that value (42) into the box labeled y"
```
```java
>> int x = 41;
>> int y = x + 1;
>> System.out.println(x);
41
>> System.out.println(y);
42
>> x = x + y;
>> System.out.println(x);
?? (1)
>> System.out.println(y);
??
```

"Find the value in x and add it to the value in y. Then place that value back into x, replacing what was there."

What value is displayed for x at ??(1)?

A. 41  
B. 42  
C. 83  
D. 84

Shorthand for the slide: NOT LEGAL JAVA!!!
```
>> int x = 42;
>> int y = x;
>> x = 101;
>> S.o.p(x);
??
>> S.o.p(y);
??
```

What values are displayed for x and y?

<table>
<thead>
<tr>
<th>x</th>
<th>y</th>
</tr>
</thead>
<tbody>
<tr>
<td>A.</td>
<td>42</td>
</tr>
<tr>
<td>B.</td>
<td>101</td>
</tr>
<tr>
<td>C.</td>
<td>101</td>
</tr>
<tr>
<td>D.</td>
<td>None of these</td>
</tr>
</tbody>
</table>

When in doubt, draw it out!!
More Practice with Data and Assignment

• What is the output of this code?

```java
int x = 3;
int y = 2;
int z = 2;
System.out.println(x == 3);
z = x + y;
System.out.println(z);
```

A) 3  B) true  

5  5

E) None of  
the above.

C) 3  D) true  

2  2
What is the output of this code?

```java
int x = 3;
Turtle y = 2;
System.out.println(x == 3);
int z = x + y;
System.out.println(z);
```

A) 3  
5  
B) true  
5  
C) 3  
false  
D) true  
false  
E) None of the above.
Summary (so far): Data Types and Assignment

```java
int x = 3;

Turtle y = new Turtle(myWorld);
```

Declared type must match assigned data

Declaration and assignment on separate lines OK:

```java
Turtle y;
y = new Turtle(myWorld);
```

Reassignment (without redeclaration) OK, even if variable is on RHS!

```java
int x = 3;
x = x + 1;
```

In java, all variables have an explicit type!
CS Concept: References

• What does this code draw?

World world1 = new World(200,100);
Turtle maria = new Turtle(25, 25, world1);
Turtle jose = new Turtle(100, 50, world1);
maria = jose;
maria.forward(50);
jose.turn(90);
jose.forward(50);

1) SOLO VOTE 
(30 secs)
2) Discuss in team 
(1 min)
3) GROUP VOTE 
(20 sec)
World world1 = new World(200, 100);
Turtle maria = new Turtle(25, 25, world1);
Turtle jose = new Turtle(100, 50, world1);

maria = jose;
World world1 = new World(200,100);
Turtle maria = new Turtle(25, 25, world1);
Turtle jose = new Turtle(100, 50, world1);

maria = jose;

The arrows in maria and jose’s boxes are just graphical representation of the reference (i.e., location of) the object in memory.
World world1 = new World(200, 100);
Turtle maria = new Turtle(25, 25, world1);
Turtle jose = new Turtle(100, 50, world1);

The arrows in maria and jose’s boxes are just graphical representation of the reference (i.e., location of) the object in memory.
CS Concepts: References

World world1 = new World(200,100);
Turtle maria = new Turtle(25, 25, world1);
Turtle jose = new Turtle(100, 50, world1);

The arrows in maria and jose’s boxes are just graphical representation of the reference (i.e., location of) the object in memory.
World world1 = new World(200, 100);
Turtle maria = new Turtle(25, 25, world1);
Turtle jose = new Turtle(100, 50, world1);

The arrows in maria and jose’s boxes are just graphical representation of the reference (i.e., location of) the object in memory.
World world1 = new World(200, 100);
Turtle maria = new Turtle(25, 25, world1);
Turtle jose = new Turtle(100, 50, world1);

The arrows in maria and jose’s boxes are just graphical representation of the reference (i.e., location of) the object in memory.
World world1 = new World(200, 100);
Turtle maria = new Turtle(25, 25, world1);
Turtle jose = new Turtle(100, 50, world1);

The arrows in maria and jose’s boxes are just graphical representation of the reference (i.e., location of) the object in memory.
World world1 = new World(200, 100);
Turtle maria = new Turtle(25, 25, world1);
Turtle jose = new Turtle(100, 50, world1);

The arrows in maria and jose's boxes are just graphical representation of the reference (i.e., location of) the object in memory.

This holds for all Object types (not for primitive types)

THIS IS THE HARDEST THING YOU WILL LEARN IN CSE8A. MASTER THIS IDEA AND YOU WILL ACE CSE8A (AND MUCH MORE)
CS concept: Writing Methods for Objects

The Turtle class

Generates

Turtle Objects

public void drawSquare()
{
    this.turnLeft();
    this.forward(100);
    this.turnLeft();
    this.forward(100);
    this.turnLeft();
    this.forward(100);
    this.turnLeft();
    this.forward(100);
}

Refers to the object that eventually calls the method

Generates

The Turtle class

Generates

Turtle Objects

maria

(just will become maria)

Refers to the object that eventually calls the method

Generates

The Turtle class

Generates

Turtle Objects

jose

(just will become jose)
Write a more general `drawSquare`

Write a new `drawSquare` that takes an int as a parameter to specify the side length of the square.

```java
public void drawSquare(int sideLength) {
    this.turnLeft();
    this.forward(sideLength);
    this.turnLeft();
    this.forward(sideLength);
    this.turnLeft();
    this.forward(sideLength);
    this.turnLeft();
    this.forward(sideLength);
    this.turnLeft();
    this.forward(sideLength);
}"

Old version for reference:
```
What’s the right way to “call” that new method to get a Turtle to draw a square?

A. 
World w = new World();
Turtle t = new Turtle(10,10, w);
t = drawSquare(50);

B. 
World w = new World();
Turtle t = new Turtle(10,10, w);
t.drawSquare(50);

C. 
World w = new World();
Turtle t = new Turtle(10,10, w);
t.drawsquare();

D. 
World w = new World();
Turtle t = new Turtle(10,10, w);
t = drawsquare();

E. None of the above

1) SOLO VOTE (30 secs)
2) Discuss in team (2 min)
3) GROUP VOTE (30 sec)
Why write methods?

A. To avoid having to copy and paste code

B. To avoid fixing problems (bugs) in more than one place

C. To make it easier for others to use your code

D. All of the above
For Reference: The anatomy of a method

```
public void drawSquare(int size) {
    this.turnLeft();
    this.forward(size);
    this.turnLeft();
    this.forward(size);
    this.turnLeft();
    this.forward(size);
    this.turnLeft();
    this.forward(size);
}
```

- **Method header**
  - Return type: `public void`  
  - Name: `drawSquare`  
  - Parameter list: `(int size)`

- **Curly braces** (identify beginning and end of method)
- **Method body**
- `this` refers to the “calling object” (i.e. the object that the method is eventually called with)