```java
package ucboulder.ooad.project3.decorator;

import ucboulder.ooad.project3.entities.Tool;

// Decorator class
// It has an instance of Tool which can be either tool alone or decorator tool.


public class AddOnDecorator extends Tool {

	 public Tool tool;
	public int price;
	public int nytNo;


	public AddOnDecorator(Tool tool,String name,int price,int nytNo) {

		this.tool = tool;
		this.name = name;
		this.price = price;
		this.nytNo = nytNo;
	}
	// Gives the price of the entire decorator tool.
	public int getPrice() {

		return price+tool.getPrice();

	}

	@Override
	public void increaseCount() {


	}




}
```

```java
package ucboulder.ooad.project3.entities;

import ucboulder.ooad.project3.decorator.AddOnDecorator;

//AddOn option for tool.
public class AccessoryKit extends AddOnDecorator{

	static final int price = 18;

	public AccessoryKit(Tool tool, String name,int nytNo) {
		super(tool, name, price,nytNo);
	}


}
```

```java
package ucboulder.ooad.project3.entities;

import java.util.ArrayList;

//Its a type of Customer. It extends from the Customer class
public class BusinessCustomer extends Customer {

	public BusinessCustomer(String name) {

		this.name=name;

	}


}
```

```java
package ucboulder.ooad.project3.entities;

import java.util.ArrayList;

//Its a type of Customer. It extends from the Customer class

public class CasualCustomer extends Customer {

	public CasualCustomer(String name) {

		this.name=name;
	}



}
```

```java
package ucboulder.ooad.project3.entities;


public class Concrete extends Tool{

	public static final  int price = 6;

	public static int count = 5;

	private  Concrete(String name,int nytNo){
		this.name = name;
		this.nytNo=nytNo;
	}

	//Returns an instance of this particular tool from a pool on an on-demand
basis.
	// Typically reducing the inventory count of this tool when rented.
	public static Concrete getInstance(String name,int nytNo) {

		if(count > 1) {
			count--;
			return new Concrete(name,nytNo);
		}
		else {
			return null;
		}

	}
	// Returns the rental price for this tool
	@Override
	public int getPrice() {

		return price*nytNo;
	}

	//ReStock Inventory when returned by customer
	@Override
	public void increaseCount() {

		 count = count+1;
	}


}
```

```java
package ucboulder.ooad.project3.entities;

import java.util.ArrayList;
import java.util.List;

public abstract class Customer {

	String name;
	List<RentalRecord> rr = new ArrayList<RentalRecord>();

	public void addRentalRecord(RentalRecord rentalRecord) {
		rr.add(rentalRecord);
	}
	public List<RentalRecord> getRentalRecordList(){
		return rr;
	}
	public int getNumberOfToolsRented() {
		int sum = 0;
		for (int i = 0; i < rr.size() ; i++) {

			sum += rr.get(i).getDocoratedTools().size();
		}
		return sum;
	}
	public String getCustomerName() {
		return this.name;
	}
}
```

```
package ucboulder.ooad.project3.entities;

import ucboulder.ooad.project3.decorator.AddOnDecorator;

//AddOn option for tool.

public class ExtensionCord extends AddOnDecorator{

	static final int price = 16;

	public ExtensionCord(Tool tool, String name,int nytNo) {
		super(tool, name, price,nytNo);
	}


}
```

```java
package ucboulder.ooad.project3.entities;

public class Painting extends Tool{

	public static final int price = 5;

	public static int count = 5;

	private Painting(String name,int nytNo){
		this.name=name;
		this.nytNo=nytNo;
	}

	//Returns an instance of this particular tool from a pool on an on-demand
basis.
		// Typically reducing the inventory count of this tool when rented.
	public static Painting getInstance(String name,int nytNo) {

		if(count > 1) {
			count--;
			return new Painting(name,nytNo);
		}
		else {
			return null;
		}

	}

	// Returns the rental price for this tool
	@Override
	public int getPrice() {

		return price*nytNo;
	}

	//ReStock Inventory when returned by customer
	@Override
	public void increaseCount() {

		 count = count+1;
	}

}
```

```java
package ucboulder.ooad.project3.entities;

public class Plumbing extends Tool{

	public static final int price = 7;

	public static int count = 5;

	private Plumbing(String name,int nytNo){
		this.name=name;
		this.nytNo=nytNo;
	}

	//Returns an instance of this particular tool from a pool on an on-demand 
basis.
