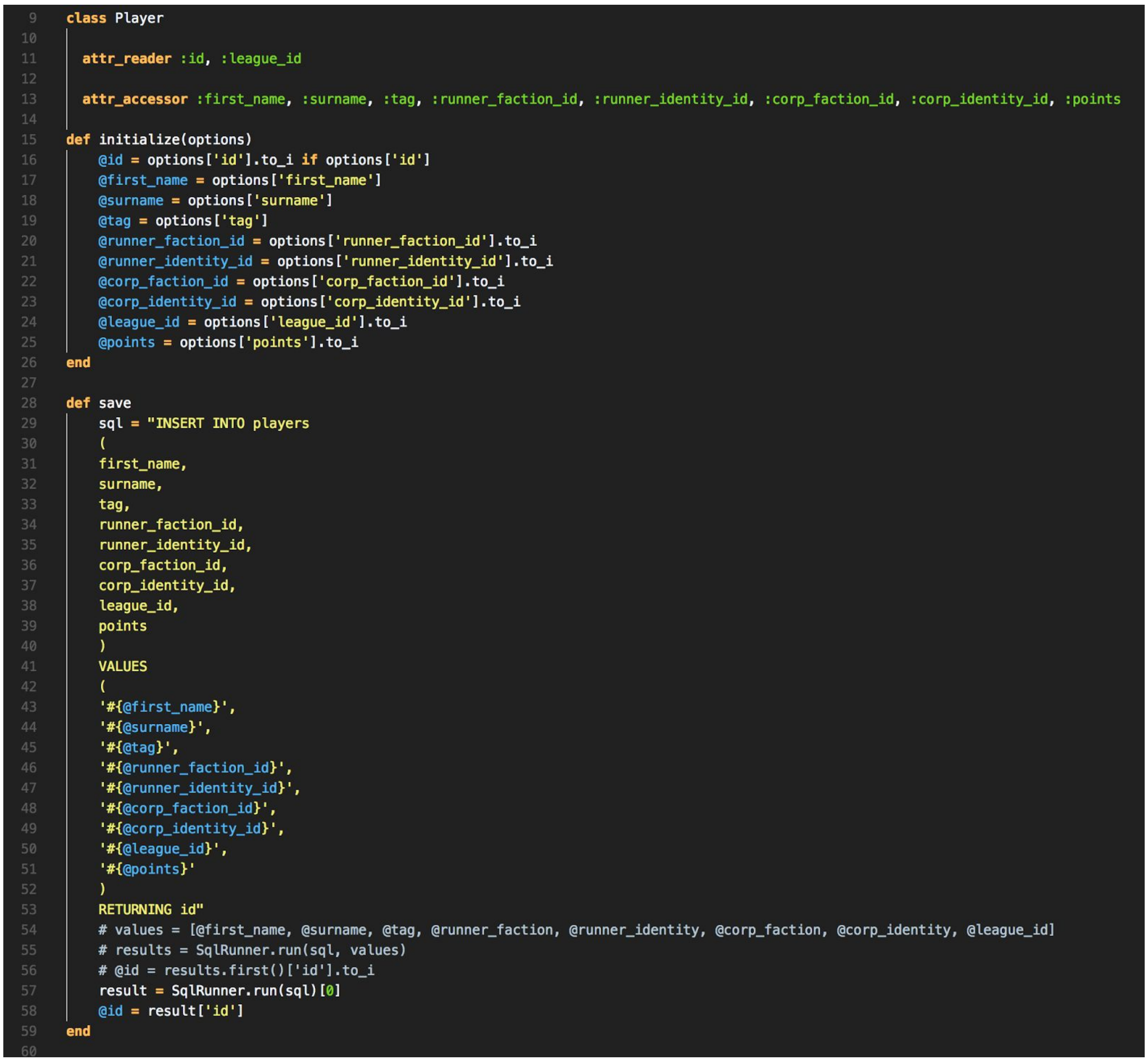
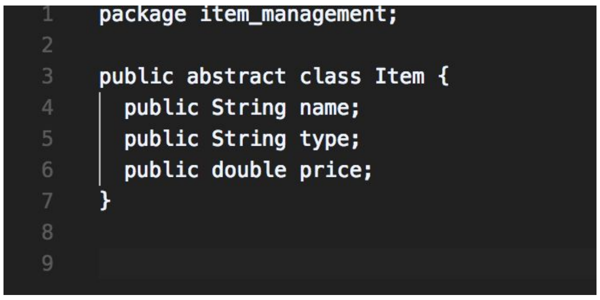
Chris Donnelly - Implementation and Testing Unit (SQA PDA: Software Development) - E14

Unit I & T - I.T 1 - A Screenshot of Encapsulation in a Program

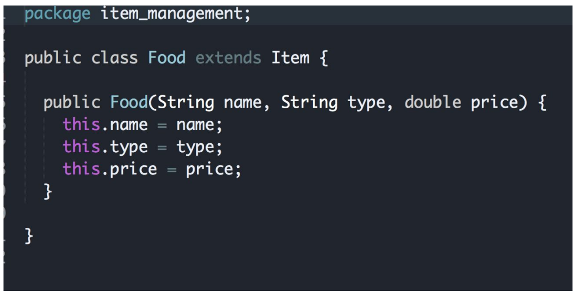


```ruby
class Player

  attr_reader :id, :league_id

  attr_accessor :first_name, :surname, :tag, :runner_faction_id, :runner_identity_id, :corp_faction_id, :corp_identity_id, :points

def initialize(options)
    @id = options['id'].to_i if options['id']
    @first_name = options['first_name']
    @surname = options['surname']
    @tag = options['tag']
    @runner_faction_id = options['runner_faction_id'].to_i
    @runner_identity_id = options['runner_identity_id'].to_i
    @corp_faction_id = options['corp_faction_id'].to_i
    @corp_identity_id = options['corp_identity_id'].to_i
    @league_id = options['league_id'].to_i
    @points = options['points'].to_i
end

def save
    sql = "INSERT INTO players
    (
    first_name,
    surname,
    tag,
    runner_faction_id,
    runner_identity_id,
    corp_faction_id,
    corp_identity_id,
    league_id,
    points
    )
    VALUES
    (
    '#{@first_name}',
    '#{@surname}',
    '#{@tag}',
    '#{@runner_faction_id}',
    '#{@runner_identity_id}',
    '#{@corp_faction_id}',
    '#{@corp_identity_id}',
    '#{@league_id}',
    '#{@points}'
    )
    RETURNING id"
    # values = [@first_name, @surname, @tag, @runner_faction, @runner_identity, @corp_faction, @corp_identity, @league_id]
    # results = SqlRunner.run(sql, values)
    # @id = results.first()['id'].to_i
    result = SqlRunner.run(sql)[0]
    @id = result['id']
end
```

