THE UNIVERSITY OF MELBOURNE

COMP90041

Programming and Software Development

Semester 1, 2021

Lab 7

Andrew Naughton

andrew.naughton@unimelb.edu.au

THE UNIVERSITY OF
MELBOURNE

Outline

- Inheritance
- `super` Constructor
- Overriding
- Overriding vs. Overloading
- Late binding
- Visibility
- Exercises

Inheritance

- ❖ When we define a new class that extends an existing class
  - ❖ The existing class is referred to as the base/parent class
  - ❖ The new class is referred to as the inherited/child class

- ❖ This concept allows us to build on previous work without reinventing the wheel

- ❖ The inherited class merely needs to specify how it differs from the base class

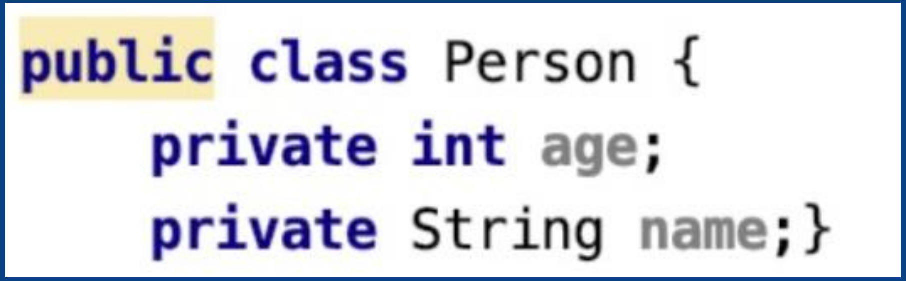
THE UNIVERSITY OF MELBOURNE

Inheritance

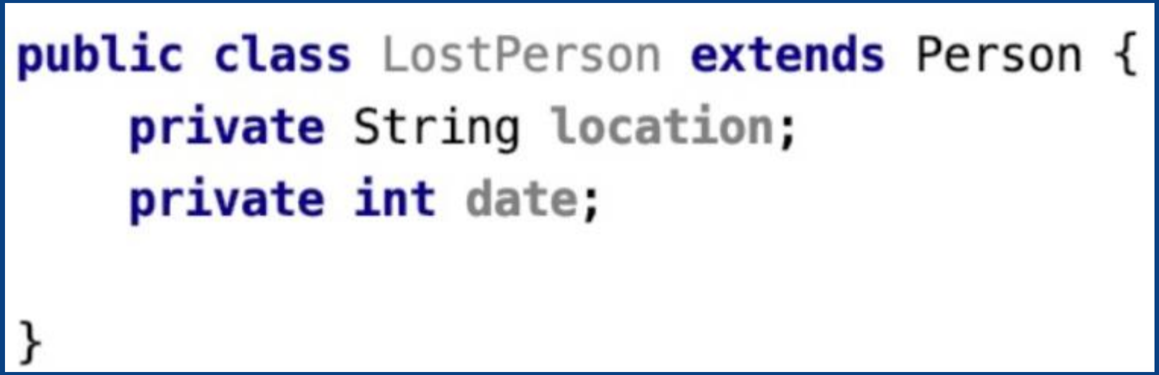
- ❖ LostPerson class inherits all the instance variables and methods of the Person class… and adds its own!

- ❖ No need to mention inherited instance variables and methods

- ❖ Every instance of the inherited class is also an instance of the base class (every LostPerson is a Person)



```
public class Person {
    private int age;
    private String name;}
```



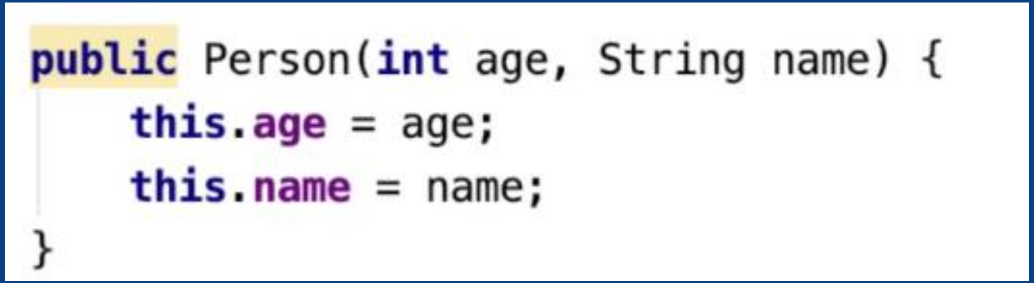
```
public class LostPerson extends Person {
    private String location;
    private int date;


}
```

