CS401 Modern Programming Practices

Midterm

Date : 05.03.16 **Max.Marks : 80** **Duration : 2.00 Hours**

Name: Mohamed Ahmed Gharieb Elsayed StudentId: 985136

Part I(T/F) (5)	Part II (MCQ) (10) (8)	Part III Theory (15) (17)	Part IV Design (30)	Part V Coding (20)	Part IV SCI - EC (3)	Total (80)
4	8	15	28.5	19.5	0	75

I. Say True or False (1 Points Each)

True 1. A Model is an abstract description of a system or process.

True 2. Temporary relationships are modeled as dependencies.

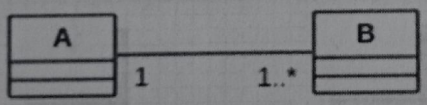
False 3. Analysis is concerned with what and Design is concerned with how.

True 4. Object diagrams don't have multiplicities.

False 5. Java supports multiple inheritance and no diamond problem among the classes.

II. Multiple Choice Questions.

6. Consider the following class diagram: **(2 Points)**



Which of the following statements is (are) correct? Circle all that are correct.

a. Each instance of the class B contains a list of instances of A.

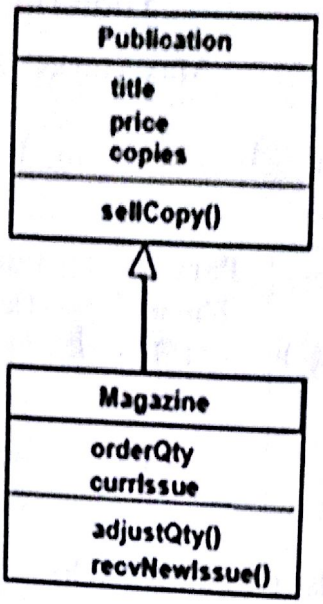
b. Each instance of the class A contains a list of instances of B.

c. A is a property/attribute of B.

d. If an instance of A has been created, at least one instance of B has also been created.

e. After an instance of A has finished creating instances of B (either inside its constructor or inside some other constructing method), the instances become null automatically after A's constructor or A's constructing method returns.

7. Look at the diagram below and decide which of the pairs of operations shown are legal and illegal. **(2 Points)**



a) `Publication p = new Publication(...);`
`p.sellCopy();`

b) `Publication p = new Publication(...);`
`p.recvNewIssue();`

c) `Publication p = new Magazine(...);`
`p.sellCopy();`

d) `Publication p = new Magazine(...);`
`p.recvNewIssue();`

a) Answer: legal

b) Answer: illegal

c) Answer: legal

d) Answer: illegal

8. In the given code, which of the following is correct regarding the relationship between Person and Email? Circle the right one. **(1 Point)**

```
public class Email {

public void sendEmail(String subject, String message){
System.out.println("Subject : " + subject + "\n Message : " + message);

}

}

public class Person {

public void greetFriend(Email ob){

ob.sendEmail("Hello", "Hello my friend :)");

}

}
```

A. There is a dependency from Person to Email

B. There is a one-way association from Person to Email

C. There is a two-way association between Person to Email

D. Not possible to determine from the code shown.

9. What happens when the following code is compiled/run? Select only one right choice. **(1 Point)**

```
public class Main {
    public static void main(String[] args) {
        MySuperSuper mss = new MyClass();
        mss.myMethod();
    }
}

public class MyClass extends MySuper {
    public void myMethod() {
        super.myMethod();
        System.out.print("Goodbye");
    }
}
public class MySuper extends MySuperSuper {
    public void myMethod() {
        System.out.println("Hello ");
        super.myMethod();
    }
}
public class MySuperSuper {
    public void myMethod() {
        System.out.println("Super super");
    }
}
```

a. Compiler error

b. Runtime exception is thrown

c. The following is printed to the console:

```
Hello
Super super
Goodbye
```

d. The following is printed to the console:

```
Super super
```

e. The following is printed to the console:
Super super
Hello
Goodbye

10. The difficulties with the `ExtendedHashSet` example, discussed in class, exemplify the following (circle *the best answer*). (Code shown below.) **(2 Points)**