		// Typically reducing the inventory count of this tool when rented.
	public static Plumbing getInstance(String name,int nytNo) {

		if(count > 1) {
			count--;
			return new Plumbing(name,nytNo);
		}
		else {
			return null;
		}

	}

	// Returns the rental price for this tool
	@Override
	public int getPrice() {

		return price*nytNo;
	}

	//ReStock Inventory when returned by customer
	@Override
	public void increaseCount() {

		 count = count+1;
	}
}
```

```java
package ucboulder.ooad.project3.entities;

import ucboulder.ooad.project3.decorator.AddOnDecorator;

//AddOn option for tool.

public class ProtectiveGearPack extends AddOnDecorator{

	static final int price = 25;

	public ProtectiveGearPack(Tool tool, String name,int nytNo) {
		super(tool, name, price,nytNo);
	}


}
```

```java
package ucboulder.ooad.project3.entities;

import java.util.ArrayList;

//Its a type of Customer. It extends from the Customer class

public class RegularCustomer extends Customer {

	public RegularCustomer(String name) {

		this.name=name;
	}



}
```

```java
package ucboulder.ooad.project3.entities;


import java.util.ArrayList;
import java.util.List;

import ucboulder.ooad.project3.decorator.AddOnDecorator;

public class RentalRecord {

	public int returnedDay;
	boolean status;

	public RentalRecord(int returnDay) {
		this.returnedDay = returnDay;
		this.status = true;
	}

	public List<AddOnDecorator> docoratedTools = new ArrayList<AddOnDecorator>();

	//Returns the list of rented tools for a given rental record
	public List<AddOnDecorator> getDocoratedTools() {
		return docoratedTools;
	}

	//Adds a tool to the list of tools for a given rental record
	public void addDocoratedTools(AddOnDecorator docoratedTool) {
	//	System.out.println(docoratedTool);
		docoratedTools.add(docoratedTool);


	}

	// Checks whether rental record is active or not
	public boolean getStatus() {
		return this.status;
	}
	public void setStatus(boolean status) {
		this.status = status;
	}

	// Returns the price of the entire rental record.
	public int getTotalPrice() {

		int price=0;

		for(int i=0;i<docoratedTools.size();i++) {

		price = price + docoratedTools.get(i).getPrice();

		}

		return price;
	}
	// Handles return of a rental and increases the inventory count for the tools 
present in the rental record.
	public void returnRental(Customer c) {
		
		//System.out.println(docoratedTools.size());
		
		for(int i=0;i<docoratedTools.size();i++) {
			
			AddOnDecorator t = docoratedTools.get(i);
			
			while(!t.getClass().getSimpleName().equals("Painting") ||
					!t.getClass().getSimpleName().equals("Plumbing") ||
					!t.getClass().getSimpleName().equals("Woodwork") ||
					!t.getClass().getSimpleName().equals("Yardwork") ||
					!t.getClass().getSimpleName().equals("Concrete")) {
				
				//System.out.println(t.getClass().getSimpleName());
				
				
				
				
				if(t.tool instanceof AddOnDecorator) {
					t = (AddOnDecorator) t.tool;
				}
				else {
					break;
				}
			}
			//System.out.println("--"+t.tool);
			t.tool.increaseCount();
			System.out.println(t.tool.name + " returned by customer " + 
c.name);
			
			
			}
			
		
	}
	
	
	
}
```

```java
package ucboulder.ooad.project3.entities;

import ucboulder.ooad.project3.decorator.AddOnDecorator;

public abstract class Tool {

	public String name;
	public abstract int getPrice();
	public abstract void increaseCount();
	public int nytNo;

}
```

```
package ucboulder.ooad.project3.entities;

public class Woodwork extends Tool{

	public static final int price = 8;

	public static int count = 5;

	private Woodwork(String name,int nytNo){
		this.name=name;
		this.nytNo = nytNo;
	}

	//Returns an instance of this particular tool from a pool on an on-demand
basis.
