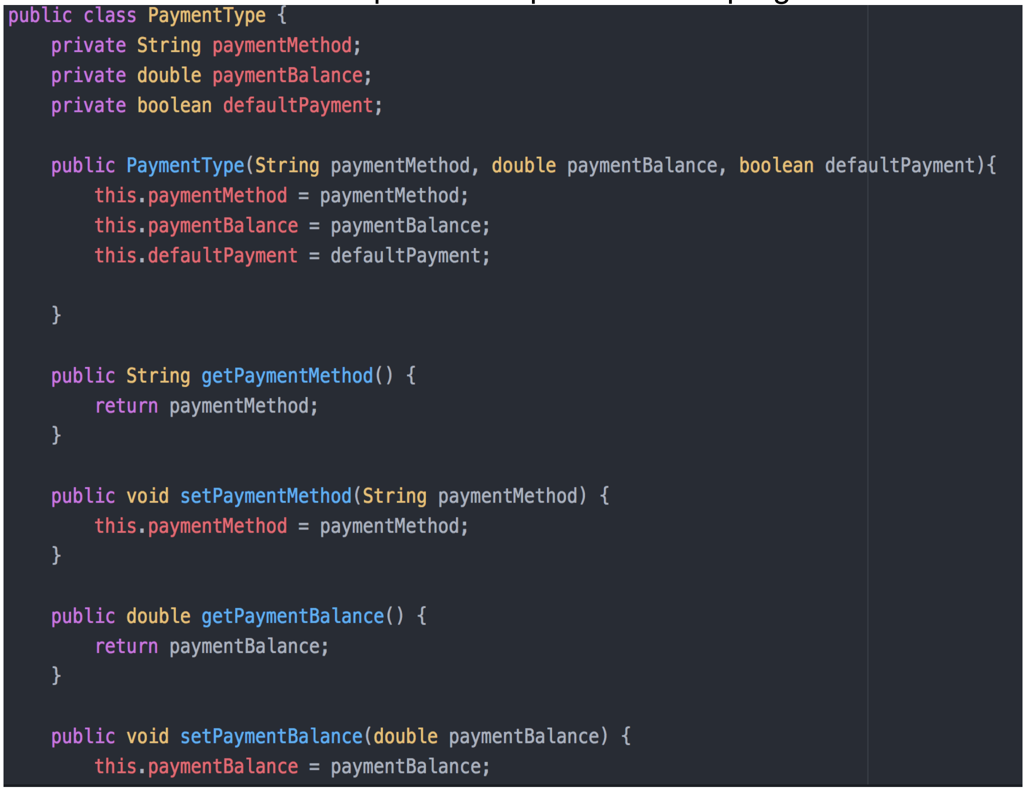
Implementation and Testing Unit

Kishan Bachoo
E19

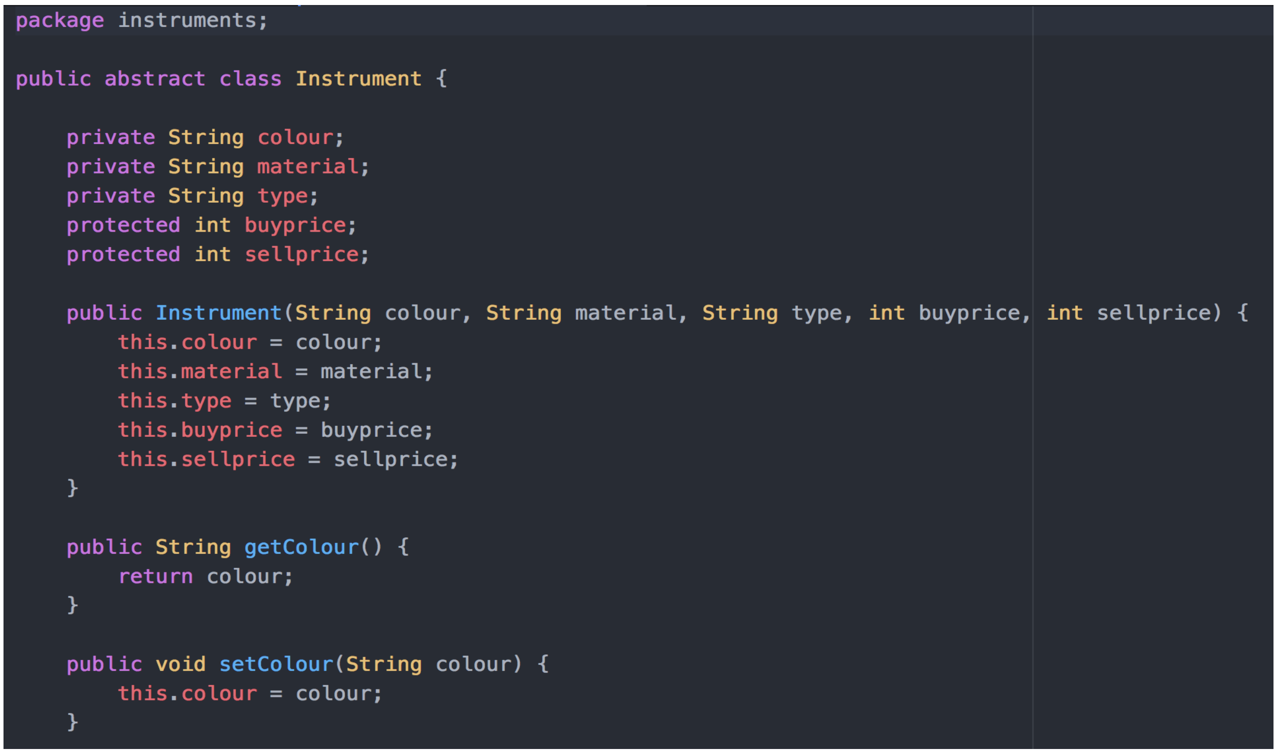
I.T 1 Take a screenshot of an example of encapsulation in a program



