

①

Dynamic vs. Static Typing

X = "xyz"
X = 3

String x = "xyz";

For loops

Python

for student in studentlist:
for x in range(0, 4):

Java

for (String s : studentlist) {
}
for (int i = 0; i < N; i++) {
}

Dictionaries built in

Python

var = { "key": 42, "another": 7 }

var["whatever"] = "hi"

var[29] = [3, 4, 5]

var = {}

Java

import Map;

import HashMap;

Map<keytype, valtype> varname = new
HashMap<>();

varname.put(key, value);

Lists

Python

```
varname = []
```

```
x = ["xyz", 3]
```

```
x[2] = 3.4
```

```
x[0]
```

Java

```
List<Integer> mylist = new ArrayList<>();
```

Classes

Python

have to import

~~class~~

~~foo~~:

```
def __init__(self, _____):
```

```
self.field1 = ...
```

foo()

Java only import if different package

```
public class foo {
```

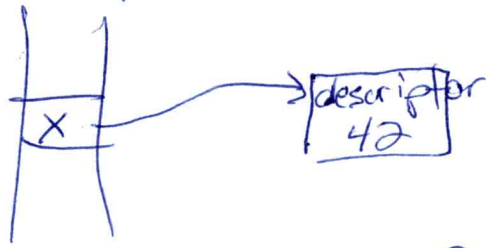
```
int field1;
```

```
int field2;
```

```
public foo() {
```

```
int this.field2 = ...
```

Primitive types vs. Reference types (3)



Java

- primitives are faster
- competing with C++

in python all ref types

Conversions to string

Java "hello" + 42

(int)

Integer.toString()

System.out.println(42)

42 + 30

python

"hello" + str(42)

indentation importance

- Java doesn't care
- Python does

inline vs. multiline comments

Python

inline

""" multiline

"""

Java

// inline

/* multiline */