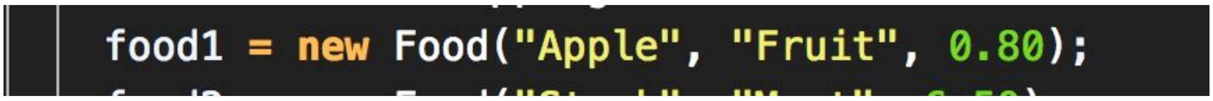
Unit I & T - I.T 2 - A Screenshot of Inheritance in a Program



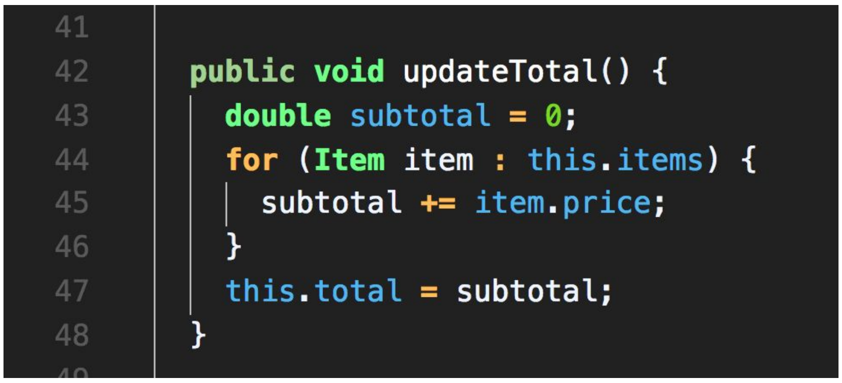
```java
package item_management;

public abstract class Item {
  public String name;
  public String type;
  public double price;
}
```



```java
package item_management;

public class Food extends Item {

  public Food(String name, String type, double price) {
    this.name = name;
    this.type = type;
    this.price = price;
  }

}
```



```java
food1 = new Food("Apple", "Fruit", 0.80);
```

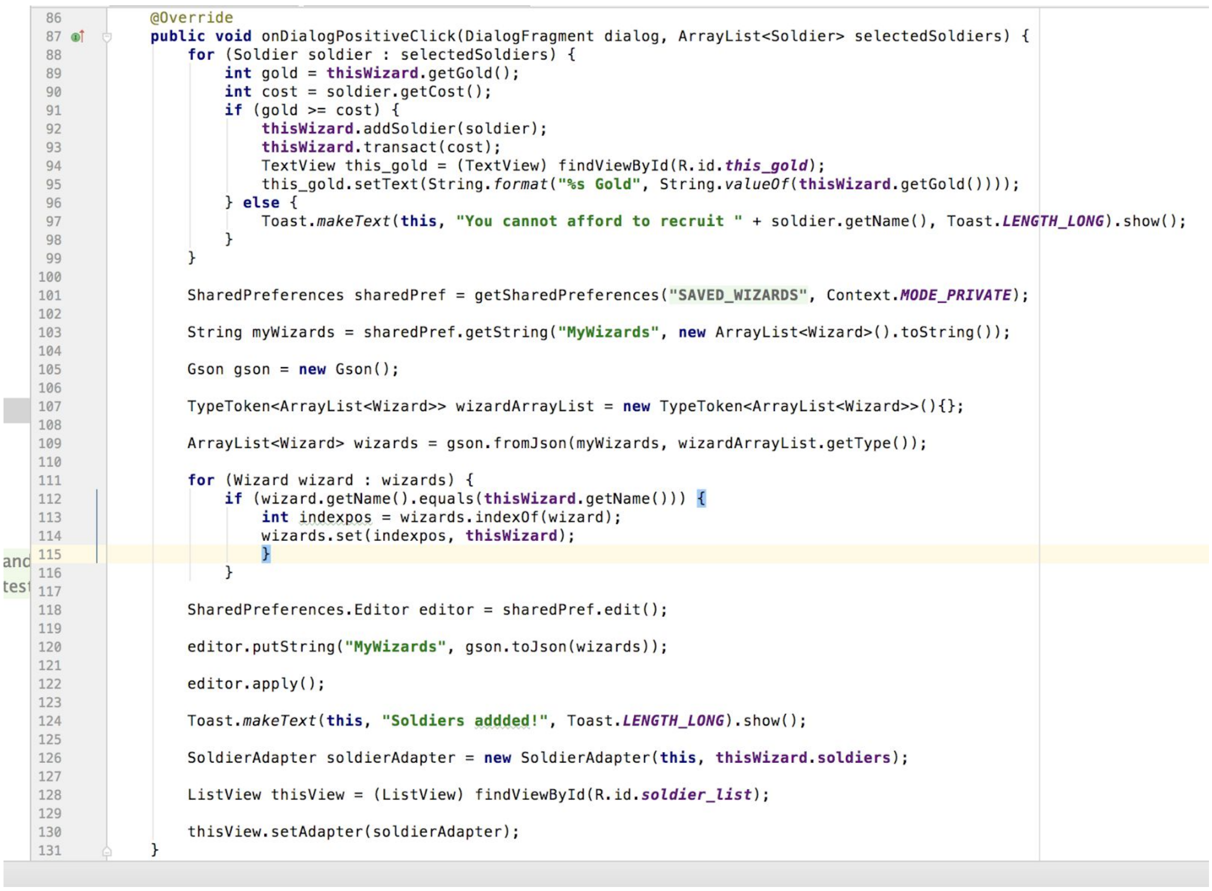


```java
public void updateTotal() {
    double subtotal = 0;
    for (Item item : this.items) {
        subtotal += item.price;
    }
    this.total = subtotal;
}
```