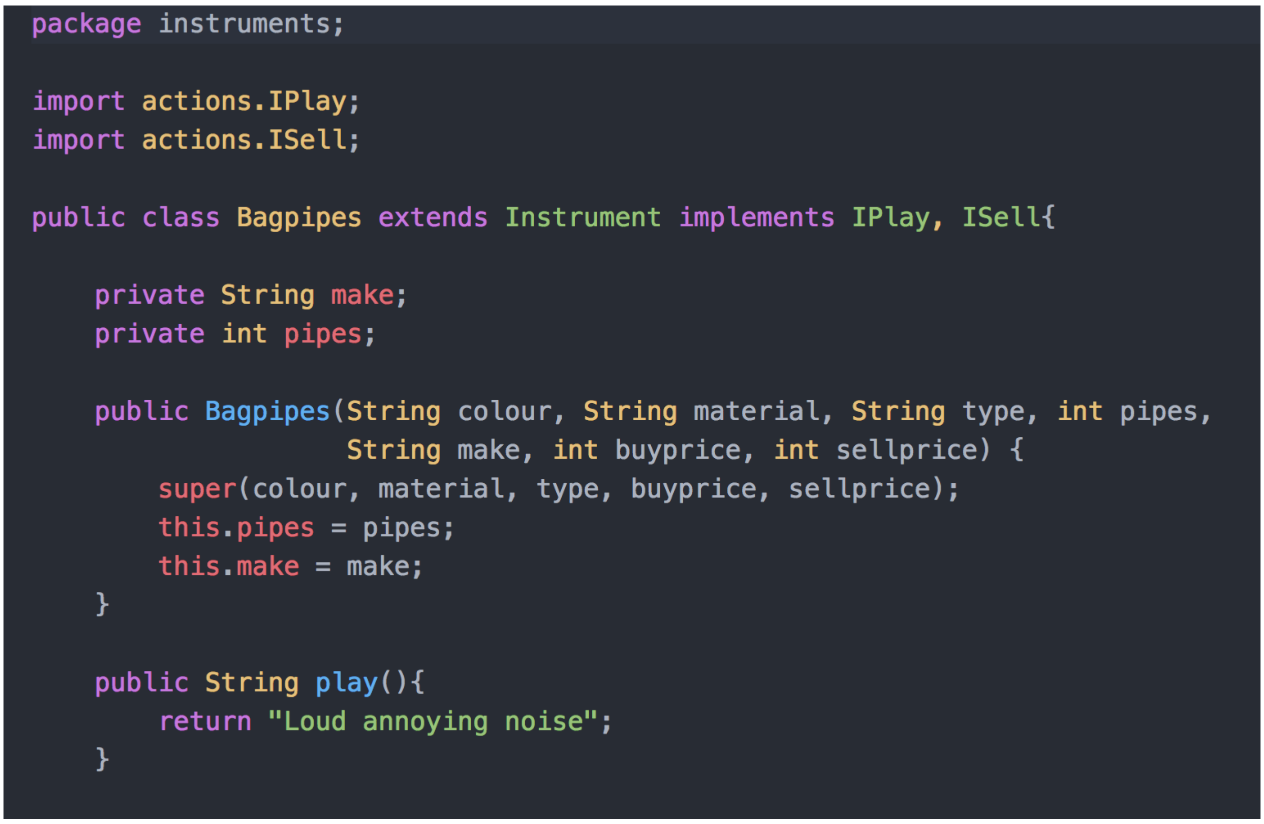
```
public class PaymentType {
    private String paymentMethod;
    private double paymentBalance;
    private boolean defaultPayment;

    public PaymentType(String paymentMethod, double paymentBalance, boolean defaultPayment){
        this.paymentMethod = paymentMethod;
        this.paymentBalance = paymentBalance;
        this.defaultPayment = defaultPayment;

    }

    public String getPaymentMethod() {
        return paymentMethod;
    }

    public void setPaymentMethod(String paymentMethod) {
        this.paymentMethod = paymentMethod;
    }

    public double getPaymentBalance() {
        return paymentBalance;
    }

    public void setPaymentBalance(double paymentBalance) {
        this.paymentBalance = paymentBalance;
```

I.T 2 Take a screenshot of the use of Inheritance in a program. Take screenshots of:

