“You guys are ahead of all my other groups. Wait, don’t get me wrong, you’re actually pretty far behind.”

- Heather Preslier
Overview

- Imperative
  - Expression vs statement execution
- Strongly typed
  - lval and rval types must match
- Statically typed
  - Variable types must be known at compile time
Architecture
Language Basics

Comments

#: I am a multi
line comment! :#

# I’m a single line comment!

Operators

Arithmetic operators:
( )  *  /  %  +  -

Comparison operators:
==  is  <  >  <=  >=  !=  isnt

Logical operators:
and  &&  or  ||

Unary operators:
not  !  -

Types

Primitive:
num - defaults to 0.0
string - defaults to null
bool - defaults to false

Non-primitive:
Array
Object
Control Flow

If - Else

```java
bool t = true;
if (not t) {
    print("false");
} else {
    print("true");
}
```

For

```java
num i;
for (i = 0; i not 5; i = i + 1) {
    print(i);
}
```

While

```java
num x;
while (x isnt 5) {
    x = x + 1;
}
```
Syntax

Functions

```cpp
num gcd (num x, num y)
{
    if(x == 0) {
        return y;
    }
    while (y != 0) {
        if (x > y) {
            x = x - y;
        } else {
            y = y - x;
        }
    }
    return x;
}

print("GCD Result: ", gcd(240, 150));
```

Objects

```cpp
Object math = {
    num someVal = 54;
    num double(num val) {
        return val * val;
    }
};

math.double(3); // returns 9
```

Arrays

```cpp
Array num a = [5];
```
The Process

Initial goals

Actual results

Problems
Demo

GCD

Fibonacci

Objects