Unit I & T - I.T 3 - Demonstrate Searching Data in a Program

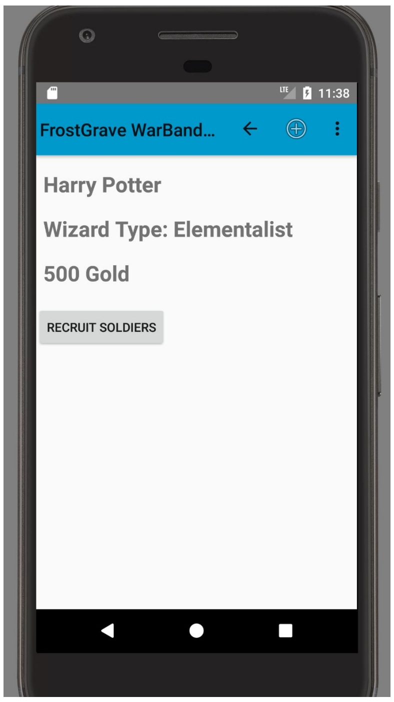
Take screenshots of;

- A function that searches data;



```java
@Override
public void onDialogPositiveClick(DialogFragment dialog, ArrayList<Soldier> selectedSoldiers) {
    for (Soldier soldier : selectedSoldiers) {
        int gold = thisWizard.getGold();
        int cost = soldier.getCost();
        if (gold >= cost) {
            thisWizard.addSoldier(soldier);
            thisWizard.transact(cost);
            TextView this_gold = (TextView) findViewById(R.id.this_gold);
            this_gold.setText(String.format("%s Gold", String.valueOf(thisWizard.getGold())));
        } else {
            Toast.makeText(this, "You cannot afford to recruit " + soldier.getName(), Toast.LENGTH_LONG).show();
        }
    }

    SharedPreferences sharedPref = getSharedPreferences("SAVED_WIZARDS", Context.MODE_PRIVATE);

    String myWizards = sharedPref.getString("MyWizards", new ArrayList<Wizard>().toString());

    Gson gson = new Gson();

    TypeToken<ArrayList<Wizard>> wizardArrayList = new TypeToken<ArrayList<Wizard>>(){};

    ArrayList<Wizard> wizards = gson.fromJson(myWizards, wizardArrayList.getType());

    for (Wizard wizard : wizards) {
        if (wizard.getName().equals(thisWizard.getName())) {
            int indexpos = wizards.indexOf(wizard);
            wizards.set(indexpos, thisWizard);
        }
    }

    SharedPreferences.Editor editor = sharedPref.edit();

    editor.putString("MyWizards", gson.toJson(wizards));

    editor.apply();

    Toast.makeText(this, "Soldiers addded!", Toast.LENGTH_LONG).show();

    SoldierAdapter soldierAdapter = new SoldierAdapter(this, thisWizard.soldiers);

    ListView thisView = (ListView) findViewById(R.id.soldier_list);

    thisView.setAdapter(soldierAdapter);
}
```

- The result of the function running;

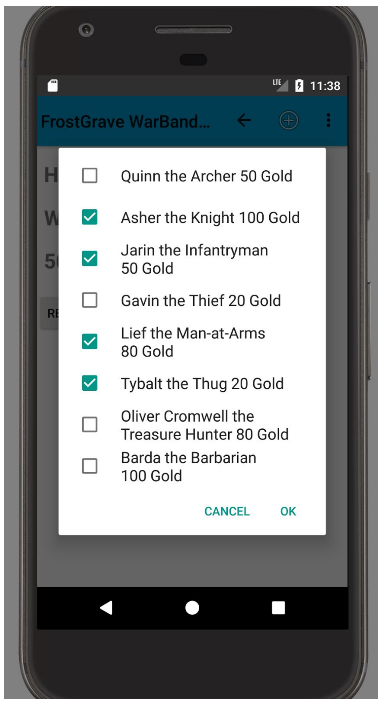

FrostGrave WarBand...

Harry Potter

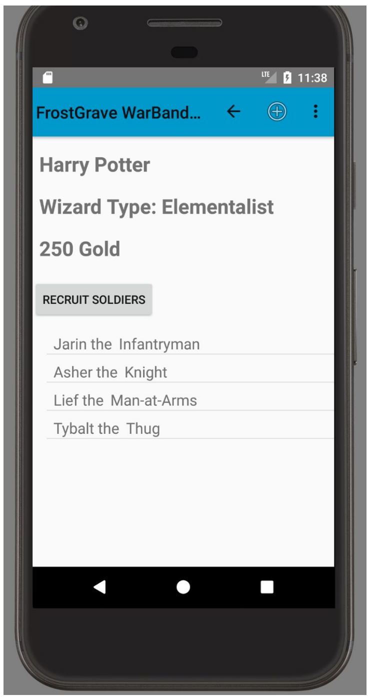
Wizard Type: Elementalist

500 Gold

RECRUIT SOLDIERS


- [ ] Quinn the Archer 50 Gold
- [x] Asher the Knight 100 Gold
- [x] Jarin the Infantryman 50 Gold
- [ ] Gavin the Thief 20 Gold
- [x] Lief the Man-at-Arms 80 Gold
- [x] Tybalt the Thug 20 Gold
- [ ] Oliver Cromwell the Treasure Hunter 80 Gold
- [ ] Barda the Barbarian 100 Gold

CANCEL OK


FrostGrave WarBand...