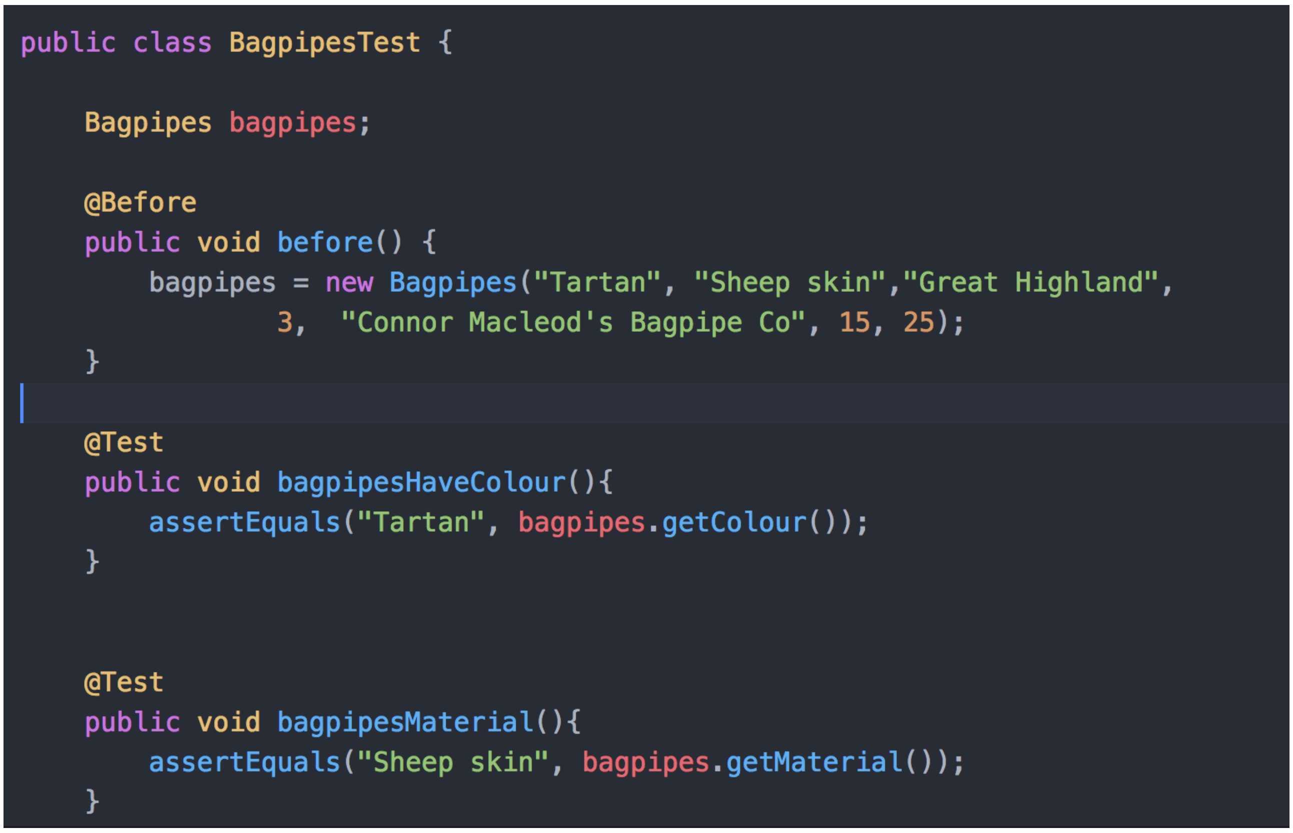
- A Class


```java
package instruments;

public abstract class Instrument {

    private String colour;
    private String material;
    private String type;
    protected int buyprice;
    protected int sellprice;

    public Instrument(String colour, String material, String type, int buyprice, int sellprice) {
        this.colour = colour;
        this.material = material;
        this.type = type;
        this.buyprice = buyprice;
        this.sellprice = sellprice;
    }

    public String getColour() {
        return colour;
    }

    public void setColour(String colour) {
        this.colour = colour;
    }
```

- A Class that inherits from the previous class

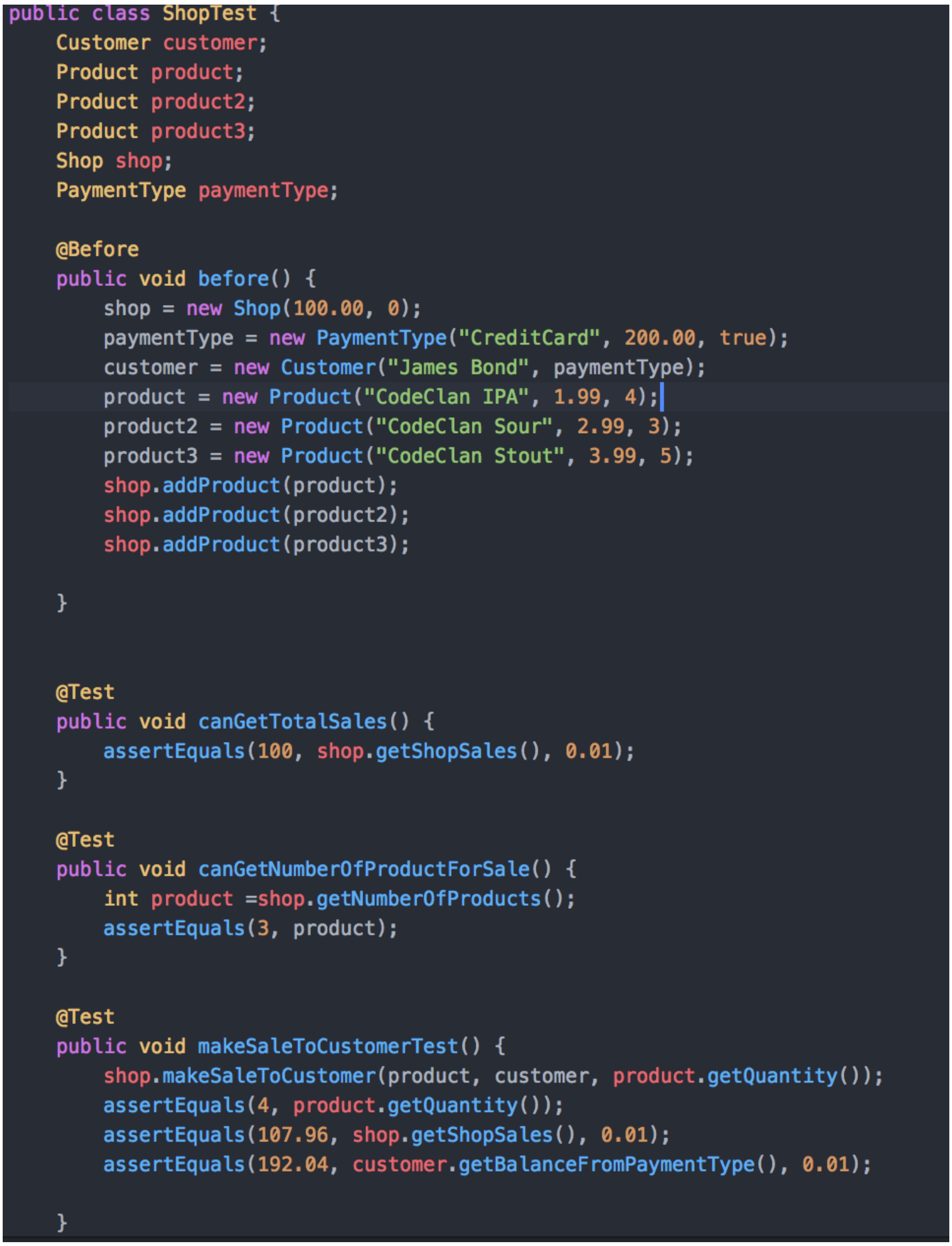

```java
package instruments;

import actions.IPlay;
import actions.ISell;

public class Bagpipes extends Instrument implements IPlay, ISell{

    private String make;
    private int pipes;

    public Bagpipes(String colour, String material, String type, int pipes,
                    String make, int buyprice, int sellprice) {
        super(colour, material, type, buyprice, sellprice);
        this.pipes = pipes;
        this.make = make;
    }

    public String play(){
        return "Loud annoying noise";
    }
```

