CS 201 - Python to Java

Functions and Methods
Operators
For loops
Data Types
  Primitives / References
  Memory Model
Lists / Arrays
Classes

Friday CMC 306

Functions / Methods

In Python

Functions
- named blocks ← denoted by a colon and indent
- take parameters (possibly 0)
- return a value

→ static methods ← denoted by curly braces

Methods
Functions that live inside objects
- have self as a parameter
- use self to access data from the object

→ instance methods
(don't have self but can access data from objects)

↓

Static methods

public static return-type method-name (parameters)
↑ also have types

↓

Guessing game

↓

1. Convert line by line to Java
2. Pull out checking code
3. Turn that into a static function

Operators

```
a = 3
b = 5
a = a * b
b = b + 1
```

→

```
int a = 3
int b = 5
a = a * b
b = b + 1
```

```
a *= b
b += 1
```

```
a *= b
b += 1
```

```
b++;
```

Common operators

+, -, *, /, %...

==

new operators

++

--

logic -> &&, ||, ~~!~~

For loops

```
for ( initialization ; continuation-test ; update ) {
    body;
}
```

NOTES: init only done once
body only executed if Test is true

Factorial program -> Convert
Trace by hand
Show debugger

Data Types

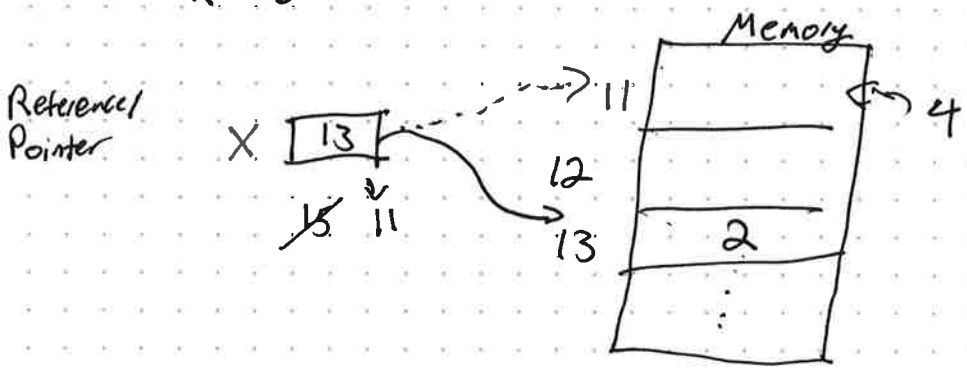
Primitives vs References

Python

- not storing a value in a variable
- In reality

Value is in memory
variable stores the memory address

x = 2



x += 2?

Demo with id()

Java

Primitive types

int
long
boolean

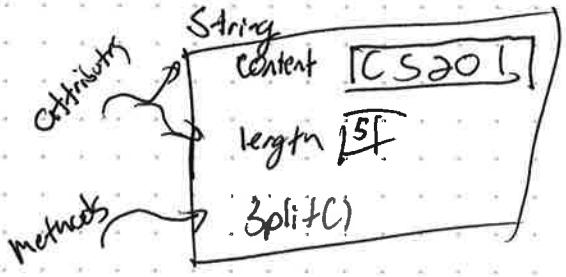
?
really are values

Why do that?
less memory
cheap

Demo

Objects

Group Data (fields or attributes)



Instance of a class

Dogs
vs
My dog

...lots ? Arrays

Python

List sequence of arbitary length with arbitary types

Ex: [2, 4, 6, 8]

['Sam', 'Sneha', 'Layla']

Bytes ['Sam', 4d, {'good': [1]?}]

Java has arrays

Elements must be the same type

- less flexible
- less functionality
- less overhead
- more efficient

Ex: Grades

Convert
Trace on board

Classes

- a blueprint / definition of a data type
  - what values does it contain (fields/attributes)
  - what methods does it have

When you actually build an Object according to the definition.

- ~~def~~ Instances of class

Constructor: __init__ -> public ClassName (NO Return type, not static)

Self -> this

Fields Nothing! -> declared

Rectangle
↓
area
↓
Square

Change dim