A. The Ripple Effect

B. The incompatibility between the OO principles of Encapsulation and Inheritance

C. A subclass may break because of a change in its superclass

D. All of the above.

E. None of the above.

```
public class ExtendedHashSet<T> extends HashSet<T> {
    //The number of attempted element insertions since its creation --
    //this value will not be modified when elements are removed
    private int addCount = 0;

    public ExtendedHashSet() {}

    @Override
    public boolean add(T a) {
        addCount++;
        return super.add(a);
    }
    @Override
    public boolean addAll(Collection<? extends T> c) {
        //addCount += c.size();
        return super.addAll(c);
    }
    public int getAddCount() {
        return addCount;
    }
}
```

III. Theory Part – To test the understanding of concepts.

11. Fill the table with appropriate answers. Answer should be **Compile time or Run time.** Don't specify other words. [4 Points]

Specific Aspects	Answer
Early Binding happens at _____	compile time
Late Binding happens at _____	Run time

Binding of private, static and final methods happens at _____	Compile Time
Binding of Overriding methods happens at _____	run Time

12. What is the order of execution for the given block? Write output as per the execution order. [3 Points]

```
public class ClassE {
        static int superInt = 10;

    static {
        System.out.println("superInt = " + superInt);
        }

    ClassE(){
        System.out.println("Running ClassE constructor");
        }

    {
        System.out.println("Running super object initialization block");
    }

    public static void main(String[] args) {
            new SubClass();
        }

}

class SubClass extends ClassE {

    static int subInt = 20;

    static {
        System.out.println("subInt =" + subInt);
        }

    SubClass(){
            System.out.println("Running SubClass constructor");
        }
        {
        System.out.println("Running sub object initialization block");
        }
        }
```

Output :

- superInt = 10
- subInt = 20
- Running class E constructor
- Running super object initialization block
- Running subclass constructor
- Running sub object initialization block

13. Comparison of Actual Classes, Abstract Classes and Interfaces with the given specific Property? Just fill the table cells with words **"Allowed"** or **"Not Allowed"**. (Please don't use other words) [4 Points]

Aspects	Actual Class	Abstract class	Interface(Pre Java 8)
Constructor	Allowed	Allowed	Not Allowed
Instances (Objects) of this class can be created directly is allowed or not.	Allowed	Not Allowed	Not Allowed
Inclusion of abstract methods	Not Allowed	Allowed	Allowed
Inclusion of instance variables	Allowed	Allowed	Not Allowed

14. Choose the right term for the given definition. [4 Points]

[LSP , Reflexive Association, Inheritance, Association]

a. A class having a reference of another class is called Association.

b. Relationship between two or more objects of the same class is called as Reflexive Association

c. As a way to generalize data and behavior of related classes is known as inheritance.

d. Subtypes must be substitutable for their base classes type is known as LSP

15. Give an example from the Java API library where the following concepts are applied. [2 Points]

a. Interfaces and abstract classes used together. Answer : Array List

b. Factory Method. Answer : Collections or Specify the method.