		// Typically reducing the inventory count of this tool when rented.
	public static Woodwork getInstance(String name,int nytNo) {

		if(count > 1) {
			count--;
			return new Woodwork(name,nytNo);
		}
		else {
			return null;
		}

	}

	// Returns the rental price for this tool
	@Override
	public int getPrice() {

		return price*nytNo;
	}

	//ReStock Inventory when returned by customer
	@Override
	public void increaseCount() {

		 count = count+1;
	}
}
```

```
package ucboulder.ooad.project3.entities;

public class Yardwork extends Tool{

	public static final int price = 8;

	public static int count = 4;

	private Yardwork(String name,int nytNo){
		this.name=name;
		this.nytNo = nytNo;
	}

	//Returns an instance of this particular tool from a pool on an on-demand 
basis.
		// Typically reducing the inventory count of this tool when rented.
	public static Yardwork getInstance(String name,int nytNo) {

		if(count > 1) {
			count--;
			return new Yardwork(name,nytNo);
		}
		else {
			return null;
		}

	}

	// Returns the rental price for this tool
	@Override
	public int getPrice() {

		return price*nytNo;
	}

	//ReStock Inventory when returned by customer
	@Override
	public void increaseCount() {

		 count = count+1;
	}

}
```

```
package ucboulder.ooad.project3.factory;


import java.util.ArrayList;

import ucboulder.ooad.project3.entities.BusinessCustomer;
import ucboulder.ooad.project3.entities.CasualCustomer;
import ucboulder.ooad.project3.entities.Customer;

import ucboulder.ooad.project3.entities.RegularCustomer;
import ucboulder.ooad.project3.entities.RentalRecord;


public class CustomerFactory {


	//Customer Factory method. Takes care of creating required Customers depending
on the type.
	public static Customer getCustomer(String type,String name) {

		if(type.equalsIgnoreCase("RegularCustomer")) return new
RegularCustomer(name);
		if(type.equalsIgnoreCase("BusinessCustomer")) return new
BusinessCustomer(name);
		if(type.equalsIgnoreCase("CasualCustomer")) return new
CasualCustomer(name);
		else
		return null;

	}

}
```

```java
package ucboulder.ooad.project3.factory;

import ucboulder.ooad.project3.entities.AccessoryKit;
import ucboulder.ooad.project3.entities.Concrete;
import ucboulder.ooad.project3.entities.ExtensionCord;
import ucboulder.ooad.project3.entities.Painting;
import ucboulder.ooad.project3.entities.Plumbing;
import ucboulder.ooad.project3.entities.ProtectiveGearPack;
import ucboulder.ooad.project3.entities.Tool;
import ucboulder.ooad.project3.entities.Woodwork;
import ucboulder.ooad.project3.entities.Yardwork;


public class ToolFactory {

	//Tool Factory Class. Takes care of creating required Tools depending on the 
type.

	public static Tool getTool(String type,String name,int nytNo) {

		if(type.equalsIgnoreCase("Concrete")) return 
Concrete.getInstance(name,nytNo);
		if(type.equalsIgnoreCase("Painting")) return 
Painting.getInstance(name,nytNo);
		if(type.equalsIgnoreCase("Plumbing")) return 
Plumbing.getInstance(name,nytNo);

		if(type.equalsIgnoreCase("Woodwork")) return 
Woodwork.getInstance(name,nytNo);
		if(type.equalsIgnoreCase("Yardwork")) return 
Yardwork.getInstance(name,nytNo);

		else
		return null;
	}

	//AddOn Factory Class. Takes care of creating required AddOn depending on the 
type.

