COMPUTER SCIENCE DEPARTMENT FACULTY OF ENGINEERING AND TECHNOLOGY

ADVANCED PROGRAMMING COMP231

Lecturer :Farid Mohammad

Object Oriented

OBJECT-ORIENTED PROGRAMMING!

Object Oriented

Java support better organization called objects defined by classes

so lets put those in

classes

Bite Size

Bubble Gum

Candy Bars

Chocolates

Fruit Drops

Lotus Rolls

البرمجة باللغات مثل ال سي

كل الاسطر في ملف واحد او مجموعة functions

Creating a Class

```
class Candy {

}

class BiteSize{

}

class BubbleGum{

}


class CandyBars{

}

class Chocolate{

}

class FruitDrops{

}

class LotusRolls{


}
```

Enhance a Class

```
class Candy {
int type;
String color;
int countPerPackage;
float weight;


}
```

Enhance further

```
class Candy {
int type;
String color;
int countPerPackage;
float weight;
int price;
int barcode;

void setColor(String c){
color=c;
}


void setCountPerPackage(int cp){
countPerPackage=cp;
}

void setWeight(float w){
weight=w;
}

void setPrice(p){
  price=p;
}

int getPrice(){
   return price;
}
}
```

What other classes do we need

<table>
<tr><td></td><td>رحلة الشراء من المول  </td></tr>
<tr><td>Shopping Cart:

class ShoppingCart{

}</td><td>سلة الشراء
  </td></tr>
<tr><td>What method should we have in this class?
<table><tr><td>Adding new item for each type</td></tr><tr><td>remove item from the cart of a type</td></tr><tr><td></td></tr></table></td><td>How we define those two methods in this class?

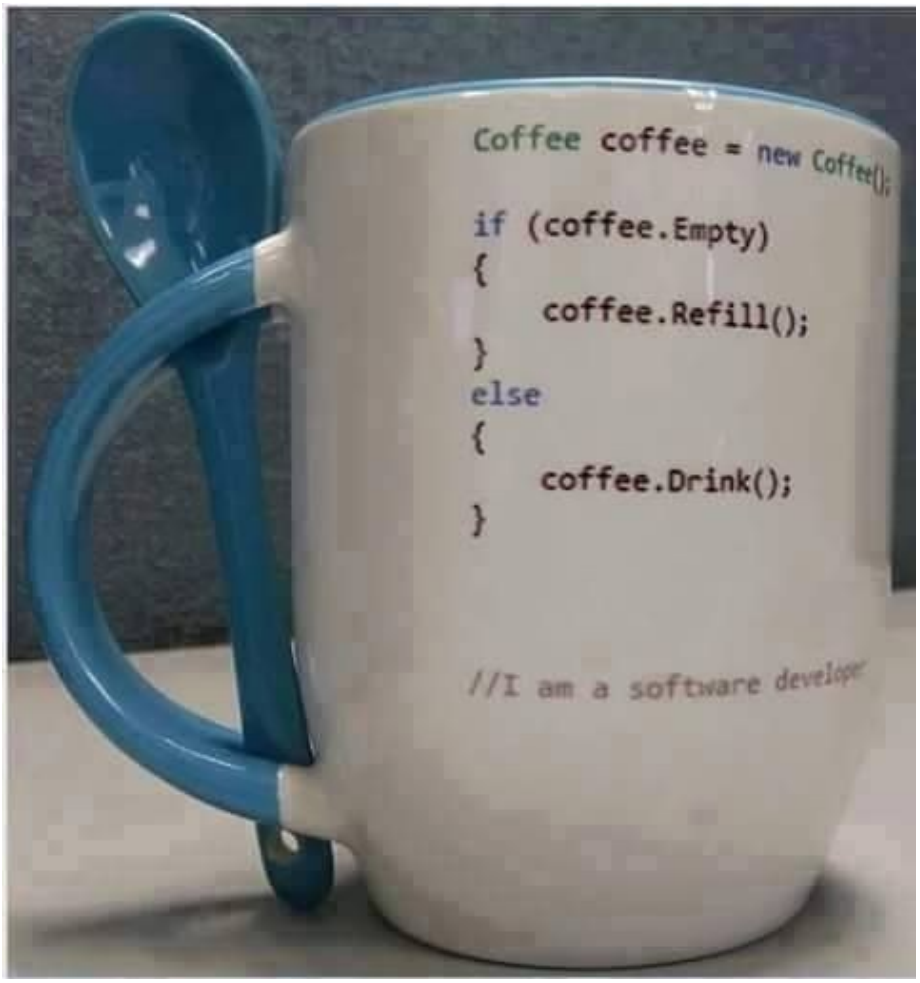
What attributes in ShoppingCart?</td></tr>
<tr><td></td><td></td></tr>
</table>