IV. Design activities.

16. Draw the sequence diagram for the given problem. [5 Points]

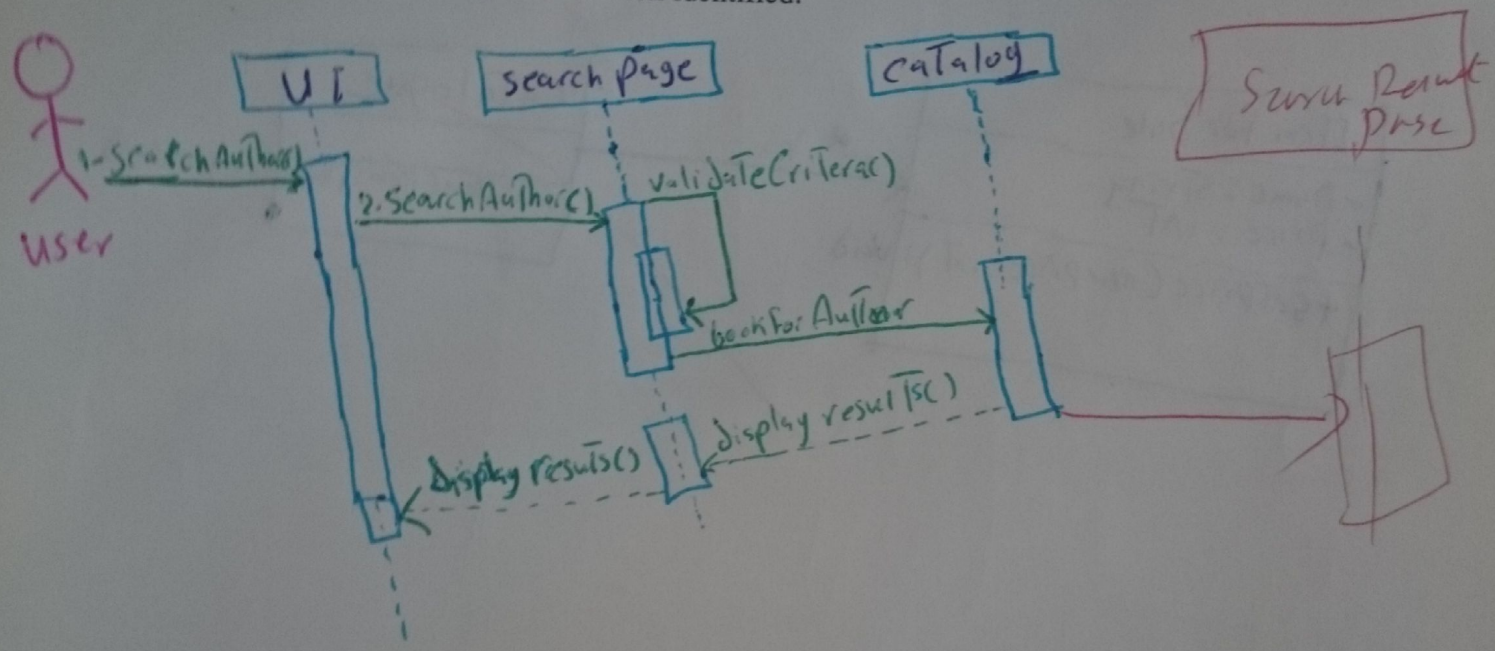
The Customer specifies an author on the Search Page and then presses the Search button.

The Search Page validates the Customer's search criteria. If the customer did not enter the name of the author, the Search Page displays an error message to that effect and prompts the customer to re-enter an author name.

The Search Page searches the Catalog for books associated with the specified author.

When the search is complete, the Catalog displays the search results on the Search Results Page.

Note : Sequence diagrams make use of an actor to initiate action; the actor talks to a UI class; and the UI class talks to a Controller class. The Controller class then communicates with the use cases that have been identified.

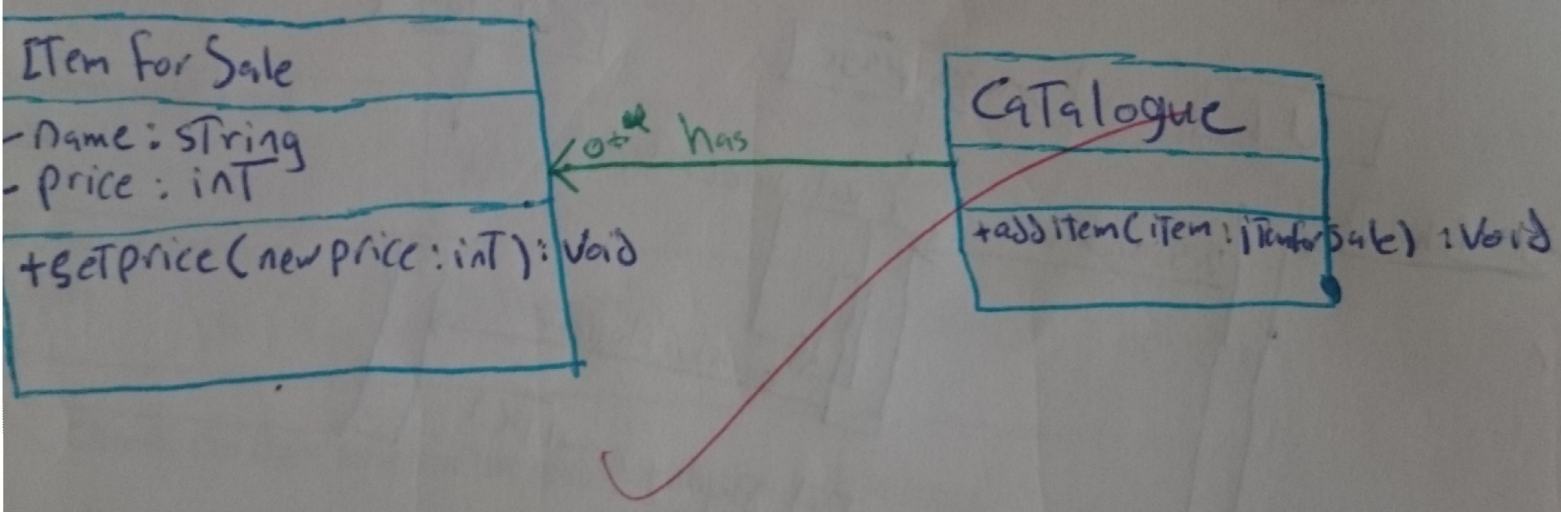


17. Draw a class diagram to represent a class called 'Catalogue' and a class called 'ItemForSale' as defined below. [10 Points]