	public static Tool getAddOnTool(String type,String name,Tool tool,int nytNo) {

		if(type.equalsIgnoreCase("AccessoryKit")) return new AccessoryKit(tool, 
name,nytNo);
		if(type.equalsIgnoreCase("ExtensionCord")) return new 
ExtensionCord(tool, name,nytNo);
		if(type.equalsIgnoreCase("ProtectiveGearPack")) return new 
ProtectiveGearPack(tool, name,nytNo);
		else
		return null;

	}



}
```

```java
package ucboulder.ooad.project3.main;

import java.util.List;
import java.util.Random;

import ucboulder.ooad.project3.entities.Customer;
import ucboulder.ooad.project3.entities.RentalRecord;
import ucboulder.ooad.project3.factory.CustomerFactory;
import ucboulder.ooad.project3.factory.ToolFactory;
import ucboulder.ooad.project3.entities.Tool;
import ucboulder.ooad.project3.decorator.AddOnDecorator;
import ucboulder.ooad.project3.entities.Painting;
import ucboulder.ooad.project3.entities.Plumbing;
import ucboulder.ooad.project3.entities.Woodwork;
import ucboulder.ooad.project3.entities.Yardwork;
import ucboulder.ooad.project3.entities.Concrete;



public class Main {

	public static void main(String[] args) {

		//Creating store instance
		Store store = new Store();


		//Creating list of 12 Customers
		for(int i = 0; i < 12 ; i++) {
			Random randomGenerator = new Random();
			int randomNumber = randomGenerator.nextInt(3) + 1;
			Customer customer;
			if (randomNumber == 1) {
				customer = CustomerFactory.getCustomer("BusinessCustomer", 
"Customer" + (i));
			}
			else if (randomNumber ==2) {
				customer = CustomerFactory.getCustomer("RegularCustomer", 
"Customer" + (i));
			}
			else {
				customer = CustomerFactory.getCustomer("CasualCustomer", 
"Customer" + (i));
			}
			store.addCustomersToStore(customer);
		}


		int sumOfAllDaysTotal = 0;

		//Simulation for first 34 days
		for (int i = 0 ; i < 34 ; i++) {
			System.out.println("------ List of tools in the inventory -----
");
```

```
                    if (Painting.count!=0) {
                          System.out.println("Painting tools " +Painting.count);
                    }
                    if (Plumbing.count!=0) {
                          System.out.println("Plumbing tools " +Plumbing.count);
                    }
                    if (Concrete.count!=0) {
                          System.out.println("Concrete tools " +Concrete.count);
                    }
                    if (Woodwork.count!=0) {
                          System.out.println("Woodwork tools " +Woodwork.count);
                    }
                    if (Yardwork.count!=0) {
                          System.out.println("Yardwork tools " +Yardwork.count);
                    }

                    List<Customer> customers = store.getCustomersFromStore();


                    // Handles the return of rentals on particular day
                    for(Customer customer:customers)
                    {
                          List<RentalRecord> customerRentalRecordList =
customer.getRentalRecordList();
                          int temp=-1;
                          for (int j = 0; j < customerRentalRecordList.size(); j++)
{
                                int noOfRentedNights =
customerRentalRecordList.get(j).getDocoratedTools().get(0).nytNo;
                                int boughtDay =
customerRentalRecordList.get(j).returnedDay;

                                if(i==(boughtDay+noOfRentedNights)) {


      customerRentalRecordList.get(j).returnRental(customer);

      customerRentalRecordList.get(j).setStatus(false);
                                }
                                else {
                                      //System.out.println("Customer name: " +
customer.getCustomerName() + " has rented ");

                                }

                          }

                    }




                    System.out.println("++++++++++++++++++++++++++++++++++++ Day
number : " +(i+1)+" ++++++++++++++++++++++++++++++++++++");
```

```
                    Random randomGenerator = new Random();
                    int randomNumber = randomGenerator.nextInt(11) + 1;

                    Customer customer = customers.get(randomNumber);
                    int totalPrice = -1;
                    System.out.println("Customer " + customer.getCustomerName() + "
comes to the store");
                    if
(customer.getClass().getSimpleName().equals("BusinessCustomer")) {
                        int numberOfTools = 3;
                        int numberOfNights = 7;
                        RentalRecord rentalRecord = new RentalRecord(i);

                        for (int j = 0 ; j < 3 ; j++) {
                            int rnum = randomGenerator.nextInt(5) + 1;
                            Tool tool;
                            Tool at;
                            String type;
                            if (rnum ==1) {
                                tool = ToolFactory.getTool("Painting",
"Painting"+(rnum+1), numberOfNights);
