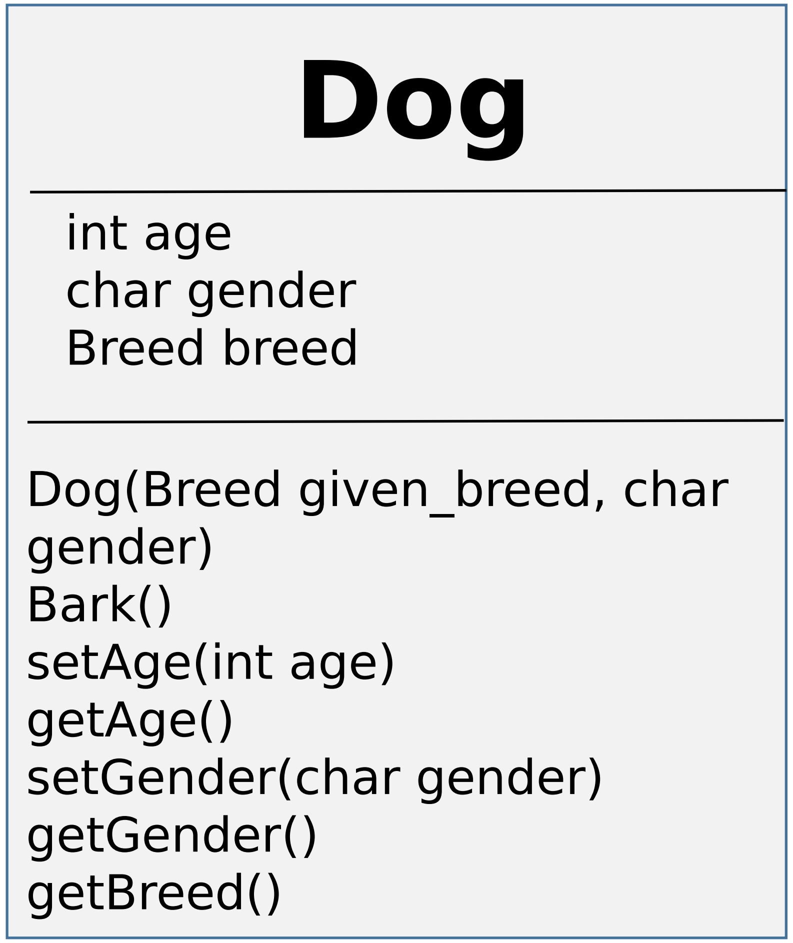
# Classes

A class is a template or blueprint for creating objects

A class contains:
  attributes (data)
  methods (behaviours/ its sub-routines)

Classes represented as a diagram:

- Name
  Attributes
  Methods
  *Constructor: – procedure to ‘build’ an object when created*
  *Getters: – functions that access and get an attribute’s value*
  *Setters: – procedures that change an attribute’s value*



**Dog**

```
int age
char gender
Breed breed
```

```
Dog(Breed given_breed, char gender)
Bark()
setAge(int age)
getAge()
setGender(char gender)
getGender()
getBreed()
```

## Constructors

Every time we create an instance of a class – an object – we need to build it

This means we set any attribute values it has or settings it needs when it's created

The constructor procedure tends to be named the same as the class

e.g. This calling code constructs a Dog of the breed collie of the male gender, when the program is run. Then it sets the age of the dog to 4 and gets the gender.

```
// Constructor in pseudocode

public procedure
Dog(given_breed, given_gender)
{
    breed = given_breed;
    age = 0;
    gender = given_gender;
}

// Calling code

Dog lassie = new Dog(collie, 'M');

lassie.setAge(4);

char gender = lassie.getGender();
```

# Encapsulation

As a general rule:

Attributes are declared private

Methods are declared public

This is so that other classes may use methods belonging to another class but may not see their attributes.

This is called ‘data hiding’ or encapsulation.

## Abstraction

Abstraction by generalisation is a grouping by common characteristics.

