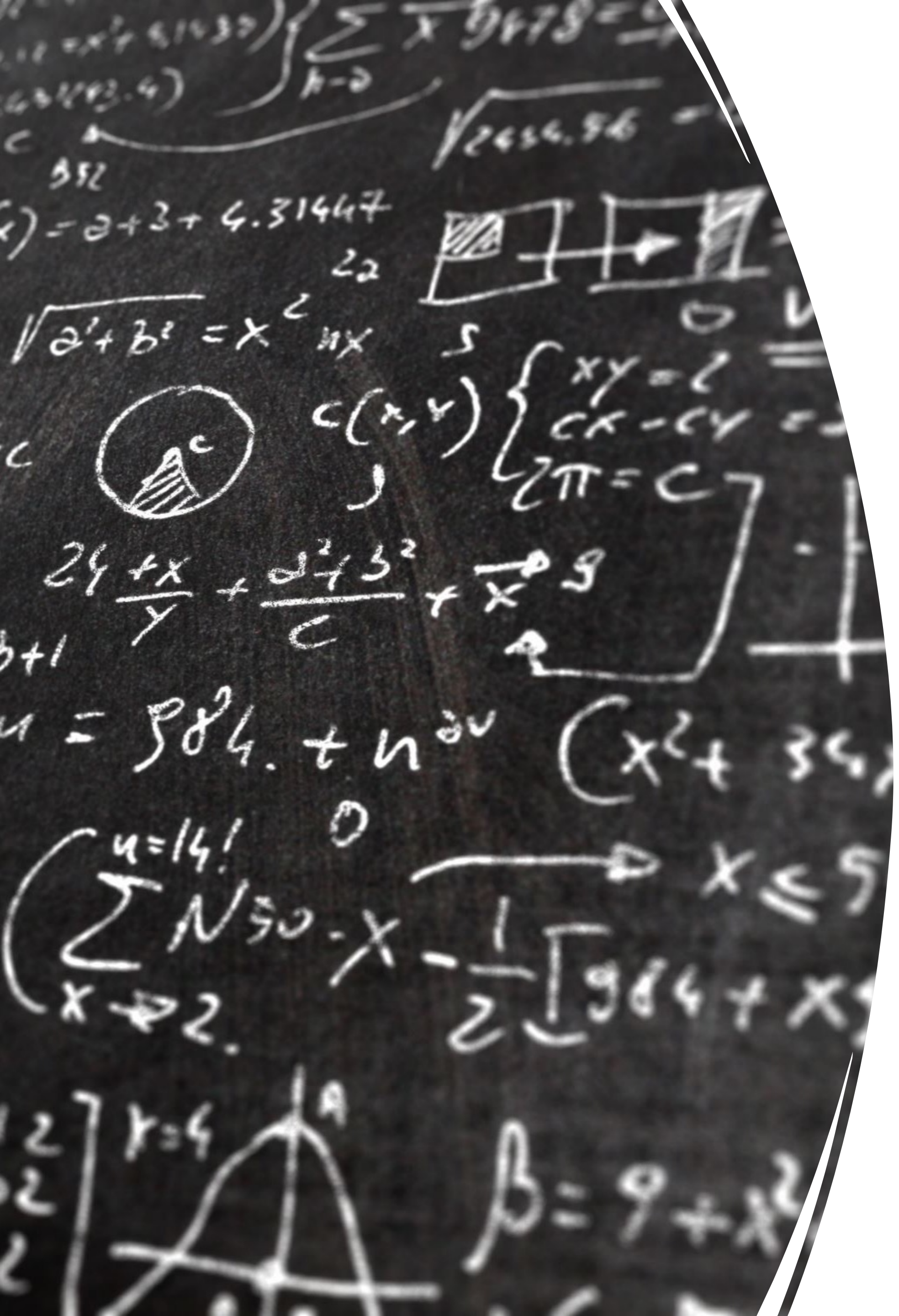


CS 113 – Computer Science I

Lecture 17 – Designing Classes II & Inheritance

Adam Poliak

03/21/2023

Announcements

- HW06
  - Due tonight 03/21
  - No autograder
- Midsemester feedback
- Scheduling announcements:
  - No class: 04/06 (Thursday)
  - Remote class: 04/11 (Tuesday) Midterm 2 review
  - 04/13 - Midterm 2

