

NAME:
SURNAME:
NIA:
GROUP:

2nd Part: Exercises (5 points out of 10)

Time: 45 minutes
Max score: 5 points
Date: March 14th, 2013

Instructions:

- No books or any resource allowed.
- Do not forget to write down your name, NIA and group in every sheet of paper.

Section 1 (1.5 points)

1.1 Define an **interface** called `Repairable` that contains a `change` method that receives an object of the `Replacement` type called `newItem` as input parameter and returns a `true` boolean if the repair has been successful or `false` otherwise.

```
public interface Repairable{
    public boolean change(Replacement newItem);
}
```

1.2. Create a printer **abstract class** that models a printer that has a boolean attribute called `duplex` to indicate whether the printer can print in duplex (on both sides) or not and a `print` **abstract method** that gets a `Document` object as parameter and returns a boolean equal to `true` if the document could be printed correctly or `false` in the case of any problem. The attribute must have the appropriate modifier keyword to make the attribute accessible only to child classes. You must also implement the access methods (`get/set`) and a constructor to create a printer to allow to assign a value to the `duplex` attribute at creation time.

```
public abstract class Printer {
    boolean duplex;

    public Printer(boolean duplex) {
        this.duplex = duplex;
    }

    public boolean getDuplex() {
        return duplex;
    }

    public void setDuplex(boolean duplex) {
        this.duplex = duplex;
    }

    public abstract boolean print(Document document);
}
```



Section 2 (1.5 points)

2.1. Create an `InkPrinter` class that inherits from `Printer` and models an ink printer that holds an attribute of `Cartridge` type called `colorCartridge`. The class must also contain a `change` method that receives as parameter an object of `Cartridge` type called `replacement` and returns a boolean. The invocation to the `change` method must print a “changing the cartridge” message and the invocation to the `print` method must print a “printing from the ink printer” message. You can assume for the exam that there is no chance of failure and both methods always return `true`. Create a constructor that invokes that of the parent class, but you do not need to create the access methods for the new attribute.

```
public class InkPrinter extends Printer{
    private Cartridge colorCartridge;
    public InkPrinter(boolean duplex, Cartridge colorCartridge){
        super(duplex);
        this.colorCartridge = colorCartridge;
    }

    public boolean print(Document document){
        System.out.println("Imprimiendo desde impresora tinta");
        return true;
    }

    public boolean change(Cartridge cartridge){
        System.out.println("Cambiando el Cartridge");
        return true;
    }
}
```

2.2 Create a `LaserPrinter` class that inherits from `Printer` and has a private attribute of the `Replacement` type, a `Toner` attribute called `colorCartridge` and a `change` method that gets a `Toner` object and returns a boolean. The invocation to `change` method must print a “changing the toner” message and the invocation to the `print` method must print a “printing from the laser printer” message. Again, you can assume for the exam that there is no chance of failure and both methods always return `true`. Create a constructor that invokes that of the parent class, but you do not need to create the access methods for the new attribute.



```
public class LaserPrinter extends Printer{
    private Toner colorToner;

    public LaserPrinter(boolean duplex, Toner colorToner){
        super(duplex);
        this.colorToner = colorToner;
    }

    public boolean print(Document document){
        System.out.println("Imprimiendo desde impresora Laser");
        return true;
    }

    public boolean change(Toner toner){
        System.out.println("Cambiano el Toner");
        return true;
    }
}
```

Section 3 (1 point)

Write an Application class that has a main method and carries out the following operations:

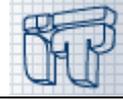
- Create an array called `stock`. Decide yourself the data type for the array to allow to store both ink and laser printers and invoke the `print` method on them, no matter what the type of printer the object is. To implement the main method you can assume that previously two objects have been created (within the main): an ink cartridge called `cartridge1` and a toner called `toner1`;
- Add two printers to the array, each of a different type.
- Traverse the array calling the `print` method for each printer stored in the array.

```
public class Test {
    public static void main(String[] args) {
        Printer[] stock = new Printer[2];
        stock[0] = new InkPrinter(true, new Cartridge());
        stock[1] = new LaserPrinter(true, new Toner());

        for (int i = 0; i < stock.length; i++) {
            stock[i].print(new Document());
        }
    }
}
```

Section 4 (1 point)

Now assume that a new abstract class called `Replacement` from which `Toner` and `Cartridge` inherit is created and the `Printer` class is written to implement the `Repairable` interface. Write the new



declarations of the `Replacement`, `Toner`, `Cartridge` and `Printer` classes, and what changed should be included in the `InkPrinter` and `LaserPrinter` classes to allow that the `change` method of the `Repairable` interface (which allows to change the cartridge or toner), could be also invoked generically on the elements of the `stock` array, independently of the type of printer.

Declaration of the `Replacement` class (do no implement)

```
public class Replacement {...}
```

Declaration of the `Toner` class (do no implement)

```
public class Toner extends Replacement {...}
```

Declaration of the `Cartridge` class (do no implement)

```
public class Cartridge extends Replacement {...}
```

Declaration of the `Printer` class (do no implement)

```
public abstract class Printer implements Repairable {...}
```

Change in the `InkPrinter` class (do no implement)

```
public boolean change (Replacement newItem)
```

Change in the `LaserPrinter` class (do no implement)

```
public boolean change (Replacement newItem)
```