ShoppingCart

```
public class ShoppingCart{
  int total_items=0;
  int total_prices=0;


public void addCandy(BiteSize bs){
   total_items++;
    total_prices+=bs.getPrice();
}

public void removeCandy(BiteSize bs){
   total_prices-=bs.getPrice();
    total_items--;
}


public int getTotalItem(){
    return    total_items;
}

public int getTotalIPrices(){
    return    total_prices;
}

}
```

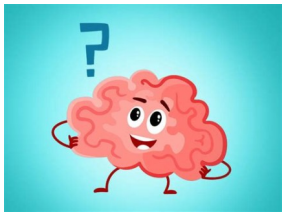
More methods for other types:

```
public void addCandy(BubbleGum bs){
   total_prices+=bs.getPrice();
    total_items++;
}


public void addCandy( CandyBars bs){
   total_prices+=bs.getPrice();
    total_items++;
}

public void addCandy( Chocolate bs){
   total_prices+=bs.getPrice();
    total_items++;
}

public void addCandy( FruitDrops bs){
   total_prices+=bs.getPrice();
    total_items++;
}


public void addCandy( LotusRolls bs){
   total_prices+=bs.getPrice();
    total_items++;
}
```


Coffee coffee = new Coffee();
if (coffee.Empty)
{
coffee.Refill();
}
else
{
coffee.Drink();
}
//I am a software develope

Main Class

To do something useful with the code above, we need a main method. We can put the main method in a separate file. The main class	
From Classes Come Objects	**How to create an Object From A Class ?**

`ShoppingCart sc=new ShoppingCart();`
 	
`This line creates a new object of type ShoppingCart and assign the value in sc`
	Object in Memory
	 

✓ An object is a thing. ✓ A class is a design plan for things of that kind.	

Some questions and answers

✓ **Can I have an object without having a class?** No, you can't. In Java, every object is an instance of a class.	
✓ **Can I have a class without having an object?** **Yes, you can.**	
✓ **After I've created a class and its instances, can I add more instances to the class?** Yes, you can.	
✓ **Can an object come from more than one class?** Bite your tongue! Maybe other object-oriented languages allow this nasty class cross-breeding, but in Java, it's strictly forbidden. That's one of the things that distinguishes Java from some of the languages that **preceded it.** **Java is cleaner, more uniform, and easier to understand.**	

Main Class

A class that have the main method called the Main Class	

`public class ProcessPurchase {`
 	
`public static void main(String[] args) {`
`ShoppingCart sc=new ShoppingCart();`	

How we can read purchases items and their prices from the user

`System.out.println("Add new Item to the Shopping Cart");`
`System.out.println("BiteSize bs");`
`System.out.println("BubbleGum bg");`
`System.out.println("CandyBars cb");`
`System.out.println("Chocolate ct");`
`System.out.println("FruitDrops fd");`
`System.out.println("LotusRolls lr");`
`Scanner scan=new Scanner(System.in);`
 	
`item=scan.next();`
`int price=0;`
 	
`if(!item.equals("e")) {`
`System.out.println("Enter Item Price");`
`price=scan.nextInt();`
`}`	