Midterm

- Hard exam

- Overall class did well:
    - Median 77%
    - Mean 68%

- When grading we'll lower the maximum
    - Instead of being out of 75, the denominator will be lower

Outline

- Review
- “this”
- Static methods
- Access modifiers
- Inheritance

Using objects: some special methods

The **constructor method** is called when you do a `new`

accesors (aka getters)
return the values of instance variables

mutators (aka setters)
set the values of instance variables

toString()
returns a string representation of an object

Defining classes

By defining our own classes, we can create our own data types

A class definition contains

- the data contained by the new type (**instance variables**)

- the operations supported by the new type (**instance methods**)

Example: Defining a class `BankAccount`

What data should it have?

- A name
- Amount of dollars

What operations should it support?

- deposit
- withdraw

this

`this` is a special keyword that refers to the object inside an instance method

Analogy:

Visualizing programs with objects

```
class BankAccount {
  public String name = ““;
  public double dollars = 0.0;

  public BankAccount() {
    this.name = “”;
    this.dollars = 0.;
  }

  public BankAccount(String clientName, double money) {
    this.name = clientName;
    this.dollars = money;
  }

 public void deposit(double money) {
     this.dollars = this.dollars + money
}
```

```
  public static void main(String[] args) {
    BankAccount acc = new BankAccount(“Kim”, 0);
    acc.depost(541);

    acc.withdraw(10);
  }
}
```

Draw a stack diagram

Draw a stack diagram

Function Stack:

Created objects

Example: Deposting using a static method

- Make a new static function called “deposit” that takes in an account and the amount to deposit and adds the amount to the account
- Should this new method return void or a value?

Exercise: Objects and Arrays

Arrays can store objects just like any other type (such as ints, Strings, etc.)

Write a program that asks the user for a number of accounts and their names and them stores the bank accounts in an array.

Exercise: Draw a stack diagram for the previous program

Access modifiers

Specify the access-level of instance variables/methods

- `public`
  - code outside of the class can access the variable/method
- `private`
  - code outside of the class cannot access the variable/method
- `protected`
  - Allow subclasses to accesses data in parent class

Default in java is `public`

Access modifiers

Default in java is `public`

In this class, make instance data private

Designing Classes

What properties does a bird have and what can it do?

- Size, color, feathers, fly

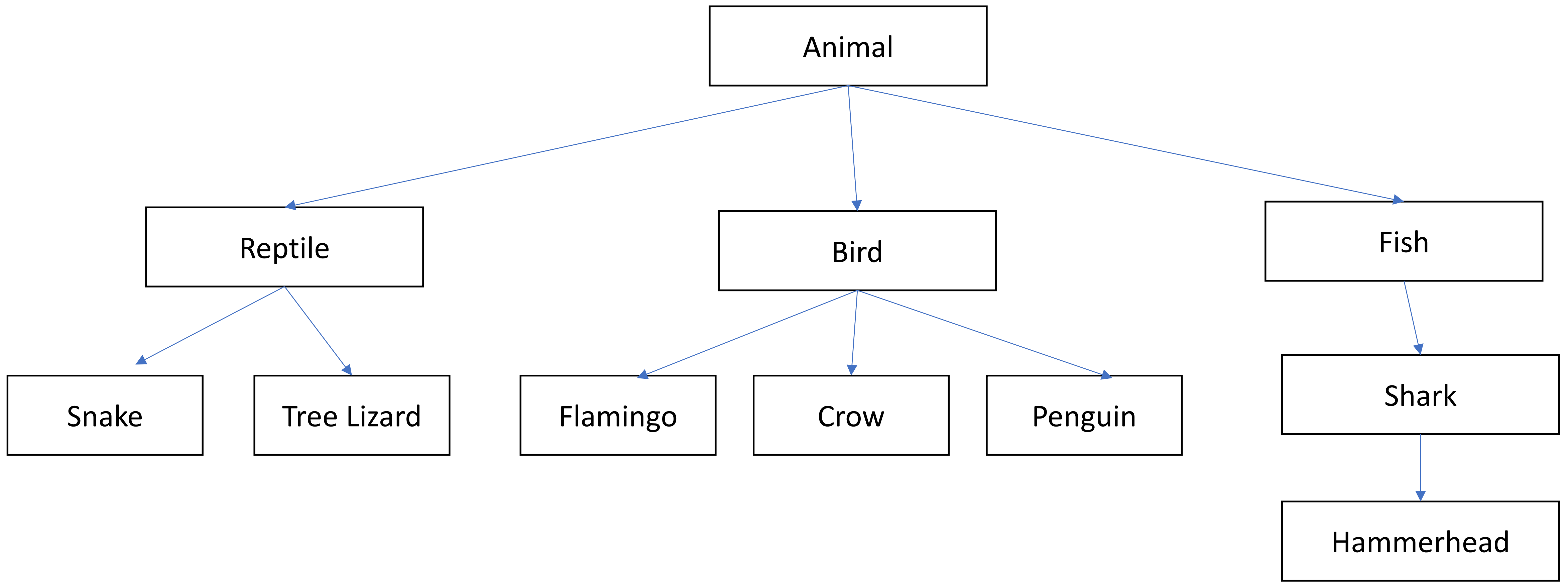
What properties does a lion have and what can it do?

