From Python to Java

by example

Example: Gear ratios

class Gears {
    // Print the gear ratio for each combination.
    public static void main(String[] args) {
        int[] RINGS = new int[] {53, 39};
        int[] COGS = new int[] {
            12, 13, 14, 15, 17, 19, 21, 23, 26
        };
        for (int ring : RINGS) {
            for (int cog : COGS) {
                float ratio = (float) ring / (float) cog;
                System.out.format("%d %d %.2f\n", ring, cog, ratio);
            }
        }
    }
}

def main() :
    ... body ...

def main() :
    public static void main(String[] args) {
        ... body ...
    }

Everything is in classes ...

"java Gears" will execute this main method

like System.argv

public: Can be called from outside the class
static: can be called on the class, rather than an object (like @classmethod in Python)
void: doesn’t return a result
Data types

<table>
<thead>
<tr>
<th>Python</th>
<th>Java</th>
</tr>
</thead>
<tbody>
<tr>
<td>integer</td>
<td>int, long</td>
</tr>
<tr>
<td>float</td>
<td>float, double</td>
</tr>
<tr>
<td>string</td>
<td>String</td>
</tr>
<tr>
<td>boolean</td>
<td>bool</td>
</tr>
<tr>
<td>-</td>
<td>char</td>
</tr>
<tr>
<td>list</td>
<td>array (fixed length) or Vector (flexible)</td>
</tr>
<tr>
<td>tuple</td>
<td>-</td>
</tr>
</tbody>
</table>

Declaring and Initializing

```java
int [ ] RINGS = new int [ ] { 53, 39 };

- OR -

int [ ] RINGS;
RINGS = new int [ ] { 53, 39 };
```

RINGS is an array of integers, int [ ];
set its value to an array of integers with these values

Iterating with for-each loops

```java
for (int ring: RINGS) {
    for (int cog: COGS) {
        ...
    }
}
```

Almost the same as in Python.
Note “ring” and “cog” are declared here (with types) and exist only within the loops.

Type conversions

Python:

```
ratio = float(ring) / float(cog)
```

Java:

```
float ratio = (float) ring / (float) cog;
```
**Formatted output**

**Python:**

```python
print ring, cog, "%.2f" % ratio
```

- **print** is a statement (until Python 3.0)
- **formatting** is an operation

**Java:**

```java
System.out.format("%.2d %.2d %.2f\n", ring, cog, ratio);
```

- **formatted output** is a method of **System.out**
- **similar format specs** (inherited from C)

---

**Syntactic differences ...**

```java
class Gears {
    /**
     * Print the gear ratio for each combination.
     */
    public static void main(String[ ] args) {
        int [ ] RINGS = new int [ ] { 53, 39 };
        int [ ] COGS = new int [ ] { 12, 13, 14, 15, 17, 19, 21, 23, 26 };
        for (int ring : RINGS) {
            for (int cog : COGS) {
                float ratio = (float) ring / (float) cog;
                System.out.format("%.2d %.2d %.2f\n", ring, cog, ratio);
            }
        }
    }
}
```

- **comments**
  - // to end of line
  - /* ... */ (multi-line)
  - /** ... */ (javadoc)
- **{ }** marking scopes
  - (in place of indent: ;)
  - ; at end of statement

---

**Javadoc comments**

*Use like Python docstrings, just before a class or method:*

```java
/**
 * What this method does, in one sentence.
 * More description is optional.
 * @param x what the value of x means
 * @param y what the value of y means
 * @return what this method returns
 */
public int foo(int x, float y) {
```

- Java doesn't care about indentation, but I do