- An object in the inherited class



```
public class BagpipesTest {

    Bagpipes bagpipes;

    @Before
    public void before() {
        bagpipes = new Bagpipes("Tartan", "Sheep skin","Great Highland",
                3,  "Connor Macleod's Bagpipe Co", 15, 25);
    }

    @Test
    public void bagpipesHaveColour(){
        assertEquals("Tartan", bagpipes.getColour());
    }


    @Test
    public void bagpipesMaterial(){
        assertEquals("Sheep skin", bagpipes.getMaterial());
    }
```

- A method that uses the information inherited from another class

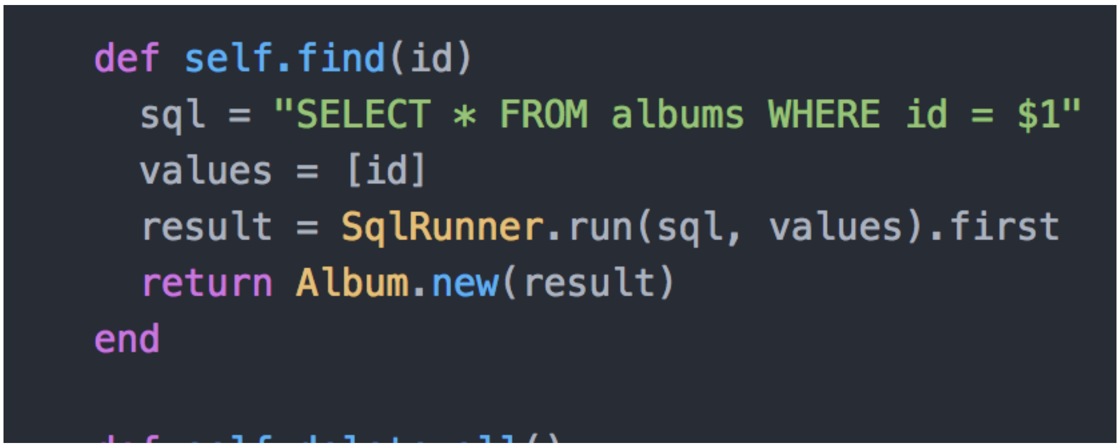


```java
public class ShopTest {
    Customer customer;
    Product product;
    Product product2;
    Product product3;
    Shop shop;
    PaymentType paymentType;

    @Before
    public void before() {
        shop = new Shop(100.00, 0);
        paymentType = new PaymentType("CreditCard", 200.00, true);
        customer = new Customer("James Bond", paymentType);
        product = new Product("CodeClan IPA", 1.99, 4);
        product2 = new Product("CodeClan Sour", 2.99, 3);
        product3 = new Product("CodeClan Stout", 3.99, 5);
        shop.addProduct(product);
        shop.addProduct(product2);
        shop.addProduct(product3);

    }


    @Test
    public void canGetTotalSales() {
        assertEquals(100, shop.getShopSales(), 0.01);
    }

    @Test
    public void canGetNumberOfProductForSale() {
        int product =shop.getNumberOfProducts();
        assertEquals(3, product);
    }

    @Test
    public void makeSaleToCustomerTest() {
        shop.makeSaleToCustomer(product, customer, product.getQuantity());
        assertEquals(4, product.getQuantity());
        assertEquals(107.96, shop.getShopSales(), 0.01);
        assertEquals(192.04, customer.getBalanceFromPaymentType(), 0.01);

    }
```

I.T 3 Demonstrate searching data in a program.