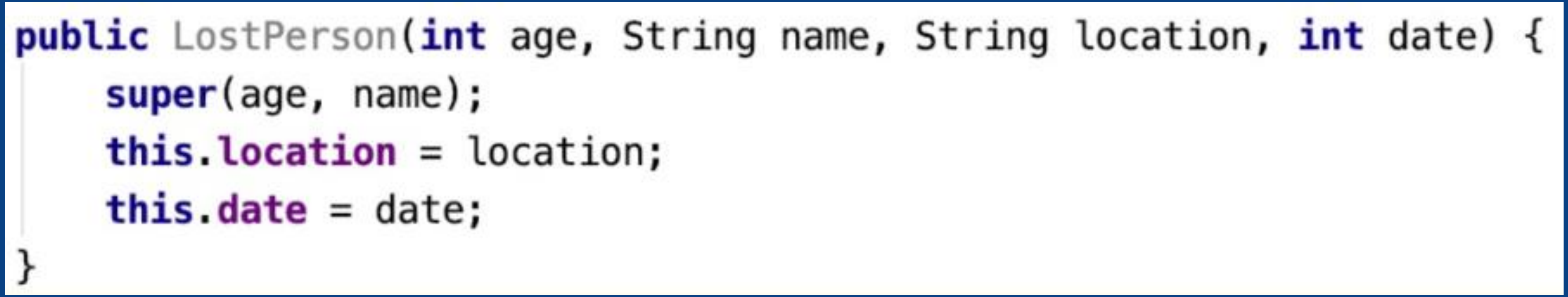
super Constructor

- Constructor are not inherited, and cannot be overridden (redefined)

- Constructor chaining is when the inherited class' constructor(s) invoke the base class' constructor first



```java
public Person(int age, String name) {
    this.age = age;
    this.name = name;
}
```



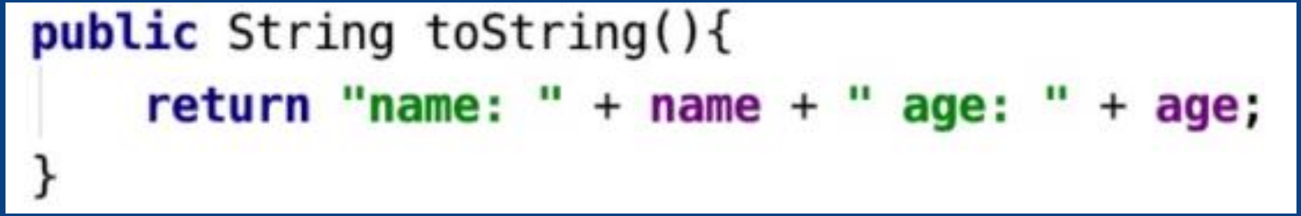
```java
public LostPerson(int age, String name, String location, int date) {
    super(age, name);
    this.location = location;
    this.date = date;
}
```

THE UNIVERSITY OF MELBOURNE

Overriding

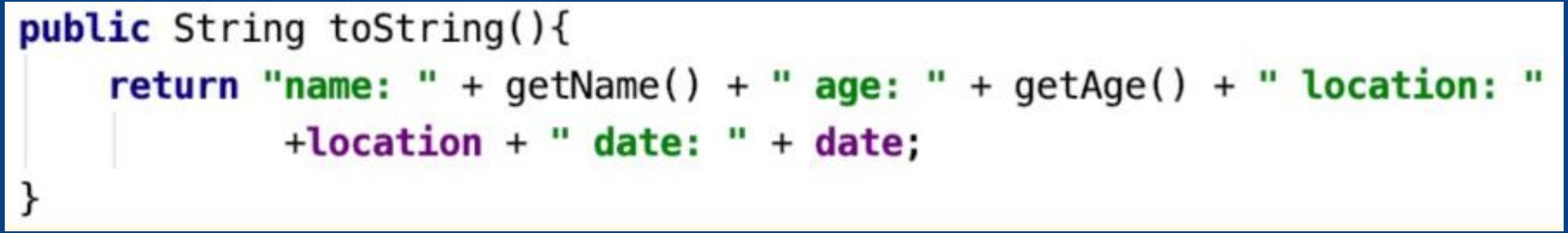
- ❖ If a class defines a method with same signature as an ancestor, its definition overrides the ancestor's

