week11

COMP90041 Programming and software development

Zhe(Zoe) Wang

github: https://github.com/Zoeewang/COMP90041-2020-sem1-tutorial

ArrayList

- an **ArrayList** is an object that can grow and shrink while your program is running

- ArrayList<String> list = new ArrayList<String>(20);

$\uparrow$

Type parameter

Generic Class : allow class declaration to specify parameters

Parameters, enclosed in the **angle brackets**, are variables ranging over types rather than values

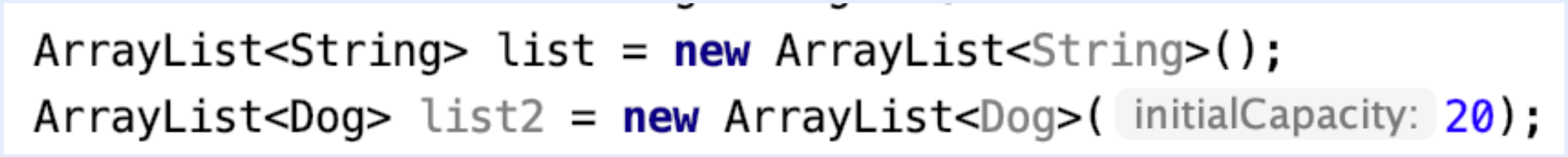
Generic class

ClassName<Type1,…..>

Construct a new object of generic type, specify both type arguments and constructor arguments

Form:

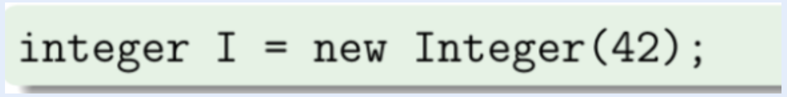
new ClassName<Type1,…>(expr1,…)



```
ArrayList<String> list = new ArrayList<String>();
ArrayList<Dog> list2 = new ArrayList<Dog>( initialCapacity: 20);
```

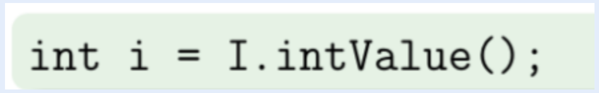
Wrapper class

- A primitive value is not an object
- Each primitive type has a wrapper class that stores one primitive value
- **Boxing :** Each has a one-argument constructor to create the object



```
integer I = new Integer(42);
```

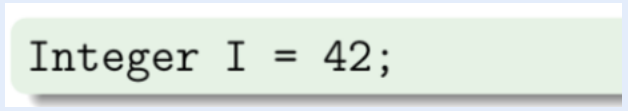
- **Unboxing:** each has a no-argument getter to get back the primitive value



```
int i = I.intValue();
```

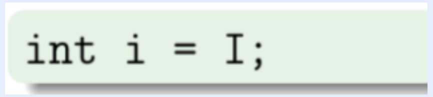
Autoboxing and auto-unboxing

- **Auto Boxing**

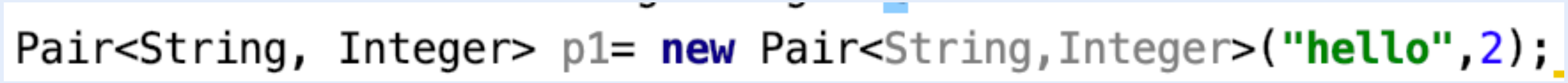


```java
Integer I = 42;
```

- **Auto unboxing**



```java
int i = I;
```



```java
Pair<String, Integer> p1= new Pair<String,Integer>("hello",2);
```

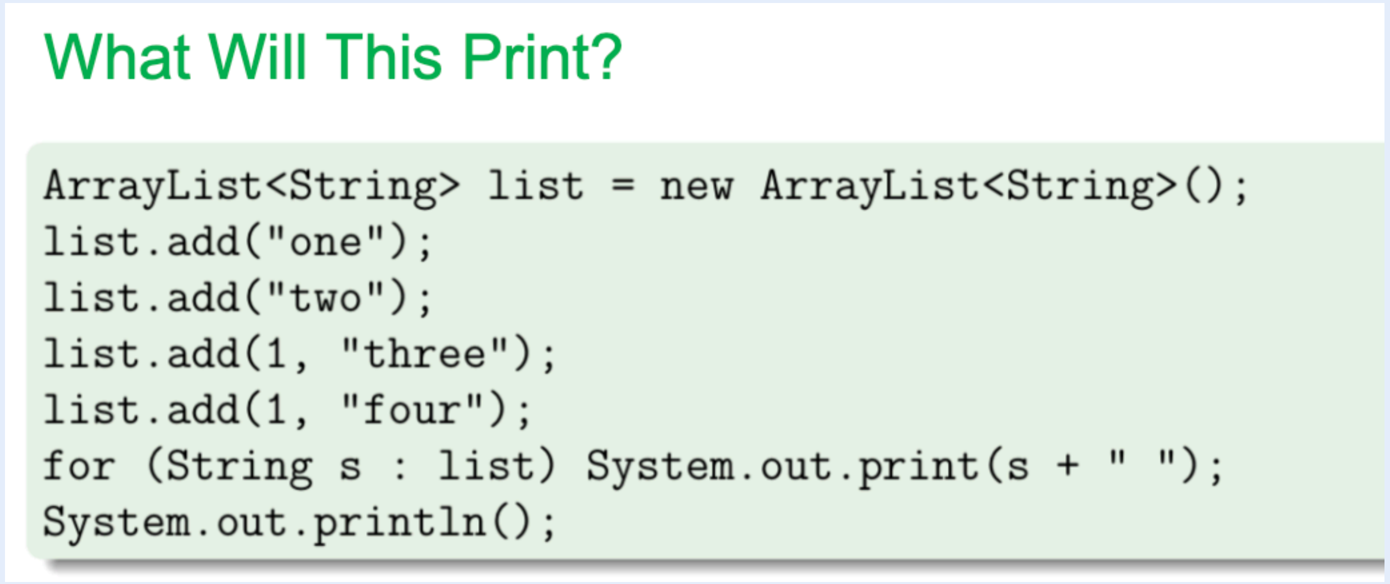
- **Auto Boxing to integer!**

ArrayList Methods

add(E elt): add elt to the end of ArrayList

add(int i, elt): insert elt at index i

Each element in the ArrayList with an index greater or equal to index is shifted upward to have an index that is one greater than the value it had previously.



What Will This Print?

```
ArrayList<String> list = new ArrayList<String>();
list.add("one");
list.add("two");
list.add(1, "three");
list.add(1, "four");
for (String s : list) System.out.print(s + " ");
System.out.println();
```

ArrayList Methods

remove(int index): Deletes and returns the element at the specified index.

Each element in the ArrayList with an index greater than index is decreased to have an index that is one less than the value it had previously.

remove(Object elt): Removes one occurrence of elt from the calling ArrayList.

If there are duplicate elements present in the list it removes the first occurrence of the specified element from the list.

ArrayList Methods

get(int i) : return element at index i

set(int i, E elt): replace obj at index i with elt; return old

indexOf(Object o): return the first index of o, or -1 if absent

int lastIndexOf(Object o): return last index of o, or -1 if absent

Final reflection

Comments for COMP90041

http://go.unimelb.edu.au/n5nj

Comments for this tutorial

https://docs.google.com/forms/d/e/1FAIpQLSfQfAkj30WMC5bAbx44KqT15atrZRjOXFbCZZYtVzJZLgb53g/viewform?usp=sf_link

THE UNIVERSITY OF MELBOURNE

Thank you