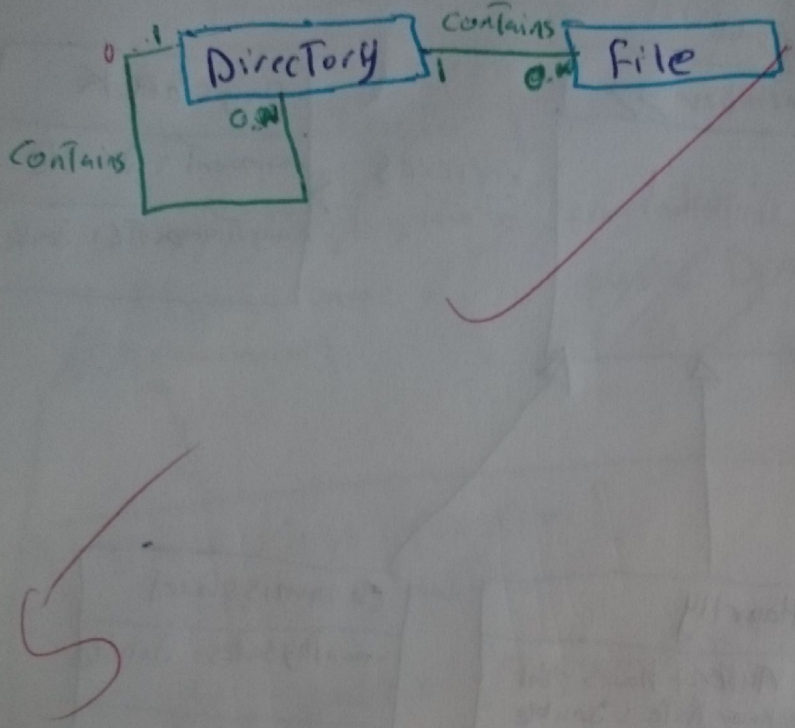
'ItemForSale' has an attribute 'name' of type String and an attribute 'price' of type int. It also has a method setPrice() which takes an integer parameter 'newPrice'.

'Catalogue' has an attribute 'listOfItems'(ie the items currently held in the catalogue) of type ItemForSale. Zero or more items can be stored in the catalogue for the attribute 'listOfItems'. 'Catalogue' also has one method additem() which takes an argument 'item' of type 'itemForSale' and adds this item to the 'listofitems'.

Draw this on a class diagram by showing association, multiplicity, appropriate visibility modifiers for attributes and methods with their argument.



18. In a Windows file system, a directory may contain files and other directories. Represent these concepts with classes Directory and File in a class diagram and show association relationships. You do not need to specify attributes or operations in these classes. Associations should be provided with multiplicities and names. [5 Points]



19. Perform the design for the given problem. [10 Points]

An organization has many members.

On the first of the month every member receives a paycheck. So organization need to calculate the payroll for the members.

Unfortunately, the way the gross amount of the paycheck is calculated depends on if the type of the member.

Salaried members receive a twelfth of the amount from their annual salary.

Hourly members receive an amount equal to the number of hours worked times the member's hourly rate.

