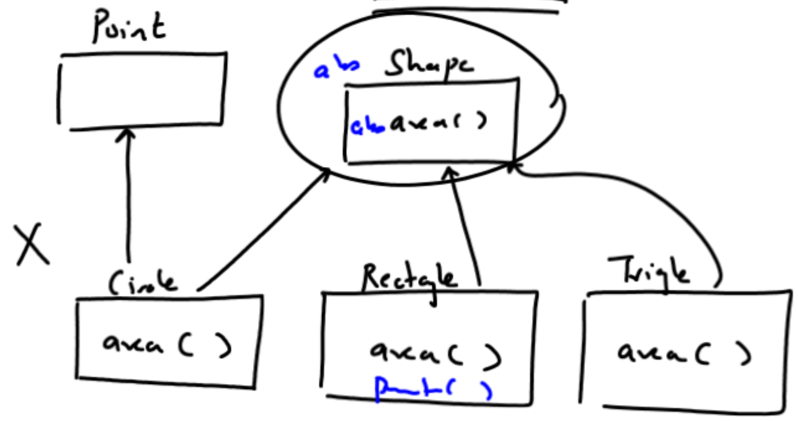
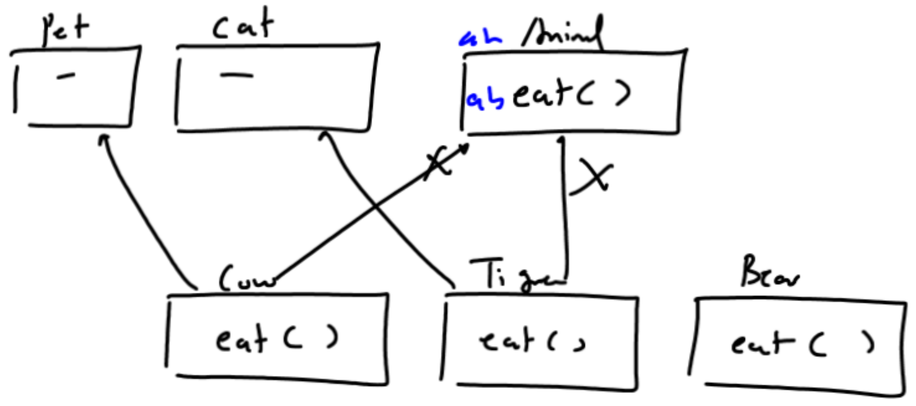
INTERFACE



```
interface Shape
{
  =
}
```



```
interface I
{
        int a=10;

}
```

```
interface I
{
public static final int a = 10;

}
```

automatically added
public static final

Syntax Of Declaring An Interface
```
=========================
interface <name_of _the_interface>
{

    <data_type> <var_name>=value;
     ........
    <ret_type> <method_name>(<arg>);
   .......
}
```

public abstract
automatically
added

Syntax Of Inheriting An Interface

```
class <class_name> implements <interface_name>
{
   // Body of all abstract methods of interface
}

class <class_name> implements <interface_name>,<interface_name>
{
   // Body of all abstract methods of BOTH the interfaces
}
class <class_name> extends <class_name> implements <interface_name>
{
   // Body of all abstract methods of interface
}
```

Some Examples
=============

```
interface Shape
{
     double area(); // will automatically be made as  public abstract
}

interface Animal
{
     void eat();// will automatically be made as  public abstract
     void sleep();// will automatically be made as  public abstract
     int NUM_OF_LEGS=4; // will automatically be made as  public static final
}
```