                                type = "Painting";
                                at =
ToolFactory.getAddOnTool("ExtensionCord", "Extension cord" +(rnum+1),
tool,numberOfNights);

                            }
                            else if (rnum ==2) {
                                 tool = ToolFactory.getTool("Plumbing",
"Plumbing"+(rnum+1), numberOfNights);
                                 type = "Plumbing";
                                 at =
ToolFactory.getAddOnTool("ProtectiveGearPack", "Protective Gear Pack" +(rnum+1),
tool,numberOfNights);

                            }
                            else if (rnum ==3) {
                                 tool = ToolFactory.getTool("Concrete",
"Concrete"+(rnum+1),numberOfNights);
                                 type = "Concrete";
                                 at =
ToolFactory.getAddOnTool("AccessoryKit", "Accessory Kit "+(rnum+1),
tool,numberOfNights);
                            }
                            else if (rnum ==4) {
                                 tool = ToolFactory.getTool("Woodwork",
"Woodwork"+(rnum+1), numberOfNights);
                                 type = "Woodwork";
                                 at =
ToolFactory.getAddOnTool("ExtensionCord","Extension cord" + (rnum+1),
tool,numberOfNights);
                            }
                            else {
```

```
                                        tool = ToolFactory.getTool("Yardwork",
"Yardwork"+(rnum+1), numberOfNights);
                                        type = "Yardwork";
                                        at =
ToolFactory.getAddOnTool("ProtectiveGearPack","Protective Gear Pack" + (rnum+1),
tool,numberOfNights);
                                  }
                                  if (tool != null)
                                  rentalRecord.addDocoratedTools((AddOnDecorator)at);
                                  if (tool == null) {
                                        System.out.println("He was not able to
purchase tool " + type);
                                  }
                                  else if (tool != null){
                                        System.out.println("He purchases tool " +
tool.name);

                                        System.out.println("Add Ons" + at.name);
                                  }



                            }
                            customer.addRentalRecord(rentalRecord);
                            totalPrice = rentalRecord.getTotalPrice();
                            sumOfAllDaysTotal = sumOfAllDaysTotal + totalPrice;
                            System.out.println("Price of the rental record is "
+totalPrice);

                      }
                      if
(customer.getClass().getSimpleName().equals("RegularCustomer")) {

                            int numberOfTools = 2;
                            int numberOfNights = 4;
                            RentalRecord rentalRecord = new RentalRecord(i);

                            for (int j = 0 ; j < 2 ; j++) {
                                  int rnum = randomGenerator.nextInt(5) + 1;
                                  Tool tool;
                                  Tool at;
                                  String type;
                                  if (rnum ==1) {
                                        tool = ToolFactory.getTool("Painting",
"Painting"+(rnum+1), numberOfNights);
                                        type = "Painting";
                                        at =
ToolFactory.getAddOnTool("ExtensionCord", "Extension cord" +(rnum+1),
tool,numberOfNights);

                                  }
                                  else if (rnum ==2) {
                                         tool = ToolFactory.getTool("Plumbing",
"Plumbing"+(rnum+1), numberOfNights);
                                         type = "Plumbing";
                                        at = 
ToolFactory.getAddOnTool("ProtectiveGearPack", "Protective Gear Pack" +(rnum+1), 
tool,numberOfNights);

                                }
                                else if (rnum ==3) {
                                        tool = ToolFactory.getTool("Concrete", 
"Concrete"+(rnum+1),numberOfNights);
                                        type = "Concrete";
                                        at = 
ToolFactory.getAddOnTool("AccessoryKit", "Accessory Kit "+(rnum+1), 
tool,numberOfNights);
                                }
                                else if (rnum ==4) {
                                        tool = ToolFactory.getTool("Woodwork", 
"Woodwork"+(rnum+1), numberOfNights);
                                        type = "Woodwork";
                                        at = 
ToolFactory.getAddOnTool("ExtensionCord","Extension cord" + (rnum+1), 
tool,numberOfNights);
                                }
                                else {
                                       tool = ToolFactory.getTool("Yardwork", 