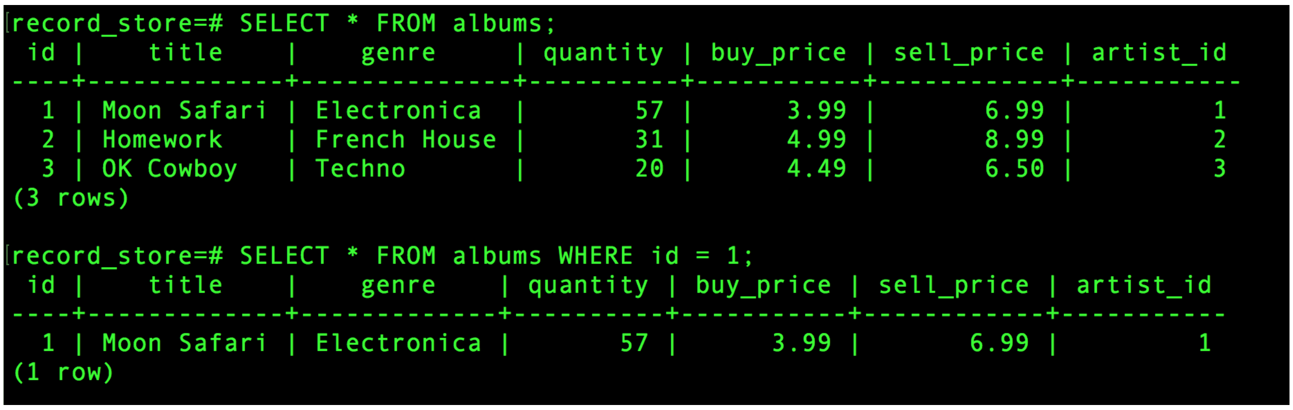
Take screenshots of:

- Function that searches data



```ruby
def self.find(id)
  sql = "SELECT * FROM albums WHERE id = $1"
  values = [id]
  result = SqlRunner.run(sql, values).first
  return Album.new(result)
end
```

- The result of the function running

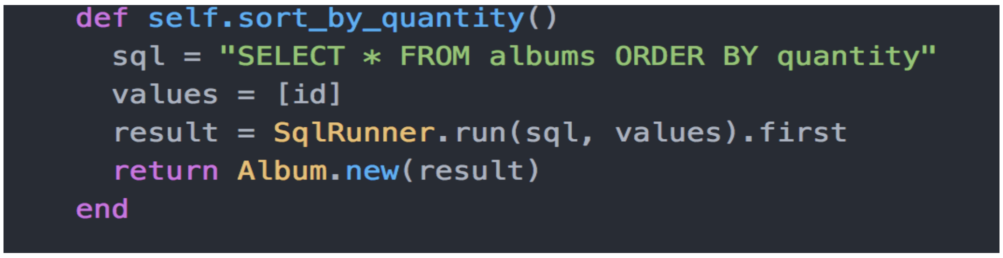


```
[record_store=# SELECT * FROM albums;
 id |    title     |    genre     | quantity | buy_price | sell_price | artist_id
----+--------------+--------------+----------+-----------+------------+-----------
  1 | Moon Safari | Electronica  |       57 |      3.99 |       6.99 |         1
  2 | Homework    | French House |       31 |      4.99 |       8.99 |         2
  3 | OK Cowboy   | Techno       |       20 |      4.49 |       6.50 |         3
(3 rows)

[record_store=# SELECT * FROM albums WHERE id = 1;
 id |    title     |    genre    | quantity | buy_price | sell_price | artist_id
----+--------------+-------------+----------+-----------+------------+-----------
  1 | Moon Safari | Electronica |       57 |      3.99 |       6.99 |         1
(1 row)
```

I.T 4 Demonstrate sorting data in a program

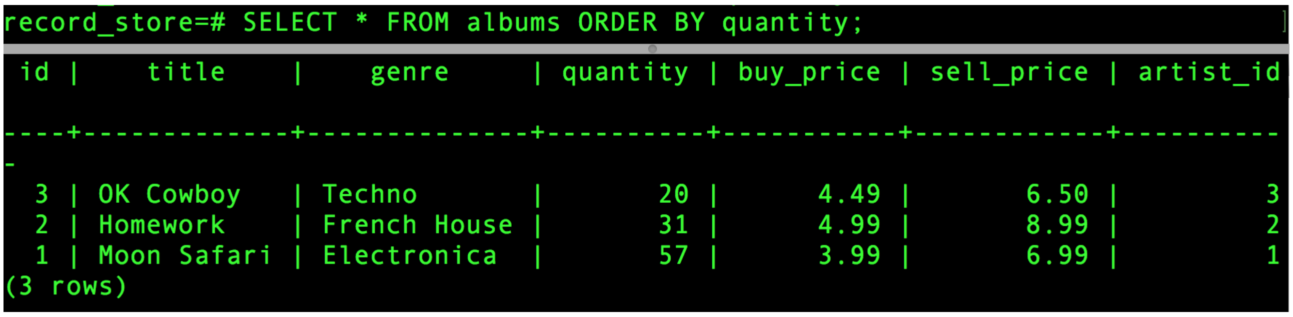
Take screenshots of:

- Function that sorts data



```ruby
def self.sort_by_quantity()
  sql = "SELECT * FROM albums ORDER BY quantity"
  values = [id]
  result = SqlRunner.run(sql, values).first
  return Album.new(result)
end
```

- The result of the function running

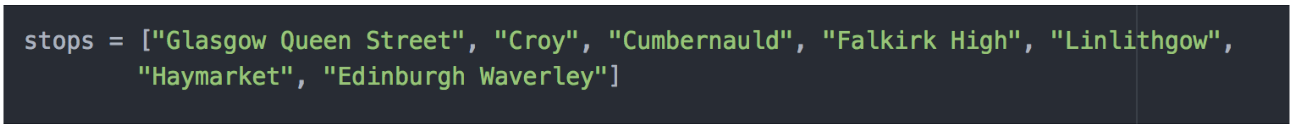


```
record_store=# SELECT * FROM albums ORDER BY quantity;
 id |    title    |     genre     | quantity | buy_price | sell_price | artist_id
----+-------------+---------------+----------+-----------+------------+----------
-
  3 | OK Cowboy   | Techno        |       20 |      4.49 |       6.50 |         3
  2 | Homework    | French House  |       31 |      4.99 |       8.99 |         2
  1 | Moon Safari | Electronica   |       57 |      3.99 |       6.99 |         1
(3 rows)
```