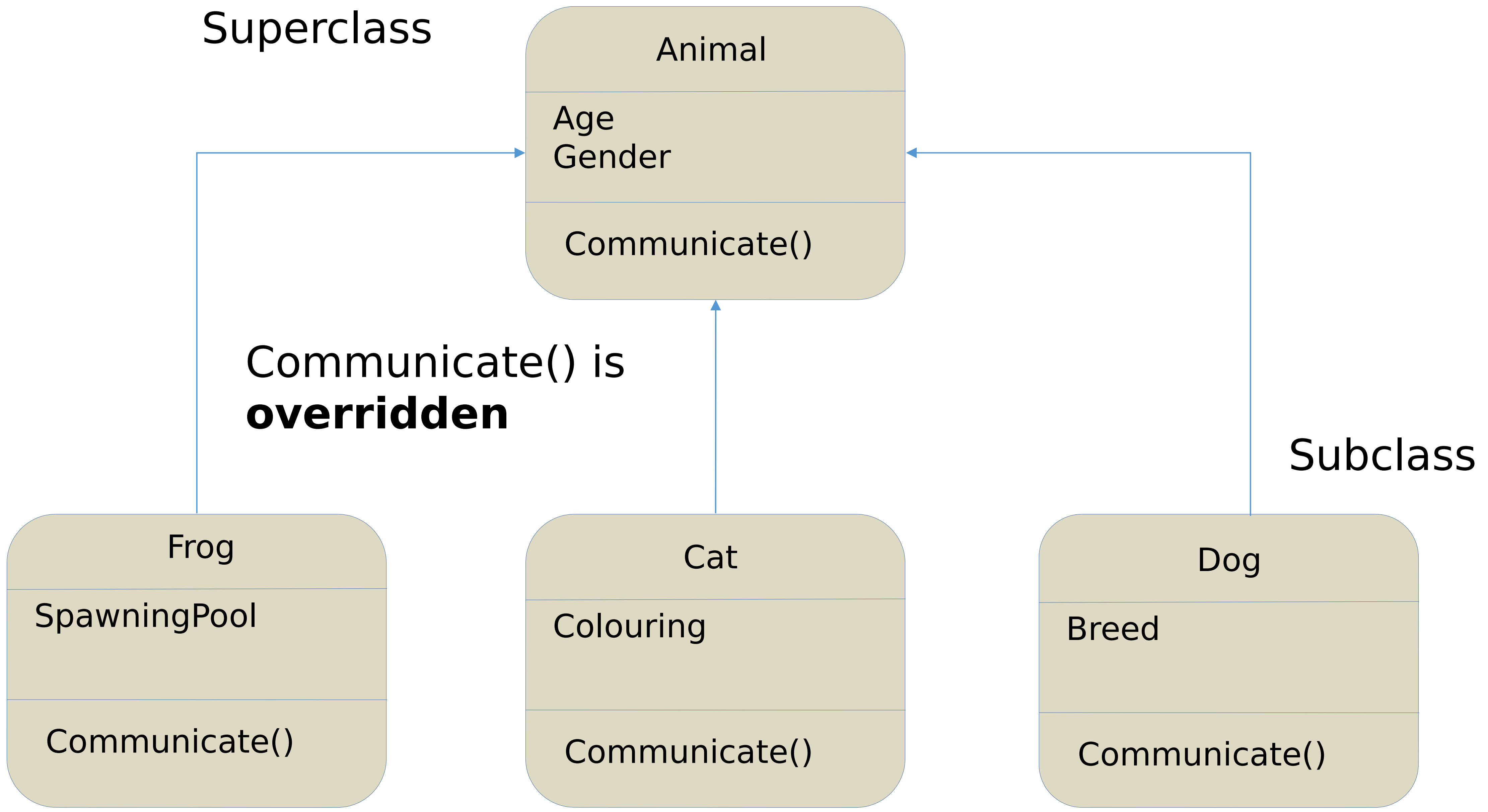
Once we start thinking  about classes in this way, we can start to consider hierarchical relationships of the ‘is a kind of’ type.

## A subclass may inherit the structure and behavior of its superclass

## Inheritance

Sub-classes of a parent (super) class can have their own attributes and methods but also use those of the parent.

Sub-classes are also known as derived classes.


Superclass
Animal
Age
Gender
Communicate()
Communicate() is **overridden**
Subclass
Frog
SpawningPool
Communicate()
Cat
Colouring
Communicate()
Dog
Breed
Communicate()

## Polymorphism

Polymorphism refers to the ability of a method to exhibit different behaviours depending on the object on which the method is invoked.

A method in a super-class can be overridden in a sub-class, so that it has its implementation changed to better suit the sub-class.

