Student ID__________________Student Name____________________________________

Advanced Software Development DE Practice Final

PRIVATE AND CONFIDENTIAL

1. **Allotted exam duration is 2 hours.**
2. **Closed book/notes.**
3. **No personal items including electronic devices (cell phones, computers, calculators, PDAs).**
4. **Cell phones must be turned in to your proctor before beginning exam.**
5. **No additional papers are allowed. Sufficient blank paper is included in the exam packet.**
6. **Exams are copyrighted and may not be copied or transferred.**
7. **Restroom and other personal breaks are not permitted.**
8. **Total exam including questions and scratch paper must be returned to the proctor.**

5 blank pages are provided for writing the solutions and/or scratch paper. All 5 pages must be handed in with the exam

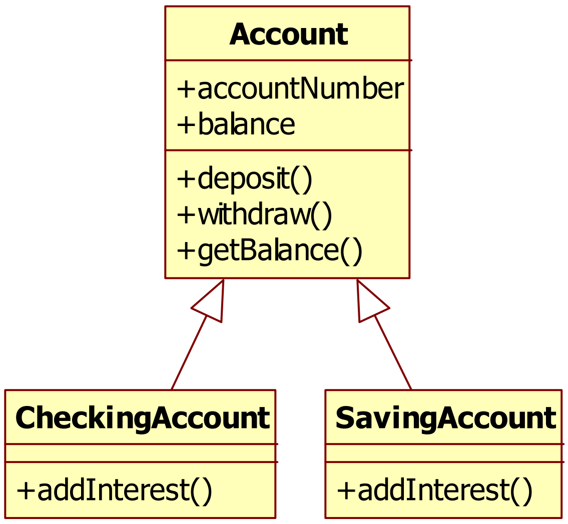
BE VERY CAREFUL WITH THE GIVEN 2 HOURS AND USE YOUR TIME WISELY. THE ALLOTED TIME IS GIVEN FOR EVERY QUESTION.

Write your name and student id at the top of this page.

Question 1 [20 points] {20 minutes}

Suppose we need to design a bankaccount framework that support saving accounts and checking accounts. One of the requirements is that we can add interest to every account. The formula to compute the interest is different for saving and checking accounts.

One colleague comes with the following design:



Applications that use this framework need to support all kind of other accounts (children accounts, business accounts, credit card accounts, etc). It is your job to design this framework.

a. Draw the class diagram of your design. This can be similar as the given design, or maybe you can make a better design. Your design should offer the same business functionality as the given design.
b. Explain the advantage(s) of your design compared to the given design

Question 2 [10 points] {10 minutes}

Consider the following application that uses a dynamic proxy:

```
public interface IVehicle {
       void start();
}


public class Car implements IVehicle {
      private String name ="Herbie";

      public void start() {
            System.out.println("Car " + name + " started");
      }

}


public class Logger implements InvocationHandler {
      private Object v;

      public Logger(Object v) {
            this.v = v;
      }

      public Object invoke(Object proxy, Method m, Object[] args)
                  throws Throwable {
            System.out.println("Logger: " + m.getName());
            Object object= m.invoke(v, args);
            return object;
      }
}


public class Notifier implements InvocationHandler {
      private Object v;

      public Notifier(Object v) {
            this.v = v;
      }

      public Object invoke(Object proxy, Method m, Object[] args)
                  throws Throwable {
            System.out.println("Notifier: " + m.getName());
            Object object= m.invoke(v, args);
            System.out.println("Notifier: " + m.getName());
            return object;
      }
}
```

What is the output written to the console when we run the following application?

```
public class Application {
	public static void main(String[] args) {
		IVehicle c = new Car();
		ClassLoader cl = IVehicle.class.getClassLoader();
		IVehicle v1 = (IVehicle) Proxy.newProxyInstance(cl, new Class[]
			{ IVehicle.class }, new Logger(c));
		IVehicle v2 = (IVehicle) Proxy.newProxyInstance(cl, new Class[]
			{ IVehicle.class }, new Notifier(v1));
		v2.start();
	}
}
```

Your answer:

Question 3 [15 points] {20 minutes}

a) Which pattern that we studied is used by the Spring framework to implement AOP? Only give the name of the pattern.

Your answer:

b) Explain why Spring beans need to implement an interface when we want to make use of the power of dependency injection.

Your answer:

c) Explain why Spring beans need to implement an interface when we want to make use of AOP in Spring.

Your answer:

d) Give 2 advantages and 2 disadvantages of Spring AOP

Your answer:

Question 4 [50 points] {70 minutes}

Suppose you have to design an expense tracking framework. The framework allows us to keep track of all our financial transactions, both income and expenses. The expense tracking framework has the following requirements:

- We should be able to add new financial transactions
- We should be able to remove existing financial transactions
- We should be able to update existing financial transactions
- For every transaction, we want to record its name, data, amount, category name and transaction type (income or expense)
- Transactions can be placed in categories, for example groceries, fuel, insurance, car, medicine, etc.
- We should be able to add a new category
- We should be able to remove an existing category
- We should be able to update an existing category
- For every transaction, we want to record its name and a short description
- All categories and transactions should be stored in a database
- We have a GUI page that allows us to add, remove or update categories
- We have a GUI page that allows us to add, remove or update financial transactions
- The application has different GUI pages (views) that show an overview of our incomes and expenses. One view shows all incomes and expenses from one month in a nice list. Another view shows all incomes and expenses from one month in a pie chart. It should be very easy to add new views like all incomes and expenses from one week or from one day.
- It is possible to show multiple views at the same time. So we might show both the list view of the monthly transactions and the pie chart view of the monthly transactions at the same time.
- The framework also supports the functionality to export the transactions either to a text file, or a file in CVS format (used for Excel).
  It should be easy to add new functionality to export to other file formats like for example PDF.

a) Draw the **UML class diagram** of the expense tracking framework. Show clearly the necessary objects, attributes, methods and associations. Use the principles and best practices we learned in this class. The evaluation criteria for this question will be how well you have designed the application.
b) Draw the **UML sequence diagram** of your design that shows what happens when a user adds a new transaction.