I.T 5 Demonstrate the use of an array in a program

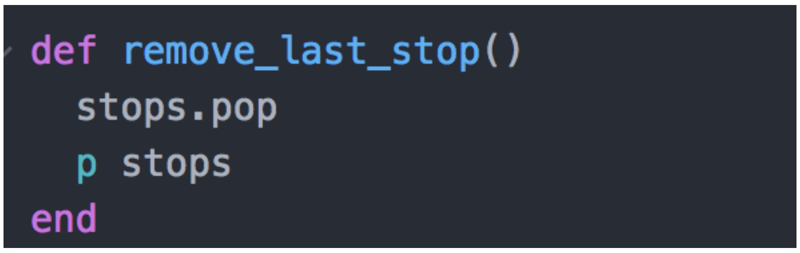
Take screenshots of:

- An array in a program



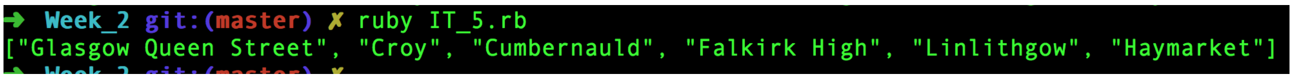
```
stops = ["Glasgow Queen Street", "Croy", "Cumbernauld", "Falkirk High", "Linlithgow",
        "Haymarket", "Edinburgh Waverley"]
```

- A function that uses the array



```
def remove_last_stop()
  stops.pop
  p stops
end
```

- The result of the function running

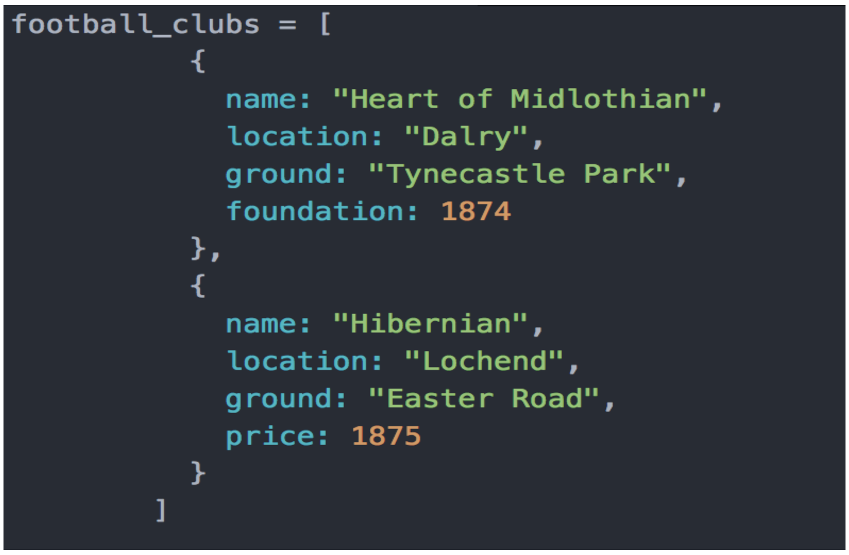


```
➜ Week_2 git:(master) ✗ ruby IT_5.rb
["Glasgow Queen Street", "Croy", "Cumbernauld", "Falkirk High", "Linlithgow", "Haymarket"]
```

I.T 6 Demonstrate the use of a hash in a program

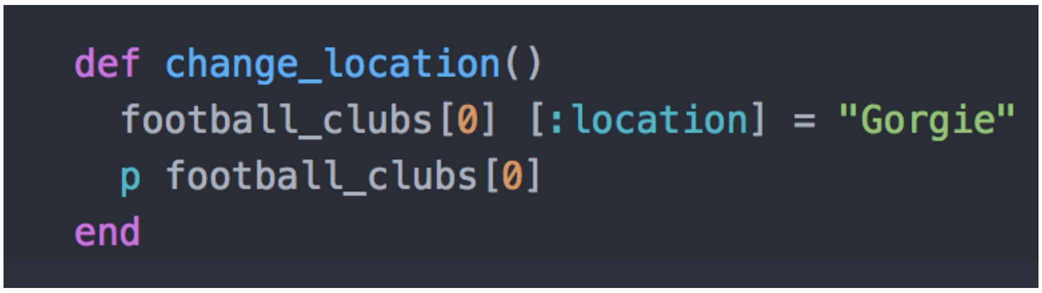
Take screenshots of:

- A hash in a program



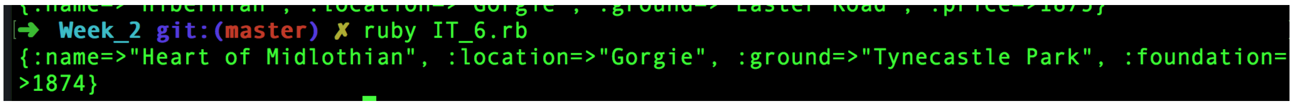
```
football_clubs = [
          {
            name: "Heart of Midlothian",
            location: "Dalry",
            ground: "Tynecastle Park",
            foundation: 1874
          },
          {
            name: "Hibernian",
            location: "Lochend",
            ground: "Easter Road",
            price: 1875
          }
        ]
```