- Size, color, hair, runs

What properties does a kangaroo have and what can it do?

- Size, color, arms, jumps

Inheritance: feature for organizing classes into hierarchies

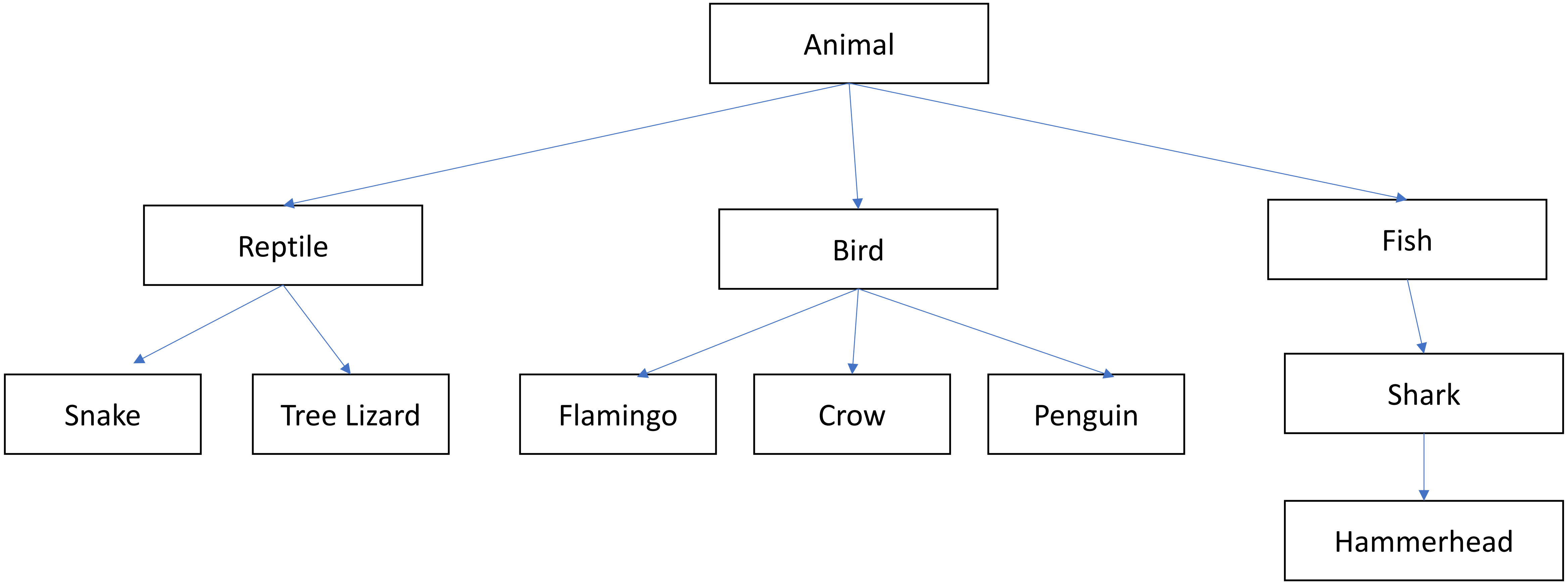


Class inheritance

Classes can be arranged hierarchically where,

    a child class ”inherits” from a parent class

Inheritance: feature for organizing classes into hierarchies



Inheritance: subclasses refine behavior/state

Subclasses can override methods from parent class

Exercise

1. Implement getter functions for instance variables inside Animal

2. In Zoo.java, call the getters and output the values to console

Polymorphism

Program can treat all objects that extend a base class the same

Java automatically calls the specific methods for each subclass

Polymorphism: Demo

