# Java Notes

```
public class Hello {

    public static void main(String[] args) {
        System.out.println("Hello World!");
    }

}
```

Primative types and wrappers

| Primitive | Object |
|---|---|
| int | Integer |
| long | Long |
| float | Float |
| double | Double |
| char | Char |
| boolean | Boolean |

Can create Scanner, like `new Scanner(System.in)`. Methods:

| Return type | Method name | Description |
|---|---|---|
| boolean | hasNext() | returns true if more data is present |
| boolean | hasNextInt() | returns true if the next thing to read is an integer |
| boolean | hasNextFloat() | returns true if the next thing to read is a float |
| boolean | hasNextDouble() | returns true if the next thing to read is a double |
| Integer | nextInt() | returns the next thing to read as an integer |
| Float | nextFloat() | returns the next thing to read as a float |
| double | nextDouble() | returns the next thing to read as a double |
| String | next() | returns the next thing to read as a String |
| String | nextLine() | returns all before the next newline |

String manipulation: Examples below use a string variable called “s”

| Python | Java | Description |
|---|---|---|
| `s[3]` | `s.charAt(3)` | Return character in 3rd position |
| `s[2:4]` | `s.substring(2,4)` | Return substring from 2nd up to but not including 4th |
| `len(s)` | `s.length()` | Return the length of the string |
| `s.find('x')` | `s.indexOf("x")` | Find the first occurrence of x |
| `s.split(',')` | `s.split(",")` | Split the string at `','` into a list/array of strings |
| `s.split()` | `s.split("\\s+")` | Split out non-whitespace strings |
| `s + s` | `s.concat(s)` | Concatenate two strings together (can still use +) |
| `s.strip()` | `s.trim()` | Remove any whitespace at the beginning or end |

| Python | Java | Description |
|---|---|---|
| `s.replace("me", "I")` | same as Python | Replace all occurrences of first parameter by second |
| `s.upper()` | `s.toUpper()` | Return string in uppercase; similarly Java toLower |

List methods: Suppose w and w2 are lists of strings --

| Python | Java | notes |
|---|---|---|
| value of `w[3]` | `w.get(3)` | Return item in 3rd position |
| `w[3] = 'a'` | `w.set(3, "a")` | Set item in 3rd position |
| `w.append('a')` | `w.add("a")` | Append item |
| `len(w)` | `w.size()` | Return the number of items in the list |
| `w.find('x')` | `w.indexOf("x")` | Find index of the first occurrence of x, or -1 |
| `w += w2` | `w.addAll(w2)` | add all of list w2, modifying w |
| `'x' in w` | `w.contains("x")` | membership test |
| `not bool(w)` | `w.isEmpty()` | Python w is True if not empty |
| `w[:]=[]` | `w.clear()` | Remove all items |
| `w.pop(2)` | `w.remove(2)` | remove and return item at position 2 |
| `w.sort()` | `w.sort()` | sort |

To separate with `", "` the strings in list `sList`:

`', '.join(sList)` in Python
`String.join(", ", sList)` in Java

Here is a function to return a new array containing the squares of the numbers in a passed array. See the use of the length attribute:

```
public static int[] squares(int[] nums)
{
    int n = nums.length;
    int[] sq = new int[n];
    for (int i = 0; i < n; i++) {
        sq[i] = nums[i]*nums[i];
    }
    return sq;
}
```

Could be called with:

```
int[] vals = {2, 5, 6, 22},
      vals2 = squares(vals);
```

Dict/Hashmap

| Python | Java |
|---|---|
| `d = dict()` | `d = new HashMap<keyType, valueType>()` |
| `d[key] = v` | `d.set(key, v)` |
| `v = d[key]` | `v = d.get(key)` // may be `null` |
| `d.keys()` | `d.getKeys()` |

Print formatted string with fieldwidth, float precision:

Python `print('{:20} and {:7.2f}'.format(a, 2.3456))`
Java `System.out.format("%20s and %7.2f%n", a, 2.3456);`

Boolean Operators

| Python | Java |
|---|---|
| and | && |
| or | \|\| |
| not | ! |

Conditionals

```
if (condition) {
    statement1
    statement2
    ...
} else {
    statement1
    statement2
    ...
}
```

Loops and Iteration

```
for (int i = 0; i < 10; i++ ) {
    System.out.println(i);
}


for (start clause; stop clause; step clause) {
    statement1
    statement2
    ...
}
```

If s is a sequence (of element type `tp` in Java)

```
for e in s:             # Python
  print(e)

for (tp e : s) {          //Java
    System.out.println(e)
}

while (condition) {
    statement1
    statement2
    ...
}
```

Class definitions:

```
class Point:             # Python
    def __init__(self, x, y):
        self.x = x
        self.y = y

    def getX(self):
        return self.x
```

```
        def __str__(self):
            return '({}, {})'.format(self.x, self.y)
```

```
    class Point            // Java
        private int x, y;

        public Point(int x, int y)
        {
            this.x = x;
            this.y = y;
        }

        public int getX()
        {
            return x;
        }

        public String toString()
        {
            return String.format("(%s, %s)", x, y);
        }
```

Interfaces

```
    public interface Response
    {
        boolean execute(String[] tokens);

        String getCommandName();

        String help();

    }
```

Heading for class using Response interface:

```
    public class Goer implements Response
```