Harry Potter

Wizard Type: Elementalist

250 Gold

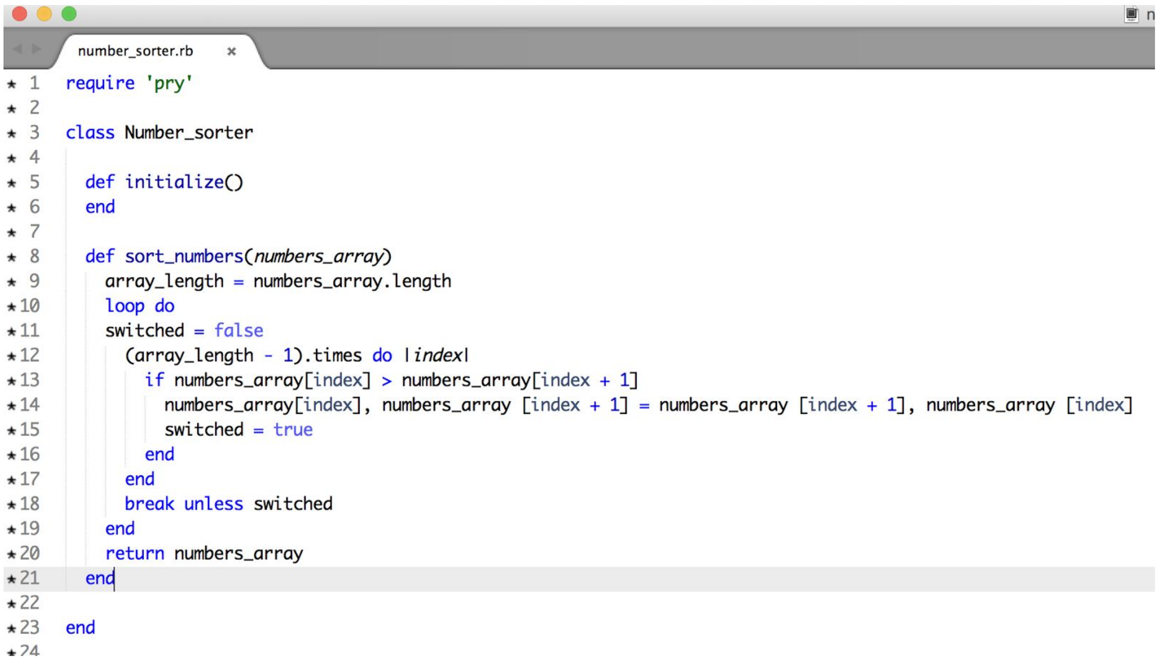
RECRUIT SOLDIERS

Jarin the Infantryman
Asher the Knight
Lief the Man-at-Arms
Tybalt the Thug

Unit I & T - I.T 4 - Demonstrate Sorting Data in a Program

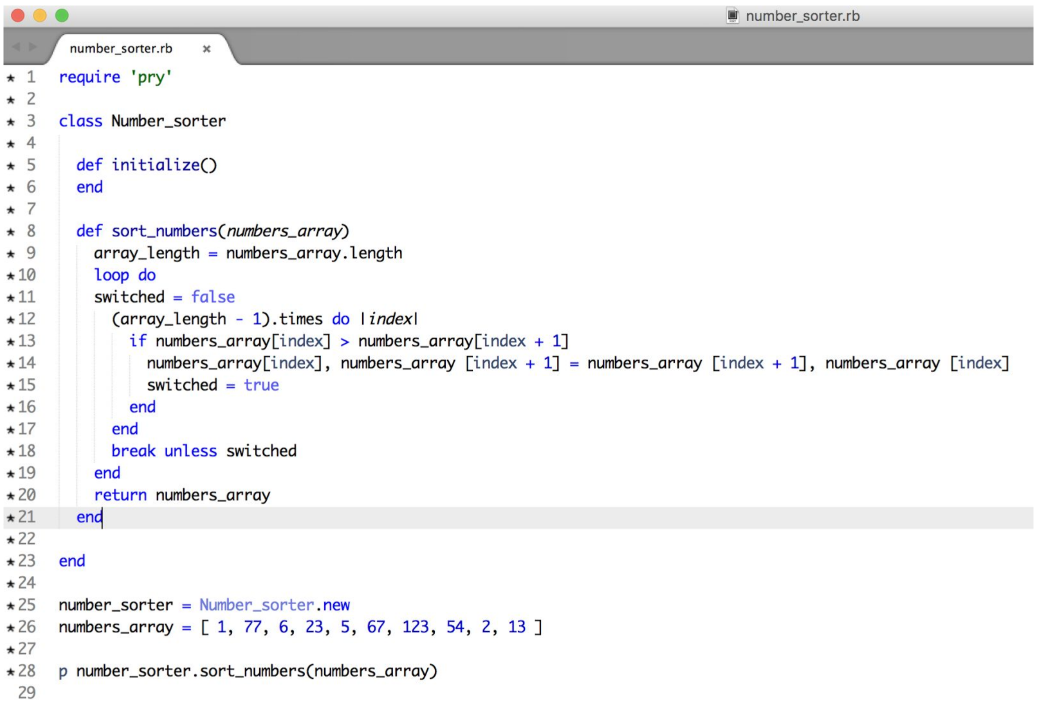
Take screenshots of;

- A function that sorts data;



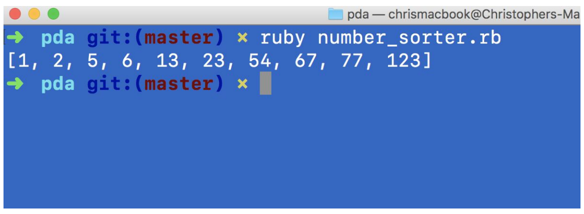
```ruby
require 'pry'

class Number_sorter

  def initialize()
  end

  def sort_numbers(numbers_array)
    array_length = numbers_array.length
    loop do
    switched = false
      (array_length - 1).times do |index|
        if numbers_array[index] > numbers_array[index + 1]
          numbers_array[index], numbers_array [index + 1] = numbers_array [index + 1], numbers_array [index]
          switched = true
        end
      end
      break unless switched
    end
    return numbers_array
  end

end
```

- The result of the function running;



number_sorter.rb

```ruby
require 'pry'

class Number_sorter

  def initialize()
  end

  def sort_numbers(numbers_array)
    array_length = numbers_array.length
    loop do
    switched = false
      (array_length - 1).times do |index|
        if numbers_array[index] > numbers_array[index + 1]
          numbers_array[index], numbers_array [index + 1] = numbers_array [index + 1], numbers_array [index]
          switched = true
        end
      end
      break unless switched
    end
    return numbers_array
  end

end

number_sorter = Number_sorter.new
numbers_array = [ 1, 77, 6, 23, 5, 67, 123, 54, 2, 13 ]

p number_sorter.sort_numbers(numbers_array)
```