```
public class Zoo {
   public static void main(String[] args) {
      Animal animal1 = new Animal();
      animal1.locomote();

      Animal animal2 = new Reptile();
      animal2.locomote();
   }
}
```

```
public class Animal {
   public Animal() {
   }
   public void locomote() {
      System.out.println("I am moving!");
   }
}
```

```
public class Reptile extends Animal {
   public Reptile() {
   }
   public void locomote() {
      System.out.println("I am walking!");
   }
}
```

Exercise: What is the output of this program?

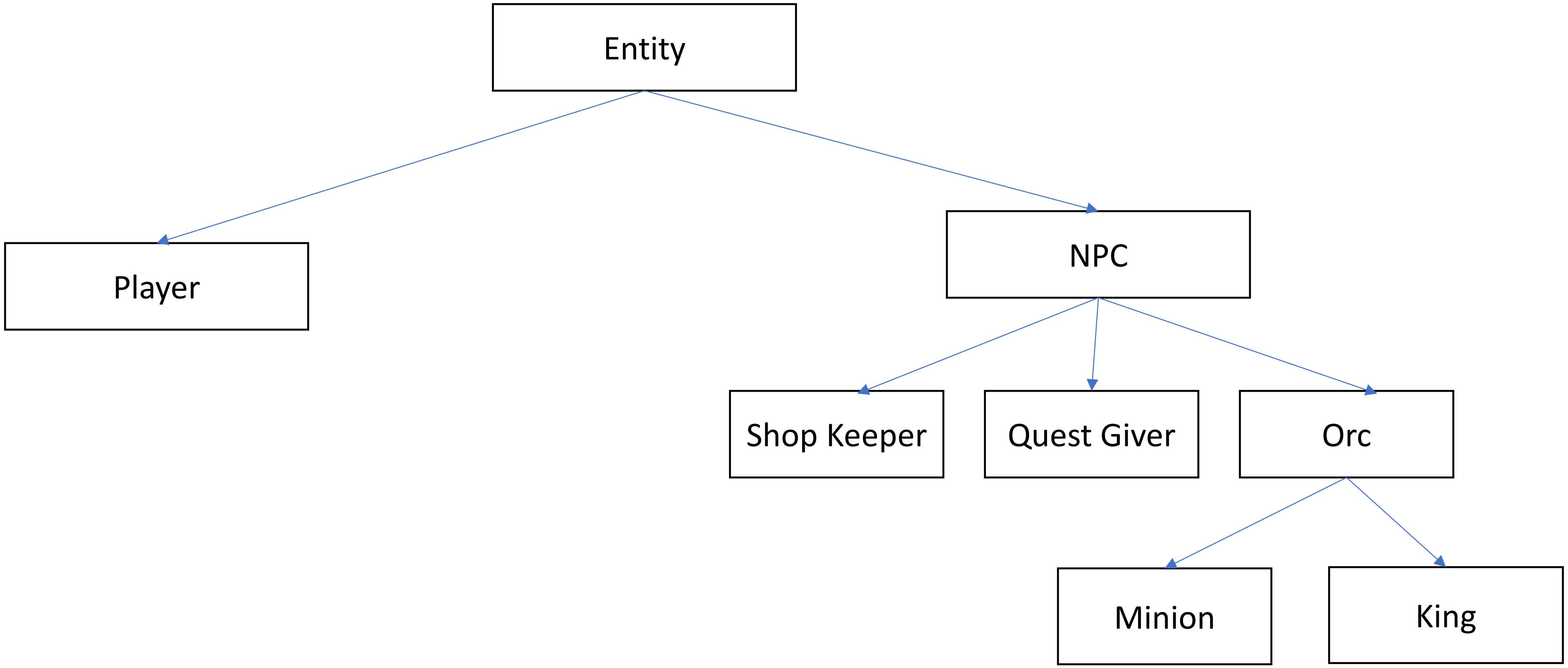
```
public class Zoo {
    public static void main(String[] args) {
        Animal animal1 = new Animal();
        animal1.locomote();

        Animal animal2 = new Fish();
        animal2.locomote();
    }
}
```

