2/2/26

Lecture # 3

In Java, datatypes are either primitive or objects, primitive: int, char, double, float, byte, short, long, boolean.

Value semantics and reference semantics are inherit to Java, and they are different between primitive and object types

Def) Value Semantics for primitive types tells us what happens in computer memory and works in C++

ex) int X = 5; // stored in memory
int y = x; // a copy is stored in memory
// the following statement would print True
System.out.println(X==y);
	// reassign x = 7
	X = 7;
	// y is still equal to 5
		When you pass a parameter to a static function (in Java), e.g. y = square(x); runs the function code under a copy of the provided parameter.
		There are 2 options (for parameter passing in C++): call by value, and call by reference, in Java, only pass by value is allowed.

In Java, data that is not primitive is an object type. e.g ACStudent (but this isn't inherit to Java), but we can set it up ourselves, i.e. determine its attributes and functionality.

Question 1) How do we create object types 2) How do we use object types. Java provides some useful types that store data, and provide methods (special actions) e.g. String (work with text), Scanner (to work with input), PrintWriter (prints in file) (Object types)

Using methods: Scanner: nextInt(), nextDouble(), nextLine() (some example methods) String: length(), indexOf(target) (some examples)

General model in using a method: [specify object] [method choice] ([facts]) (Selection operator)

ex) // read an int from the user (via input)

```
Scanner s; //you will need to import using: java.util.Scanner within your file
S = new Scanner(System.in); // "new" keyword for creating objects.
int X = s.nextInt();
```