- A function that uses the hash



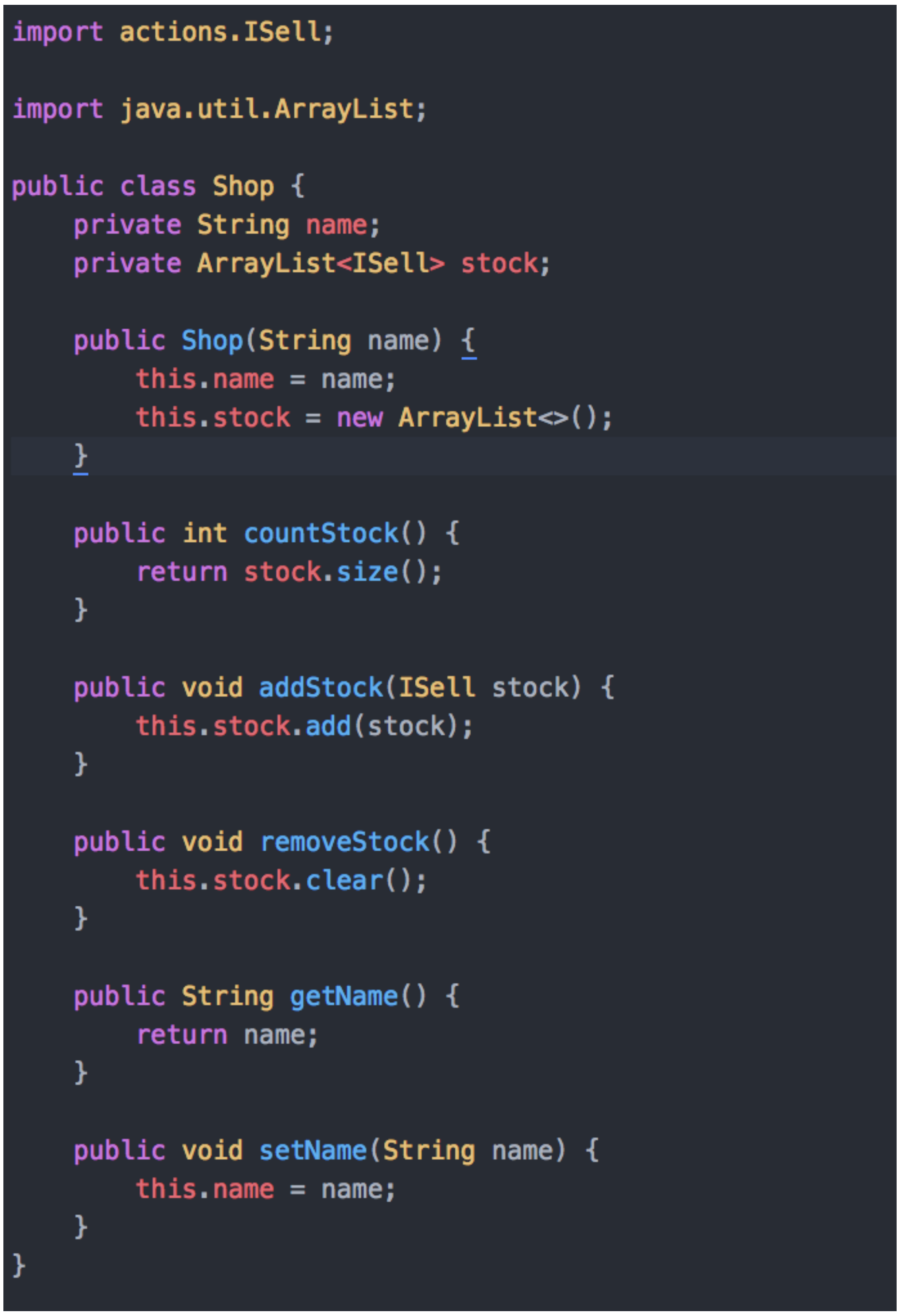
```
def change_location()
  football_clubs[0] [:location] = "Gorgie"
  p football_clubs[0]
end
```

- The result of the function running

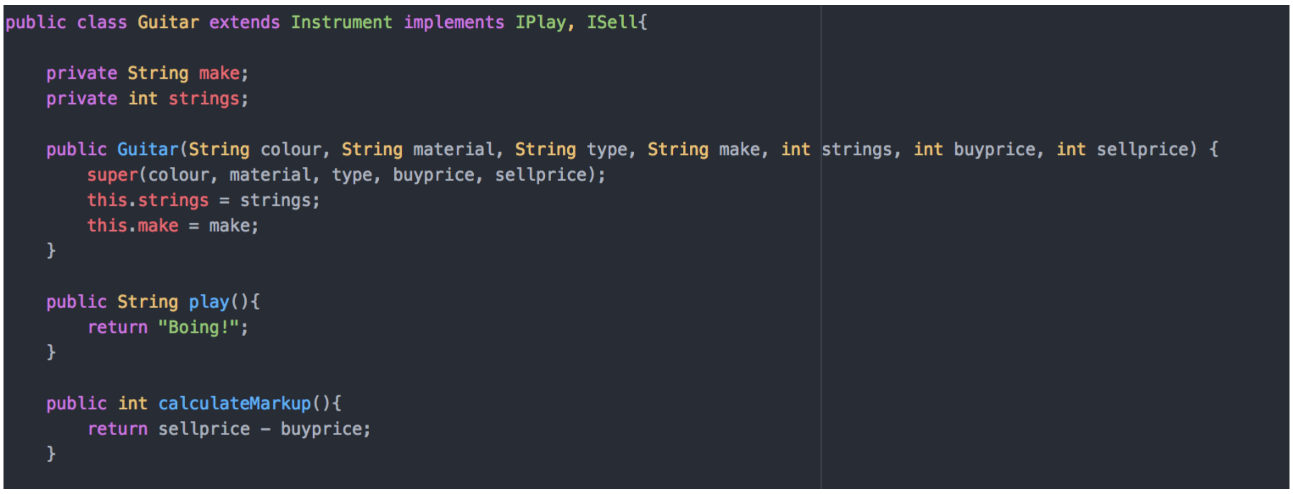


```
{:name=>"Hibernian", :location=>"Gorgie", :ground=>"Easter Road", :price=>1875}
➜ Week_2 git:(master) ✗ ruby IT_6.rb
{:name=>"Heart of Midlothian", :location=>"Gorgie", :ground=>"Tynecastle Park", :foundation=
>1874}
```