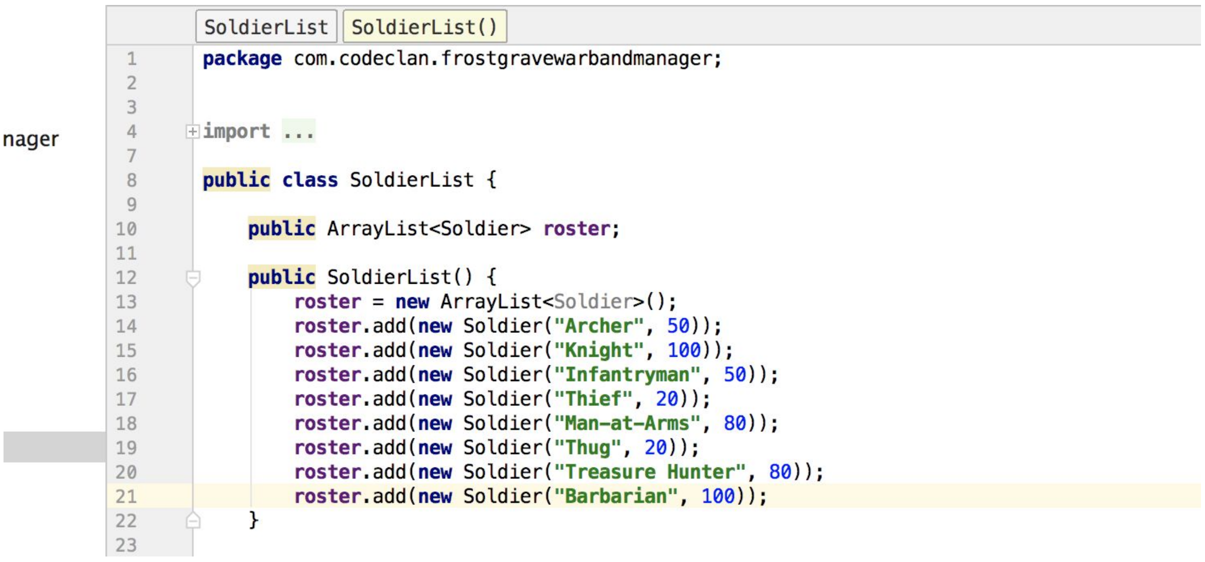
pda — chrismacbook@Christophers-Ma

```
➜  pda git:(master) ✗ ruby number_sorter.rb
[1, 2, 5, 6, 13, 23, 54, 67, 77, 123]
➜  pda git:(master) ✗
```

<u>Unit I & T - I.T 5 - Demonstrate the use of an Array in a Program</u>

Take screenshots of:

- An array in a program;

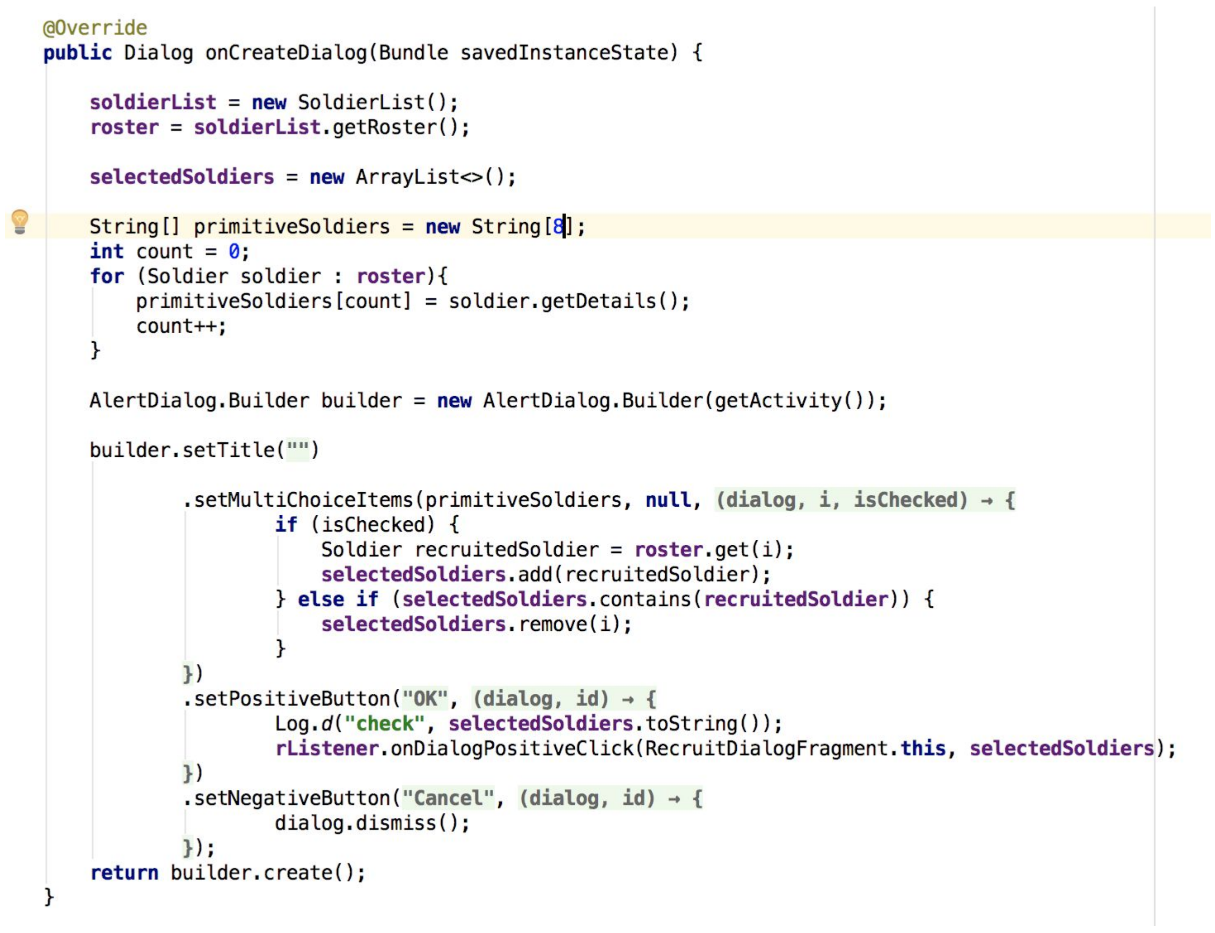


SoldierList | SoldierList()

nager

```java
package com.codeclan.frostgravewarbandmanager;


import ...

public class SoldierList {

    public ArrayList<Soldier> roster;

    public SoldierList() {
        roster = new ArrayList<Soldier>();
        roster.add(new Soldier("Archer", 50));
        roster.add(new Soldier("Knight", 100));
        roster.add(new Soldier("Infantryman", 50));
        roster.add(new Soldier("Thief", 20));
        roster.add(new Soldier("Man-at-Arms", 80));
        roster.add(new Soldier("Thug", 20));
        roster.add(new Soldier("Treasure Hunter", 80));
        roster.add(new Soldier("Barbarian", 100));
    }