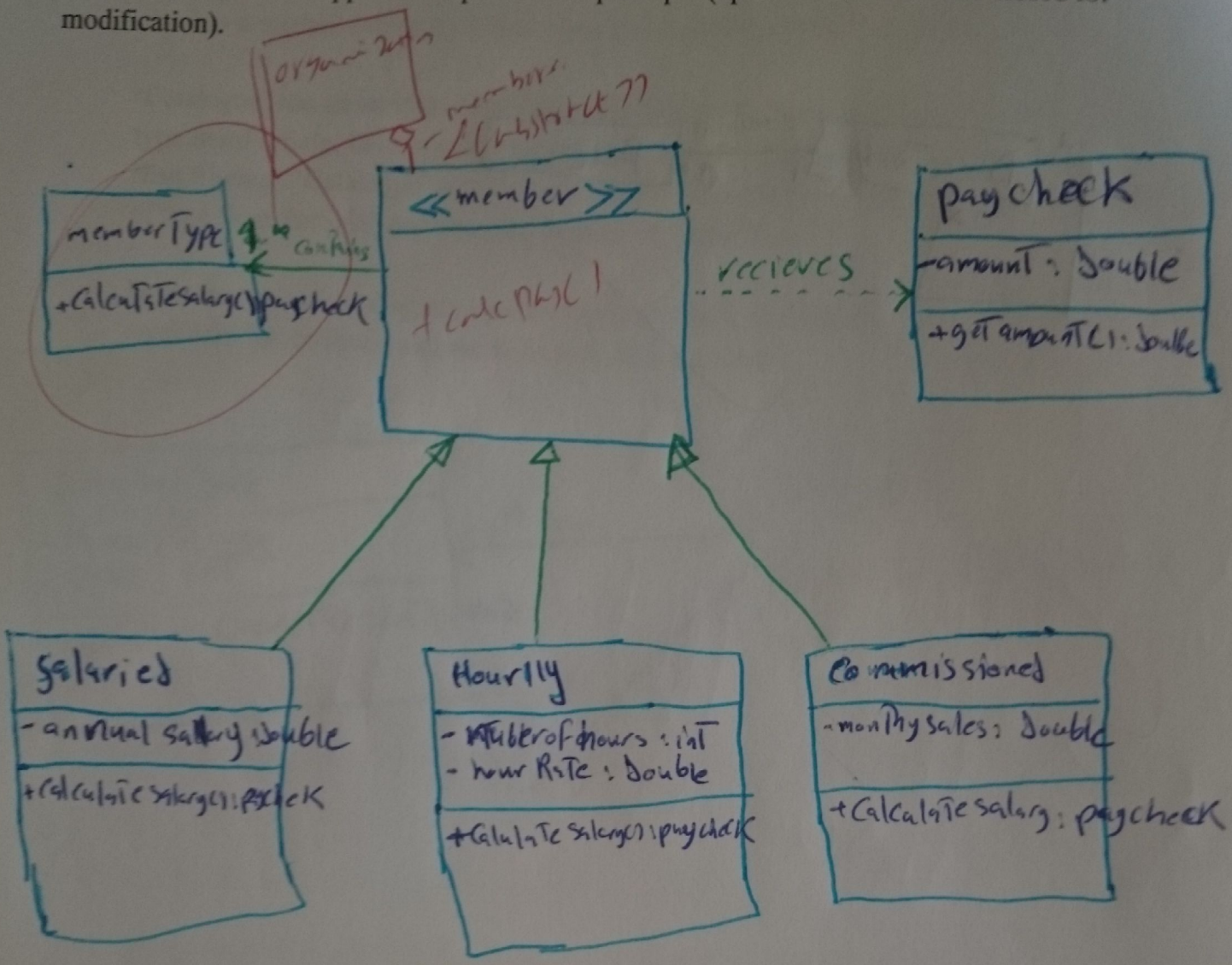
Commissioned members receive 10% of their monthly sales.

Your design should have the following:

a) Design an overall class diagram with proper relationship(IS-A, Has-A) and multiplicity.

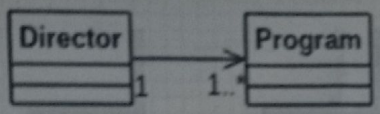
b) In each class specify the necessary attributes and behavior.

c) Your system must support the Open Closed principle (open for extension and closed for modification).



V. Coding Part – To test your Programming Skill

20. Write the code for the given UML diagrams to correctly model the relationship between Director and Program with proper constructor. Director has name property and Program has programId. [5 Points]




```
public class Director {

    ArrayList<Program> programs;
    String name;
    //(Constructor)
    public Director(String name) {
        this.name = name;
    }
}
```

```
public class Program {
    int programID;

    public Program(int programID) {
        this.programID = programID;
    }
}
```

```
public Director(String name, Program program) {
    this.name = name;
    programs = new ArrayList<Program>();
    programs.add(program);
}
```

