

**Ministry of Higher Education and Scientific Research**

**Salahaddin University / Erbil  
College of Engineering  
Dept. of Software Engineering**

**2<sup>nd</sup> Term Exam  
2011-2012**

**Subject: Systems Analysis and Design  
Time: 90 Minutes  
Lecturer: Amanj Sherwany  
Date: 29 Apr. 2012**

*The highest obtainable mark is 100, the minimum passing mark is 50*

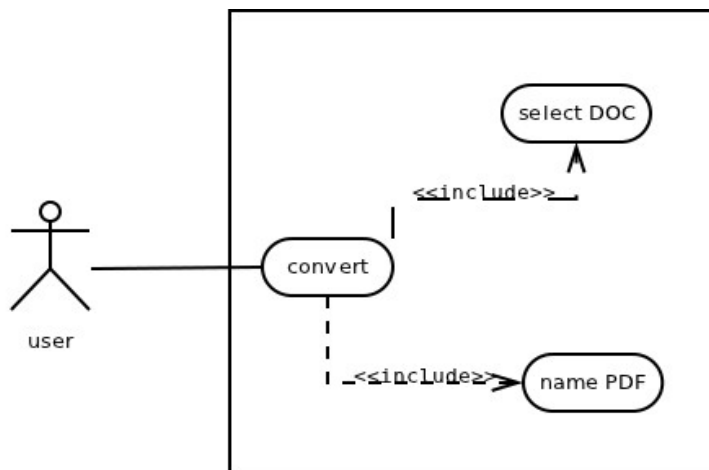
**Q1: (40 points)**

**A (20 points):**

You are designing a program to convert DOC files to PDF files, draw the use-case diagram of the following scenario:

A user selects a DOC file to be converted, and enters the name of the PDF file, then the user clicks on the convert button and the program starts converting the DOC file to PDF.

*Answer*



**B (20 points):**

Draw the class diagram for the following classes:

```
class CPU{
    private int speed;
}
class RAM{}
class Screen{
    public int size;
}
class TouchScreen extends Screen{}
```