```

- A function that uses the array;

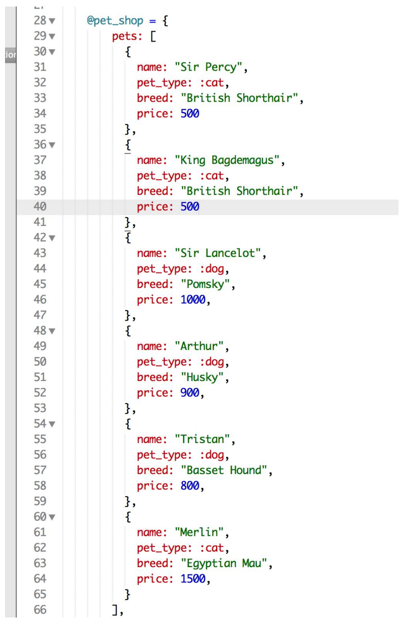


```
@Override
public Dialog onCreateDialog(Bundle savedInstanceState) {

    soldierList = new SoldierList();
    roster = soldierList.getRoster();

    selectedSoldiers = new ArrayList<>();

    String[] primitiveSoldiers = new String[8];
    int count = 0;
    for (Soldier soldier : roster){
        primitiveSoldiers[count] = soldier.getDetails();
        count++;
    }

    AlertDialog.Builder builder = new AlertDialog.Builder(getActivity());

    builder.setTitle("")

            .setMultiChoiceItems(primitiveSoldiers, null, (dialog, i, isChecked) → {
                    if (isChecked) {
                        Soldier recruitedSoldier = roster.get(i);
                        selectedSoldiers.add(recruitedSoldier);
                    } else if (selectedSoldiers.contains(recruitedSoldier)) {
                        selectedSoldiers.remove(i);
                    }
            })
            .setPositiveButton("OK", (dialog, id) → {
                    Log.d("check", selectedSoldiers.toString());
                    rListener.onDialogPositiveClick(RecruitDialogFragment.this, selectedSoldiers);
            })
            .setNegativeButton("Cancel", (dialog, id) → {
                    dialog.dismiss();
            });
    return builder.create();
}
```

Unit I & T - I.T 6 - Demonstrate the use of a Hash in a Program

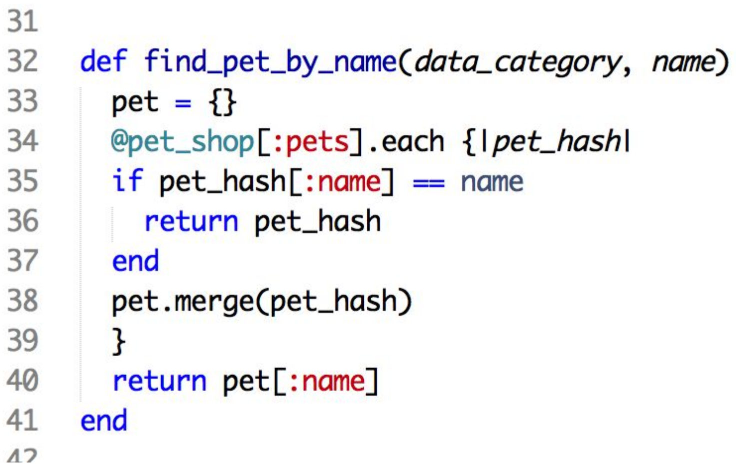
Take screenshots of:

- A hash in a program;



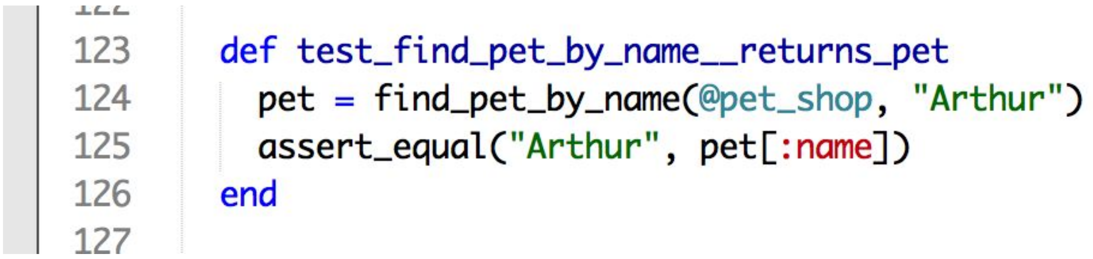
```
@pet_shop = {
    pets: [
      {
        name: "Sir Percy",
        pet_type: :cat,
        breed: "British Shorthair",
        price: 500
      },
      {
        name: "King Bagdemagus",
        pet_type: :cat,
        breed: "British Shorthair",
        price: 500
      },
      {
        name: "Sir Lancelot",
        pet_type: :dog,
        breed: "Pomsky",
        price: 1000,
      },
      {
        name: "Arthur",
        pet_type: :dog,
        breed: "Husky",
        price: 900,
      },
      {
        name: "Tristan",
        pet_type: :dog,
        breed: "Basset Hound",
        price: 800,
      },
      {
        name: "Merlin",
        pet_type: :cat,
        breed: "Egyptian Mau",
        price: 1500,
      }
    ],
```

- A function that uses the hash;