"Yardwork"+(rnum+1), numberOfNights);
                                       type = "Yardwork";
                                       at = 
ToolFactory.getAddOnTool("ProtectiveGearPack","Protective Gear Pack" + (rnum+1), 
tool,numberOfNights);
                                }
                                if (tool != null)
                                rentalRecord.addDocoratedTools((AddOnDecorator)at);
                                if (tool == null) {
                                       System.out.println("He was not able to 
purchase tool " + type);
                                }
                                else if (tool != null){
                                       System.out.println("He purchases tool " + 
tool.name);

                                       System.out.println("Add Ons" + at.name);
                                }



                          }
                          customer.addRentalRecord(rentalRecord);
                          totalPrice = rentalRecord.getTotalPrice();
                          sumOfAllDaysTotal += totalPrice;
                          System.out.println("Price of the rental record is " 
+totalPrice);
                  }
                          if 
(customer.getClass().getSimpleName().equals("CasualCustomer")) {

                          int numberOfTools = 1;
                          int numberOfNights = 2;
```

```
                                RentalRecord rentalRecord = new RentalRecord(i);

                                for (int j = 0 ; j < 2 ; j++) {
                                        int rnum = randomGenerator.nextInt(5) + 1;
                                        Tool tool;
                                        Tool at;
                                        String type;
                                        if (rnum ==1) {
                                                tool = ToolFactory.getTool("Painting",
"Painting"+(rnum+1), numberOfNights);
                                                type = "Painting";
                                                at =
ToolFactory.getAddOnTool("ExtensionCord", "Extension cord" +(rnum+1),
tool,numberOfNights);

                                        }
                                        else if (rnum ==2) {
                                                 tool = ToolFactory.getTool("Plumbing",
"Plumbing"+(rnum+1), numberOfNights);
                                                 type = "Plumbing";
                                                 at =
ToolFactory.getAddOnTool("ProtectiveGearPack", "Protective Gear Pack" +(rnum+1),
tool,numberOfNights);

                                        }
                                        else if (rnum ==3) {
                                                 tool = ToolFactory.getTool("Concrete",
"Concrete"+(rnum+1),numberOfNights);
                                                 type = "Concrete";
                                                 at =
ToolFactory.getAddOnTool("AccessoryKit", "Accessory Kit "+(rnum+1),
tool,numberOfNights);
                                        }
                                        else if (rnum ==4) {
                                                 tool = ToolFactory.getTool("Woodwork",
"Woodwork"+(rnum+1), numberOfNights);
                                                 type = "Woodwork";
                                                 at =
ToolFactory.getAddOnTool("ExtensionCord","Extension cord" + (rnum+1),
tool,numberOfNights);
                                        }
                                        else {
                                                tool = ToolFactory.getTool("Yardwork",
"Yardwork"+(rnum+1), numberOfNights);
                                                type = "Yardwork";
                                                at =
ToolFactory.getAddOnTool("ProtectiveGearPack","Protective Gear Pack" + (rnum+1),
tool,numberOfNights);
                                        }
                                        if (tool != null)
                                        rentalRecord.addDocoratedTools((AddOnDecorator)at);
                                        if (tool == null) {
                                                System.out.println("He was not able to
purchase tool " + type);
                                        }
```

```
                                                else if (tool != null){
                                                        System.out.println("He purchases tool " +
tool.name);

                                                        System.out.println("Add Ons" + at.name);
                                                }


                                        }
                                        customer.addRentalRecord(rentalRecord);
                                        totalPrice = rentalRecord.getTotalPrice();
                                        sumOfAllDaysTotal += totalPrice;