- ❖ Person:



```
public String toString(){
    return "name: " + name + " age: " + age;
}
```

- ❖ LostPerson:

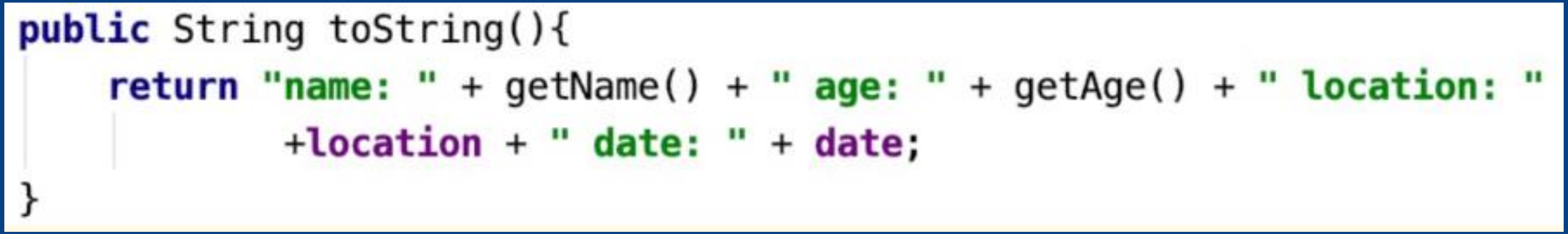


```
public String toString(){
    return "name: " + getName() + " age: " + getAge() + " location: "
            +location + " date: " + date;
}
```

Overriding

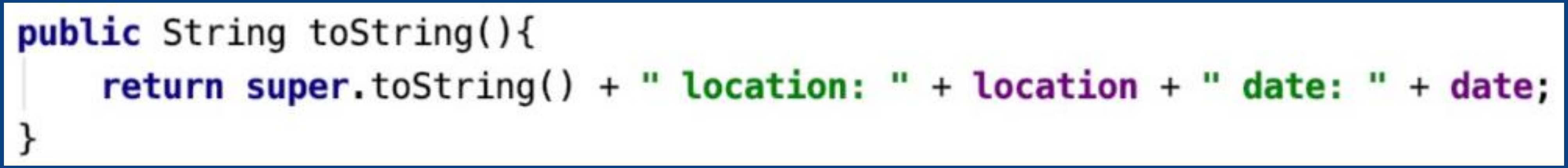
- ❖ We can use overridden methods of our parent with `super.methodName(…)`

- ❖ E.g. This ->



```
public String toString(){
    return "name: " + getName() + " age: " + getAge() + " location: "
            +location + " date: " + date;
}
```

- ❖ Could be ->



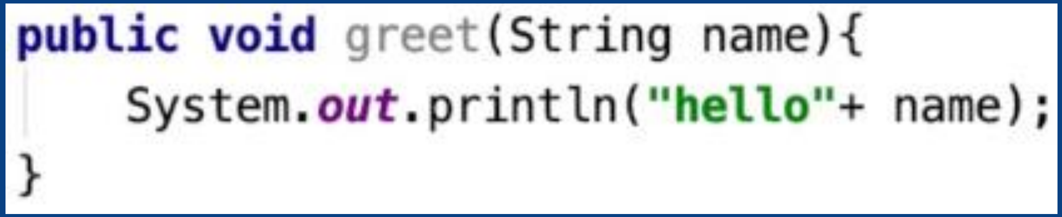
```
public String toString(){
    return super.toString() + " location: " + location + " date: " + date;
}
```