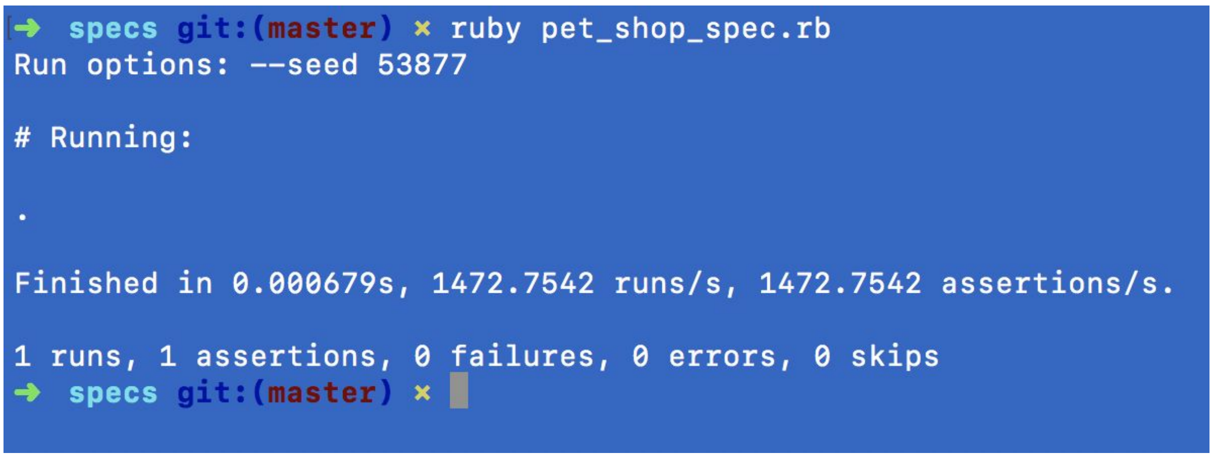


```ruby
def find_pet_by_name(data_category, name)
  pet = {}
  @pet_shop[:pets].each {|pet_hash|
  if pet_hash[:name] == name
    return pet_hash
  end
  pet.merge(pet_hash)
  }
  return pet[:name]
end
```

- The result of the function running;

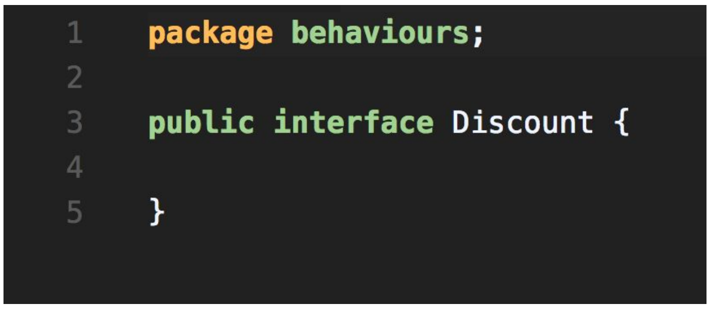


```ruby
def test_find_pet_by_name__returns_pet
  pet = find_pet_by_name(@pet_shop, "Arthur")
  assert_equal("Arthur", pet[:name])
end
```

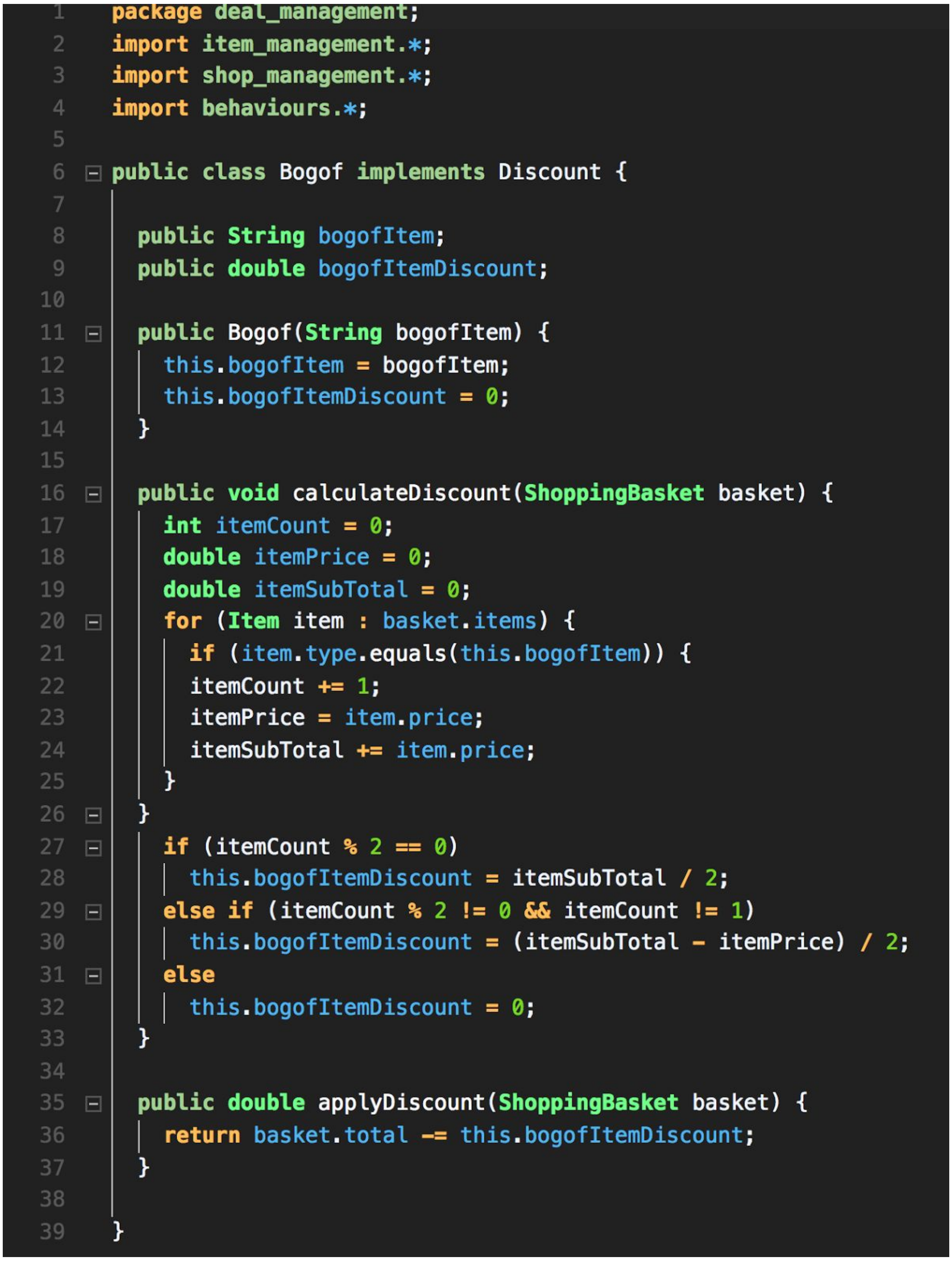


```
➜  specs git:(master) ✗ ruby pet_shop_spec.rb
Run options: --seed 53877

# Running:

.

Finished in 0.000679s, 1472.7542 runs/s, 1472.7542 assertions/s.

1 runs, 1 assertions, 0 failures, 0 errors, 0 skips
➜  specs git:(master) ✗ 
```