The significance of polymorphism is that the object itself knows which implementation of a method to use.

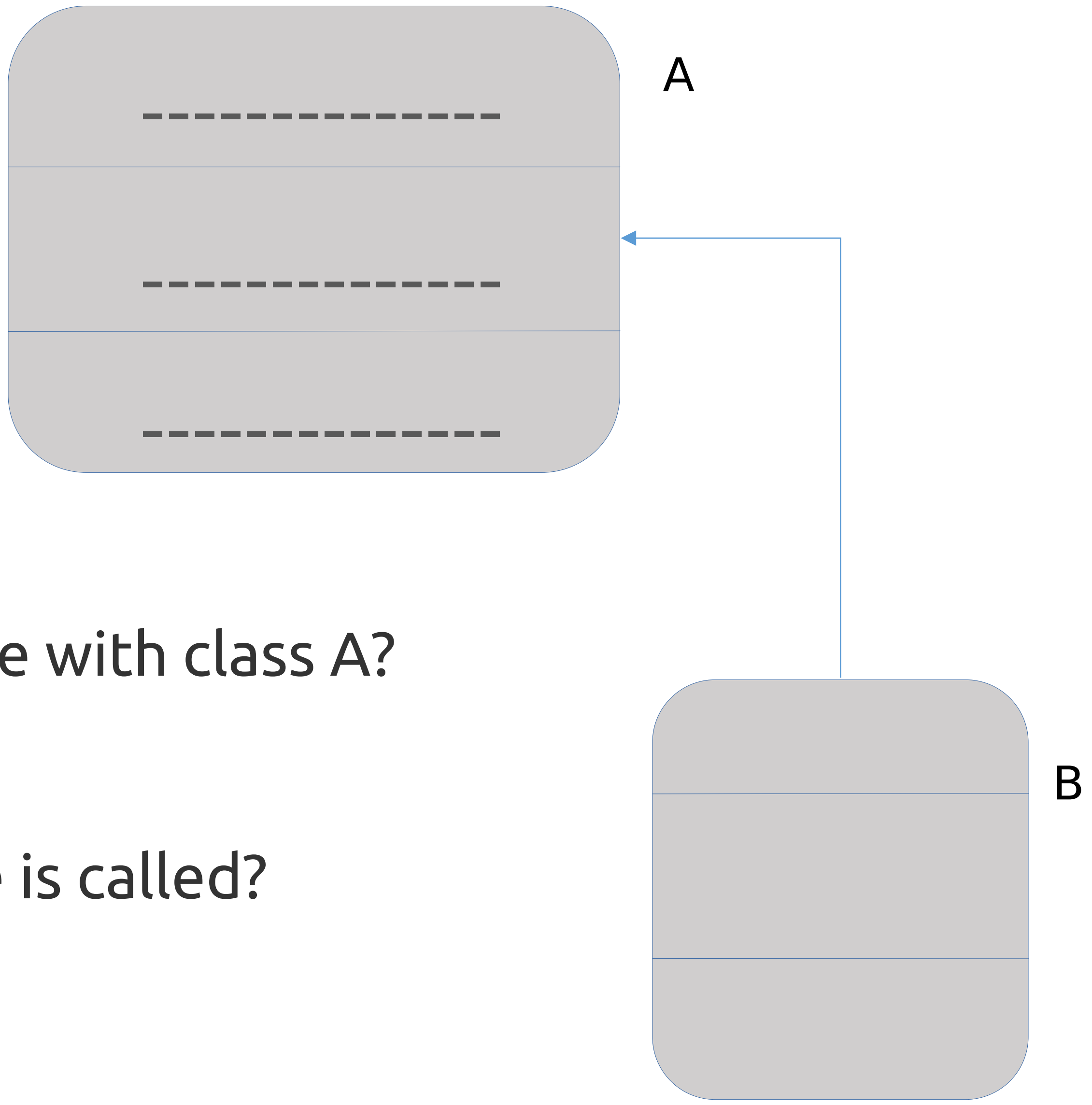
1. Fill-in the features of a class that should appear in each section of the class diagram.

2. Indicate which section normally contains private data?

3. Which procedure always exists in the bottom section?

4. What type of relationship does class B have with class A?

5. What is making an object from a class type is called?



## Animalopoly

You have been hired to create a text based board game. Your employer has requested the code is created using object oriented programming and to be maintainable to enable other coders can work on it in the future. This board game needs to run on the Windows operating system so they have requested you write it in C#.