                                        System.out.println("Price of the rental record is "
+totalPrice);



                                }
                                        if (totalPrice != -1) {
                                                System.out.println("Total price that the store made
for day is " +totalPrice);

                                        }




                        }
                        List<Customer> customers = store.getCustomersFromStore();//
                                System.out.println("type is " +type);

                        System.out.println("total number of completed rentals");
                        int totalCompletedRentals = 0;
                        int countBusiness = 0;
                        int countRegular = 0;
                        int countCasual = 0;
                        for(Customer customer:customers)
                        {
                                String type = customer.getClass().getSimpleName();

                                List<RentalRecord> customerRentalRecordList =
customer.getRentalRecordList();
                                int temp=-1;
                                for (RentalRecord rentalRecord: customerRentalRecordList) {
                                        if (rentalRecord.getStatus() == false ) {
                                                totalCompletedRentals++;

                                                if (type.equals("RegularCustomer")){
                                                        countRegular++;
                                                }
                                                else if (type.equals("BusinessCustomer")) {
                                                        countBusiness++;
                                                }
                                                else if (type.equals("CasualCustomer")) {
                                                        countCasual++;
                                }
                            }
                        }

                }

                //Summary of 34 days
                System.out.println("++++++++++++++++++++++++++++++++++++++ Summary 
++++++++++++++++++++++++++++++");
                System.out.println("Total completed rentals is " 
+totalCompletedRentals);
                System.out.println("Total completed rentals for regular customer is " 
+countRegular);
                System.out.println("Total completed rentals for business customer is " 
+countBusiness);
                System.out.println("Total completed rentals for casual customer is " 
+countCasual);
                System.out.println("Total profit of store for the entire period " 
+sumOfAllDaysTotal);

        }

}
```

```java
package ucboulder.ooad.project3.main;


import java.util.ArrayList;
import java.util.List;
import java.util.Random;

import ucboulder.ooad.project3.decorator.AddOnDecorator;
import ucboulder.ooad.project3.entities.Customer;
import ucboulder.ooad.project3.entities.Painting;
import ucboulder.ooad.project3.entities.RentalRecord;
import ucboulder.ooad.project3.entities.Tool;
import ucboulder.ooad.project3.factory.CustomerFactory;

import ucboulder.ooad.project3.factory.ToolFactory;

public class Store {
	public List<Customer> customers = new ArrayList<Customer>();

	//Adds the customers to store
	public void addCustomersToStore(Customer customer) {
		this.customers.add(customer);
	}

	// returns list of customers present in store
	public List<Customer> getCustomersFromStore(){
		return this.customers;
	}



}
```

```
package ucboulder.ooad.project3.Test;

import org.junit.Test;

import junit.framework.TestCase;
import ucboulder.ooad.project3.decorator.AddOnDecorator;
import ucboulder.ooad.project3.entities.AccessoryKit;
import ucboulder.ooad.project3.entities.BusinessCustomer;
import ucboulder.ooad.project3.entities.CasualCustomer;
import ucboulder.ooad.project3.entities.Concrete;
import ucboulder.ooad.project3.entities.Customer;
import ucboulder.ooad.project3.entities.ExtensionCord;
import ucboulder.ooad.project3.entities.Painting;
import ucboulder.ooad.project3.entities.Plumbing;
import ucboulder.ooad.project3.entities.ProtectiveGearPack;
import ucboulder.ooad.project3.entities.RegularCustomer;
import ucboulder.ooad.project3.entities.RentalRecord;
import ucboulder.ooad.project3.entities.Tool;
import ucboulder.ooad.project3.entities.Woodwork;
import ucboulder.ooad.project3.entities.Yardwork;
import ucboulder.ooad.project3.factory.CustomerFactory;
import ucboulder.ooad.project3.factory.ToolFactory;

//This test class contains
public class MyUnitTest extends TestCase{

	Tool  t;


	public void setUp(){

	   }


	// This test method checks if right instance of tool is created for right type
(Method name : getTool)
	@Test
	public void test_1(){
		assertTrue(ToolFactory.getTool("Concrete","ConcreteTool 1", 3)
instanceof Concrete &&
					ToolFactory.getTool("Painting","PaintingTool 1", 3)