Ok now what?	
We need to call overloaded methods in `ShoppingCart` `sc.addCandy(bs);`	
But we need to create object of the right type first	

Let's create those objects:

`switch(item) {`
`case "bs":`
`BiteSize bs=new BiteSize();`
`bs.setPrice(price);`
`sc.addCandy(bs);`
`break;`
`case "bg":`
`BubbleGum bg=new BubbleGum();`
`bg.setPrice(price);`
`sc.addCandy(bg);`
`break;`
`case "cb":`
`CandyBars cb=new CandyBars();`	

```
                cb.setPrice(price);
                sc.addCandy(cb);
                break;
            case "ct":
                Chocolate ct=new Chocolate();
                ct.setPrice(price);
                sc.addCandy(ct);
                break;
            case "fd":
                FruitDrops fd=new FruitDrops();
                fd.setPrice(price);
                sc.addCandy(fd);
                break;
            case "lr":
                LotusRolls lr=new LotusRolls();
                lr.setPrice(price);
                sc.addCandy(lr);
                break;
            default:
                if(!item.equals("e"))
System.out.println("Wrong choice");

            }
```

Lets put it all together

```
import java.util.Scanner;

public class ProcessPurchase {

	public static void main(String[] args) {
		ShoppingCart sc=new ShoppingCart();

		Scanner scan=new Scanner(System.in);

		String item;
		do {
		System.out.println("Add new Item to the Shopping Cart");
		System.out.println("BiteSize bs");
		System.out.println("BubbleGum bg");
		System.out.println("CandyBars cb");
		System.out.println("Chocolate ct");
		System.out.println("FruitDrops fd");
		System.out.println("LotusRolls lr");


		 item=scan.next();
		 int price=0;

		 if(!item.equals("e")) {
			System.out.println("Enter Item Price");
			price=scan.nextInt();
		 }


		switch(item) {
		case "bs":
			BiteSize bs=new BiteSize();
			bs.setPrice(price);
			sc.addCandy(bs);
			break;
		case "bg":
			BubbleGum bg=new BubbleGum();
			bg.setPrice(price);
			sc.addCandy(bg);
			break;
		case "cb":
			CandyBars cb=new CandyBars();
			cb.setPrice(price);
			sc.addCandy(cb);
			break;
		case "ct":
			Chocolate ct=new Chocolate();
			ct.setPrice(price);
			sc.addCandy(ct);
			break;
		case "fd":
			FruitDrops fd=new FruitDrops();
```

```
                fd.setPrice(price);
                sc.addCandy(fd);
                break;
            case "lr":
                LotusRolls lr=new LotusRolls();
                lr.setPrice(price);
                sc.addCandy(lr);
                break;
            default:
                if(!item.equals("e")) System.out.println("Wrong choice");

            }

            }while(!item.equals("e"));


        /**
         * total purchase
         */

        System.out.printf("\n\t\t\tYou have purchases %d items in total\
n",sc.getTotalItem());
        System.out.printf("\n\t\t\t\t\t\tTotal prices: %d\
n",sc.getTotalIPrices());


        scan.close();
        }

}
```

Sample Run

```
Add new Item to the Shopping Cart
BiteSize bs
BubbleGum bg
CandyBars cb
Chocolate ct
FruitDrops fd
LotusRolls lr
bs
Enter Item Price
12
Add new Item to the Shopping Cart
BiteSize bs
BubbleGum bg
CandyBars cb
Chocolate ct
FruitDrops fd
LotusRolls lr
bs
Enter Item Price
13
Add new Item to the Shopping Cart
BiteSize bs
BubbleGum bg
CandyBars cb
Chocolate ct
FruitDrops fd
LotusRolls lr
lr
Enter Item Price
33
Add new Item to the Shopping Cart
BiteSize bs
BubbleGum bg
CandyBars cb
Chocolate ct
FruitDrops fd
LotusRolls lr
e

                You have purchases 3 items in total

                            Total prices: 3
```