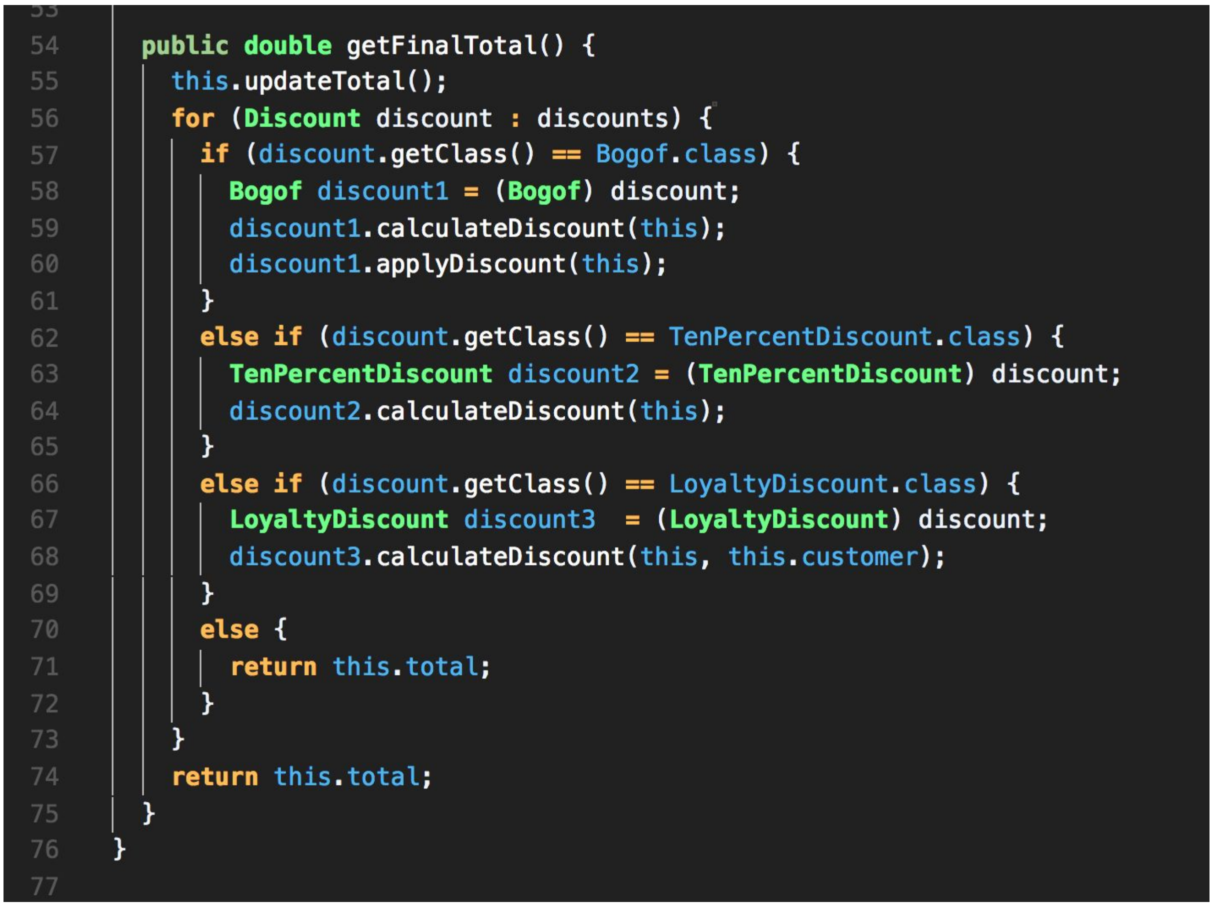
Unit I & T - I.T 7 - Demonstrate the use of Polymorphism in a Program



```java
package behaviours;

public interface Discount {

}
```



```java
package deal_management;
import item_management.*;
import shop_management.*;
import behaviours.*;

public class Bogof implements Discount {

  public String bogofItem;
  public double bogofItemDiscount;

  public Bogof(String bogofItem) {
    this.bogofItem = bogofItem;
    this.bogofItemDiscount = 0;
  }

  public void calculateDiscount(ShoppingBasket basket) {
    int itemCount = 0;
    double itemPrice = 0;
    double itemSubTotal = 0;
    for (Item item : basket.items) {
      if (item.type.equals(this.bogofItem)) {
      itemCount += 1;
      itemPrice = item.price;
      itemSubTotal += item.price;
      }
  }
    if (itemCount % 2 == 0)
      this.bogofItemDiscount = itemSubTotal / 2;
    else if (itemCount % 2 != 0 && itemCount != 1)
      this.bogofItemDiscount = (itemSubTotal - itemPrice) / 2;
    else
      this.bogofItemDiscount = 0;
  }

  public double applyDiscount(ShoppingBasket basket) {
    return basket.total -= this.bogofItemDiscount;
  }

}
```



```java
    public double getFinalTotal() {
        this.updateTotal();
        for (Discount discount : discounts) {
            if (discount.getClass() == Bogof.class) {
                Bogof discount1 = (Bogof) discount;
                discount1.calculateDiscount(this);
                discount1.applyDiscount(this);
            }
            else if (discount.getClass() == TenPercentDiscount.class) {
                TenPercentDiscount discount2 = (TenPercentDiscount) discount;
                discount2.calculateDiscount(this);
            }
            else if (discount.getClass() == LoyaltyDiscount.class) {
                LoyaltyDiscount discount3  = (LoyaltyDiscount) discount;
                discount3.calculateDiscount(this, this.customer);
            }
            else {
                return this.total;
            }
        }
        return this.total;
    }
}
```