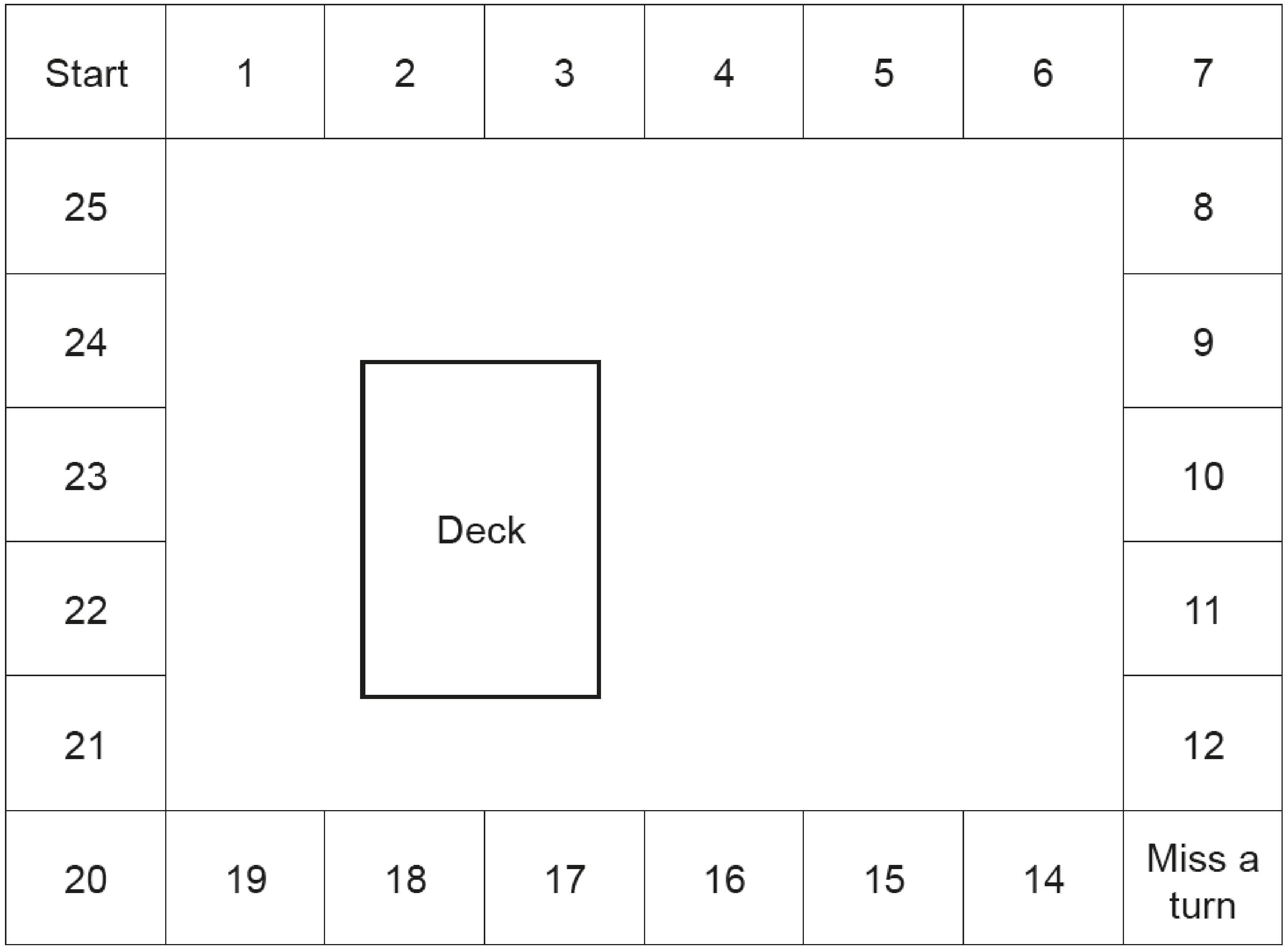
You are creating an animal themed version of a famous board game. In this game players move around a board buying animals, you then charge other players to visit that animal. The last player that still has money wins.

# Animalopoly

Your players move around a board with 26 spaces. They move around by rolling two dice which are added together to determine how far they move.