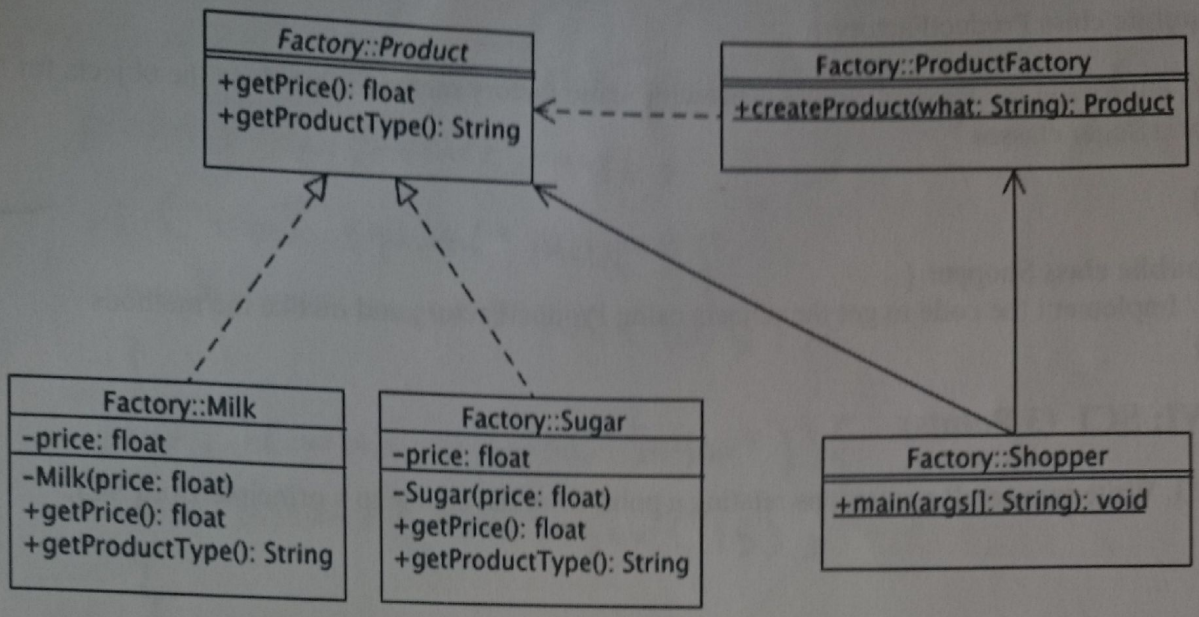
21. Complete the given code to guarantee that only one instance can be created for this class. [5 Points]

```
class SingletonClass {
        String key;
        private static SingletonClass singletonClass = new SingletonClass();
        private SingletonClass() {
            key = "K135J89L35";
        }
        public static SingletonClass getSingletonObject(){
            if (singletonclass == null) {
            System.out.println("unsuccessfull To produce TheKey...");
            return null
            }
            else {
            System.out.println("your keyToactivAvasT is " + key);
        SingletonClass Temp = singletonclass;
        singletonclass = null;
            return Temp;
        } }
    private void getKey() {
            System.out.println("Your key to activate Avast antivirus is:" + key);
        }
}
public class SingletonObjectDemo {
public static void main(String[] args) {

// First object needs to produce the key
SingletonClass obj = SingletonClass.getSingletonObject();

// Second object is unsuccessful to produce the key
SingletonClass obj1 = SingletonClass.getSingletonObject();
 }
}
```

Your program expected to produce the following output.

Key generated Successfully
Your key to activate Avast antivirus is:K135J89L35
Unsuccessful to produce the key....

22. Write the Implementation for the given Class diagram. [10 Points]




```
public abstract class Product {

    abstract public double
getPrice();

    public String getProductType()
{
        return "Unknown Product";
    }

}
public class Milk extends Product
{

    private double price;

    protected Milk(double price) {
        this.price = price;
    }

    public double getPrice() {
        return price;
    }

    public String getProductType()
{
        return "Milk";
    }
}
```

```
public class Sugar extends Product
{

    private double price;


    public Sugar(double price1) {
          this.price= price;
     }

     public double getPrice() {
        return price;
    }

    public String getProductType()
{
        return "Sugar";
    }

}
```

Do the implementation for the following class.

```
public class ProductFactory {

// Implement the ProductFactory class with Static factory method to produce the objects for Milk
and Sugar classes

}

public class Shopper {
// Implement the code to get the objects using ProductFactory and invoke the methods
}
```

VI: SCI (3 Points)

23. Write two small paragraphs relating a point from the course to a principle from SCI.