THE UNIVERSITY OF MELBOURNE

Overriding vs. Overloading

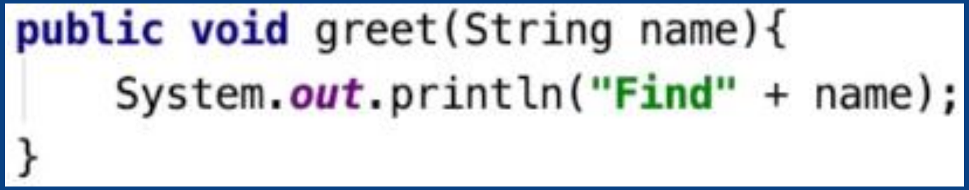
- ❖ Overriding:
- ❖ An inherited class can supply its own implementation for a method that also exists in the superclass

- ❖ Person:



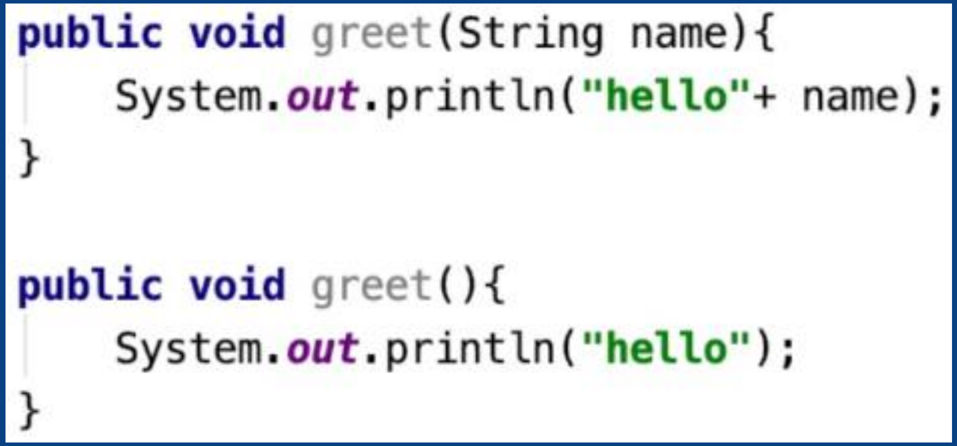
```java
public void greet(String name){
    System.out.println("hello"+ name);
}
```

- ❖ LostPerson:



```java
public void greet(String name){
    System.out.println("Find" + name);
}
```

- ❖ Overloading
- ❖ Two methods have the same name but different signatures



```java
public void greet(String name){
    System.out.println("hello"+ name);
}


public void greet(){
    System.out.println("hello");
}
```

Late binding

$$\underbrace{\textbf{Person}}_{\substack{\textbf{Declared type} \\ \textbf{(what methods} \\ \textbf{available)}}}\ \text{p1} = \textbf{new}\ \underbrace{\textbf{LostPerson}}_{\substack{\textbf{actual type} \\ \textbf{(which method} \\ \textbf{implementation will be used)}}}(\ldots)$$

```
Person person = new LostPerson(60, “Fred”, “Melbourne”, 01012021);
System.out.println(person);
```

❖ Whose toString method is used? (Person / LostPerson)