If a player rolls a double, they are given the chance to draw a card from the deck. The card will contain a scenario (e.g. “You have won the lottery” / “You owe debt”) and an amount of money they player is given or must pay.

When a player passes start they are given £500. If they land on start they are given £1000. Landing on “miss a turn” means they miss their next turn.


Start
1
2
3
4
5
6
7
8
9
10
11
12
Miss a turn
14
15
16
17
18
19
20
21
22
23
24
25
Deck

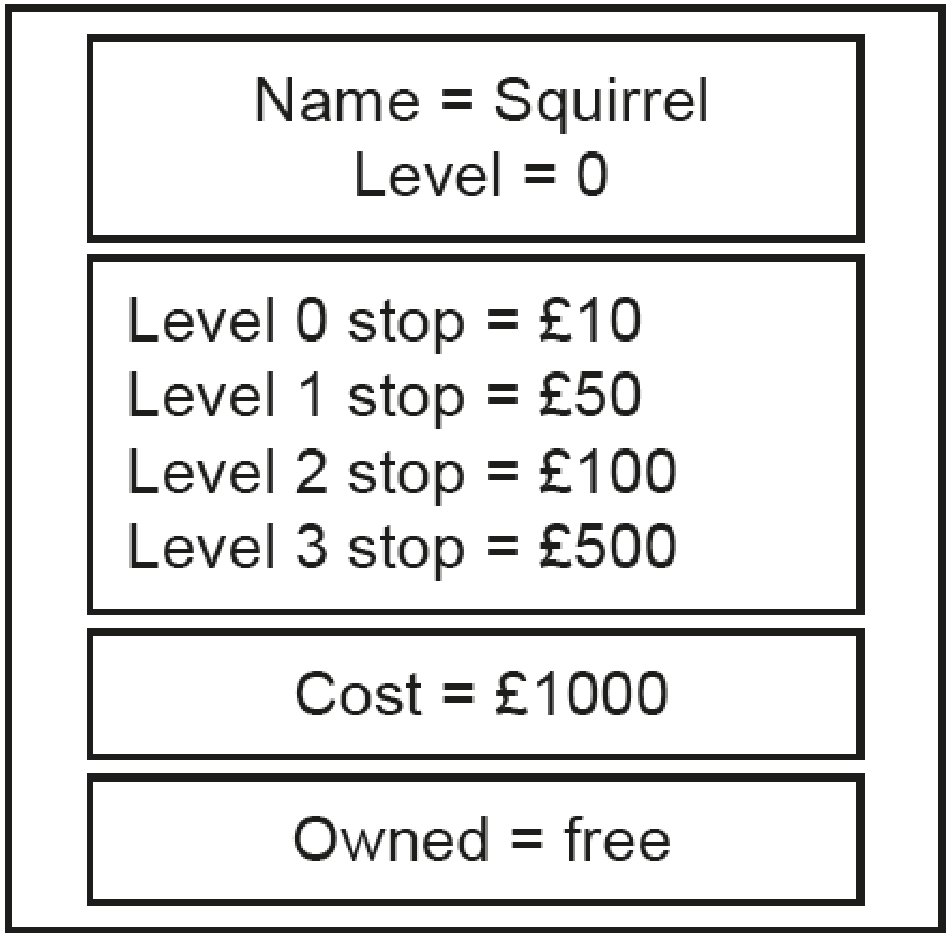
# Animalopoly

On each space they land there is information about the animal that lives in that space. The animal has a name/species, a level, a cost to stop/visit, a cost to buy and an owner.

If the animal has no owner the player is offered the chance to buy it at its set cost.

If the animal is already owned by someone else, you must pay the stop cost. The level determines the cost to visit and can be upgraded by the owner.

An example of a squirrel animal might be:



Name = Squirrel
Level = 0

Level 0 stop = £10
Level 1 stop = £50
Level 2 stop = £100
Level 3 stop = £500

Cost = £1000

Owned = free

## Animalopoly – deliverables

# Animalopoly
# What classes do you need?

Dice

Have two dice that the player can roll

Tell the player how many spaces to move by

Confirm when two of the same dice have been rolled

# Questions to consider

How do we keep track of our Players?

How do we keep track of our Animals?

How do we keep track of the Cards?

What is the relationship between the Board and the Players?

What is the relationship between the Board and Animals?