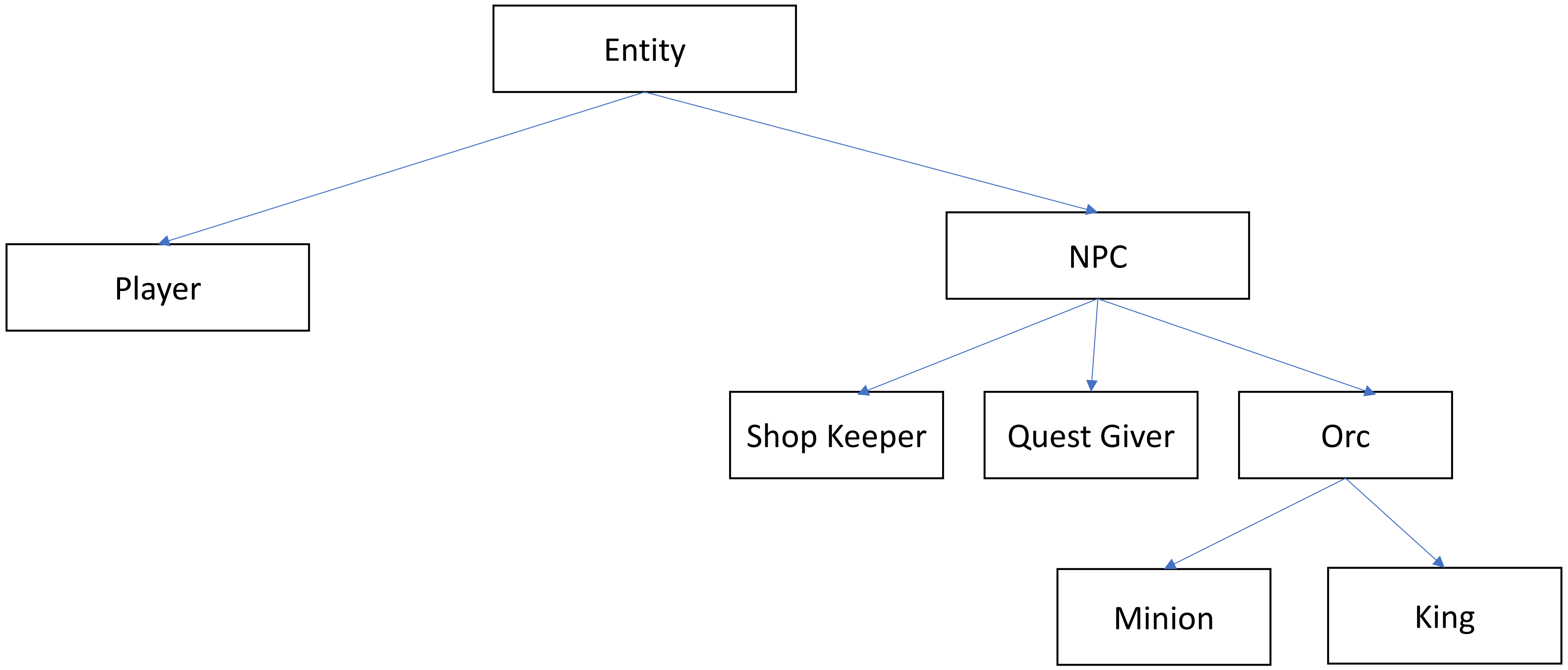
```
public class Animal {
    public Animal() {
    }
    public void locomote() {
        System.out.println("I am moving!");
    }
}
```

```
public class Fish extends Animal {
  public Fish() {
  }
  public void locomote() {
    System.out.println("I am swimming!");
  }
}
```

Question: How would we implement Minion?



Inheritance



Exercise: Implement a Bird animal