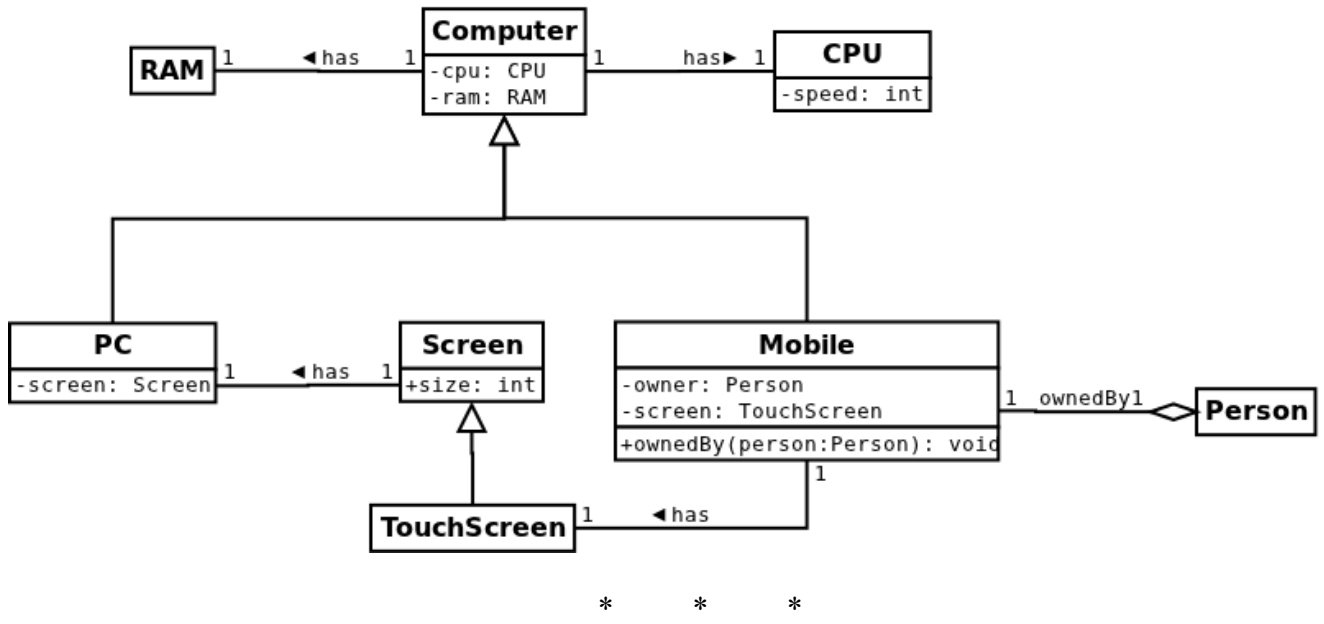
```
class Computer{
    private CPU cpu;
    private RAM ram;
}
class PC extends Computer{
    private Screen screen;
```

```

}
class Mobile extends Computer{
    private Person owner;
    private TouchScreen screen;
    void ownedBy(Person person){
}
}
class Person{}
    owner = person;
}

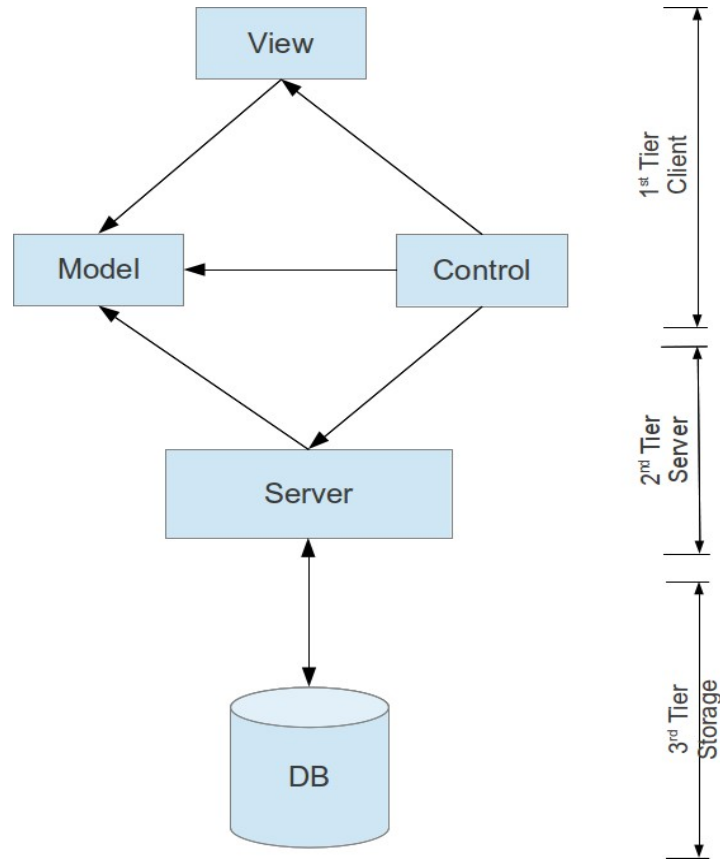
```

*Answer*



**Q2: (15 points)**

Design a possible architecture for the library system of our department, give details for each possible tier (follow 3-tier pattern).



### Q3: (20 points)

What is the expected output of the following program if the binding is:

- 1- Dynamic
- 2- Static

```
public class Foo{
    int getRandom(){
        return 6;
    }
}
public class Bar extends Foo{
    int getRandom(){
        return 7;
    }
}
Foo f1 = new Bar();
Foo f2 = new Foo();
System.out.println(f1.getRandom());
System.out.println(f2.getRandom());
```

**Answer**

**1- Dynamic:**

7  
6

**2- Static:**

6  
6

\* \* \*

### Q4: (25 points)

Discuss how the following code can be improved using *Template Design Pattern*:

```
class MyListener implements ActionListener {
    public void actionPerformed(ActionEvent ae) {
        try {
            firstMethod();
        } catch (SomeException se) {
            se.printStackTrace();
            messageTextField.setText(se.getMessage());
        }
    }
}
class MyOtherListener implements ActionListener {
    public void actionPerformed(ActionEvent ae) {
        try {
            anotherMethod();
        } catch (SomeException se) {
            se.printStackTrace();
        }
    }
}
```

```
        messageTextField.setText (se.getMessage ());
    }
}
}
```

### ***Answer***

```
abstract class TemplateListener implements ActionListener {
    public void actionPerformed(ActionEvent ae) {
        try {
            action();
        } catch (SomeException se) {
            se.printStackTrace();
            messageTextField.setText (se.getMessage ());
        }
    }
    public abstract void action();
}
class MyListener extends TemplateListener{
    public void action(){
        firstMethod();
    }
}
class MyOtherListener extends TemplateListener{
    public void action(){
        anotherMethod();
    }
}
```

**Good Luck**