instanceof Painting &&
					ToolFactory.getTool("Plumbing","PlumbingTool 1", 3)
instanceof Plumbing &&
					ToolFactory.getTool("Woodwork","WoodworkTool 1", 3)
instanceof Woodwork &&
					ToolFactory.getTool("Yardwork","YardworkTool 1", 3)
instanceof Yardwork &&
					ToolFactory.getTool("Yardhjsjdbwork","RandomTool 1",
3)==null);
	   }
	//This test method checks if right instance of add on tool is created for 
right type  (Method name : getAddOnTool)
	@Test
	  public void test_2(){
		
		assertTrue(ToolFactory.getAddOnTool("AccessoryKit","AccessoryKit 1",t, 
3)
			  instanceof AccessoryKit && 
			  ToolFactory.getAddOnTool("ExtensionCord","ExtensionCord 1",t, 3) 
instanceof 
			  ExtensionCord && 
			  ToolFactory.getAddOnTool("ProtectiveGearPack","ProtectiveGearPack 
1",t, 3)
			  instanceof ProtectiveGearPack && 
			  ToolFactory.getAddOnTool("Yardhjsjdbwork","RandomTool 1",t, 
3)==null);
		}


	//This test method check whether particular tool  is moved out of inventory 
after purchase  (Method name : getInstance)
	@Test
	public void test_3(){
		
		 int before = Concrete.count;
		 Concrete.getInstance("Concrete Tool", 3);
		assertTrue((before-1) == Concrete.count);
	  }
	
	//This test method checks price of decorated tool is valid or not  (Method 
name : getTotalPrice)
	@Test
	public void test_4(){
		 Tool t1 = ToolFactory.getTool("Yardwork","YardworkTool 1", 3) ;
		 AddOnDecorator dc = (AddOnDecorator) 
ToolFactory.getAddOnTool("ProtectiveGearPack","ProtectiveGearPack 1",t1, 3) ;
		 RentalRecord rr = new RentalRecord(3);
		 rr.addDocoratedTools(dc);
		 
		 assertTrue(rr.getTotalPrice() == 8*3+25);
	  }
	
	//This test method checks for return are properly handles by increasing the 
inventory count (Method name : returnRental)
	@Test
	public void test_5(){
		 Customer c = CustomerFactory.getCustomer("BusinessCustomer","Business 
Customer 1");
		 
		 Tool t1 = ToolFactory.getTool("Yardwork","YardworkTool 1", 3) ;
		 AddOnDecorator dc = (AddOnDecorator) 
ToolFactory.getAddOnTool("ProtectiveGearPack","ProtectiveGearPack 1",t1, 3) ;
		 RentalRecord rr = new RentalRecord(3);
		 rr.addDocoratedTools(dc);
		 int countBeforeReturing = Yardwork.count;
```

```
            rr.returnRental(c);
            int countAfterReturning = Yardwork.count;
            assertTrue(countBeforeReturing+1 == countAfterReturning);
        }

    //This test method checks for price of decorated tool only (Method name :
getPrice)
    @Test
    public void test_6(){

                Tool t1 = ToolFactory.getTool("Plumbing","Plumbing Tool 1", 3) ;
            AddOnDecorator dc = (AddOnDecorator)
ToolFactory.getAddOnTool("ExtensionCord","ExtensionCord 1",t1, 3) ;

            int price = dc.getPrice();

            assertTrue(price == 7*3+16);

        }

    //This test method checks for price of  tool only (Method name : getPrice)
    @Test
    public void test_7(){
             Tool t1 = ToolFactory.getTool("Woodwork","Woodwork Tool 1", 3) ;
             assertTrue(t1.getPrice()==Woodwork.price*3);
        }

    //This test method checks if right instance of Customer is created for right
type (Method name : getCustomer)
    @Test
    public void test_8(){
             assertTrue(CustomerFactory.getCustomer("RegularCustomer","Regular
Customer 1") instanceof RegularCustomer &&
                          CustomerFactory.getCustomer("BusinessCustomer","Business
Customer 1") instanceof BusinessCustomer &&
                          CustomerFactory.getCustomer("CasualCustomer","Casual
Customer 1") instanceof CasualCustomer &&
                          CustomerFactory.getCustomer("Yardhjsjdbwork","Random
Cutomser 1")==null);

        }

    //This test method check whether particular tool  is moved out of inventory
after purchase  (Method name : getInstance)
    @Test
    public void test_9(){
             int before = Painting.count;
             Painting.getInstance("Painting Tool", 3);
          assertTrue((before-1) == Painting.count);
        }

    //This test method check for validity of number of tools rented by customer
(Method name : getNumberOfToolsRented)
    @Test
    public void test_10(){
```

```
                Tool t1 = ToolFactory.getTool("Plumbing","Plumbing Tool 1", 3) ;
             AddOnDecorator dc = (AddOnDecorator)
ToolFactory.getAddOnTool("ExtensionCord","ExtensionCord 1",t1, 3) ;

             Tool t2 = ToolFactory.getTool("Woodwork","Woodwork Tool 1", 3) ;
             AddOnDecorator dc2 = (AddOnDecorator)
ToolFactory.getAddOnTool("AccessoryKit","AccessoryKit 1",t2, 3) ;

             Customer c = CustomerFactory.getCustomer("BusinessCustomer","Business
Customer 1");

             RentalRecord rr = new RentalRecord(3);
             rr.addDocoratedTools(dc);

             RentalRecord rr2 = new RentalRecord(3);
             rr2.addDocoratedTools(dc2);

             c.addRentalRecord(rr);
             c.addRentalRecord(rr2);

            assertTrue(c.getNumberOfToolsRented()==2);


          }

}
```