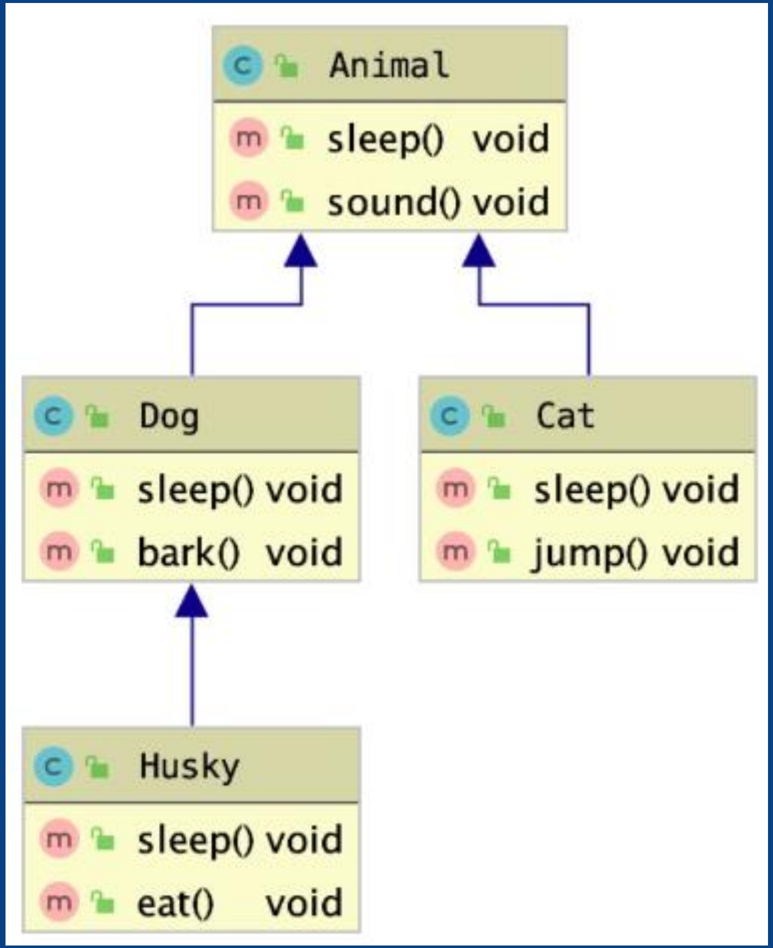
THE UNIVERSITY OF MELBOURNE

Late binding

```
Animal a1 = new Dog();
Animal a2 = new Cat();
Dog d1 = new Dog();
Dog d2 = new Husky();
```

❖ Which of the following statements are illegal?

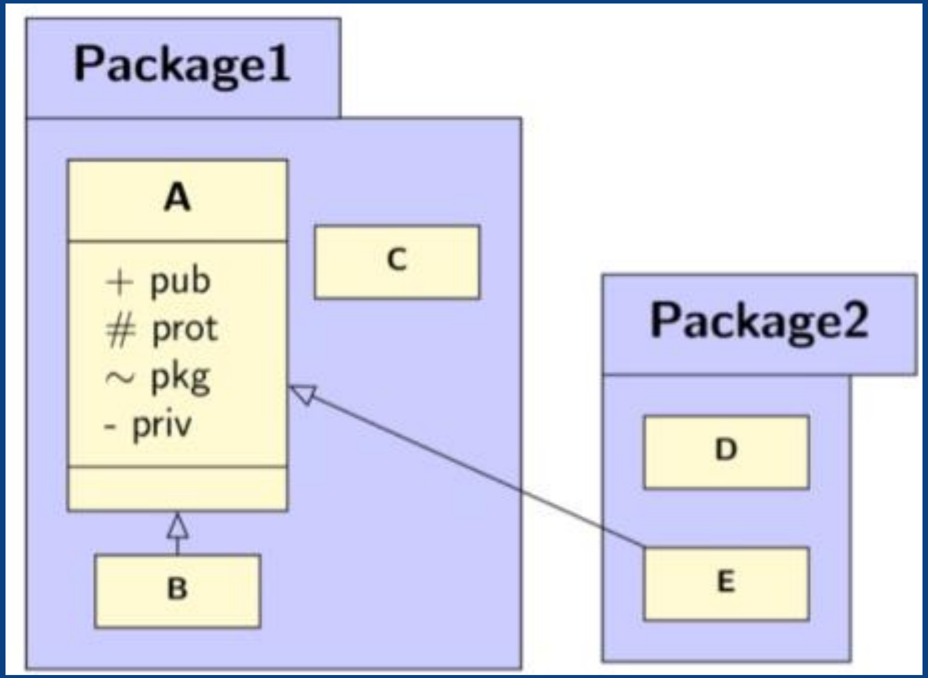
```
a1.sleep();
a1.bark();
a2.sleep();
a2.sound();
d1.bark();
d2.eat();
```



Visibility

```
private < package < protected < public
                   (package + subclass)
```

```
A sees pub, prot, pkg, priv
B sees pub, prot, pkg
C sees pub, prot, pkg
D sees pub
E sees pub, prot
```



THE UNIVERSITY OF MELBOURNE

Exercises