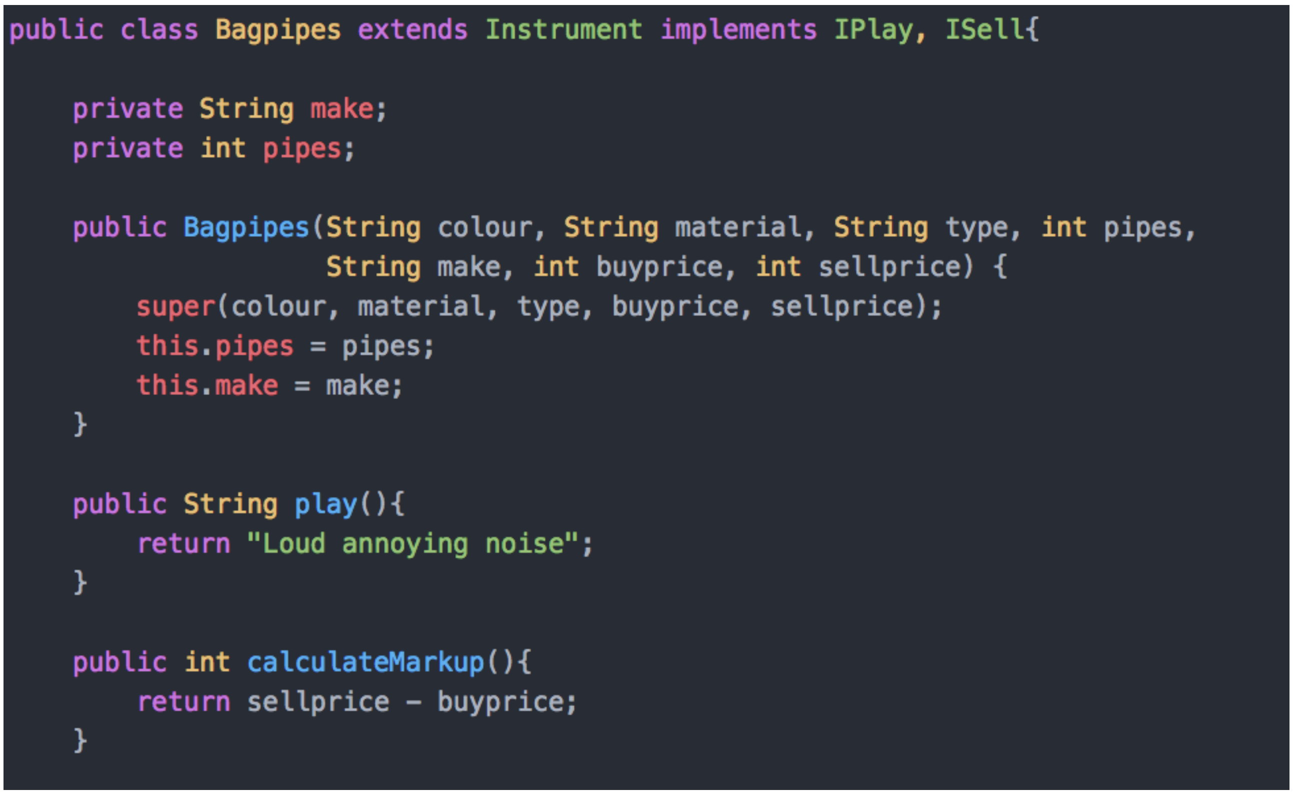
I.T 7 Demonstrate the use of Polymorphism in a program



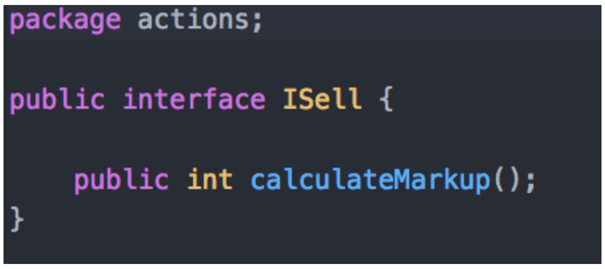
```java
import actions.ISell;

import java.util.ArrayList;

public class Shop {
    private String name;
    private ArrayList<ISell> stock;

    public Shop(String name) {
        this.name = name;
        this.stock = new ArrayList<>();
    }

    public int countStock() {
        return stock.size();
    }

    public void addStock(ISell stock) {
        this.stock.add(stock);
    }

    public void removeStock() {
        this.stock.clear();
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }
}
```



```java
public class Guitar extends Instrument implements IPlay, ISell{

    private String make;
    private int strings;

    public Guitar(String colour, String material, String type, String make, int strings, int buyprice, int sellprice) {
        super(colour, material, type, buyprice, sellprice);
        this.strings = strings;
        this.make = make;
    }

    public String play(){
        return "Boing!";
    }

    public int calculateMarkup(){
        return sellprice - buyprice;
    }
```



```java
public class Bagpipes extends Instrument implements IPlay, ISell{

    private String make;
    private int pipes;

    public Bagpipes(String colour, String material, String type, int pipes,
                    String make, int buyprice, int sellprice) {
        super(colour, material, type, buyprice, sellprice);
        this.pipes = pipes;
        this.make = make;
    }

    public String play(){
        return "Loud annoying noise";
    }

    public int calculateMarkup(){
        return sellprice - buyprice;
    }
```



```java
package actions;

public interface ISell {

    public int calculateMarkup();
}
```