Listagem A

```
class Numero{
      private int num;
      public static int quantos = 0;
      public Numero( int n){ num = n; ++quantos; }
      public String toString(){
            return "(" + num + "," + quantos +") ";
      }
}
public class Teste {
      public static void main( String[] args){
            System.out.print(" quantos=" + Numero.quantos);
               Numero [] a = new Numero[2];
               System.out.print(" quantos=" + Numero.quantos);
            Numero x = new Numero(10);
            for (int i=0 ; i < a.length ; i++)
                  a[i] = x;
            System.out.print("; a[1]=" + a[1] );
            a[1] = new Numero(11);
            System.out.print("; a[0]=" + a[0] );
      }
}
```

Listagem B

```
class Animal {
}
class Ave extends Animal {
    public boolean equals(Object obj) {
        return true;
    }
}
class Canario extends Ave {
    public boolean equals(Canario obj) {
        return false;
    }
}
public class Teste3 {
    static public void main(String[] args) {
        Animal animal1 = new Animal(), animal2 = new Animal(),
                ave1 = new Ave(),  ave2 = new Ave(),
                canario1 = new Canario(),  canario2 = new Canario();
        System.out.print("  " + (animal1 == animal2));
        System.out.print("  " + animal1.equals(animal2));
        System.out.print("  " + ave1.equals(ave2));
        System.out.print("  " + ave1.equals("string"));
        System.out.print("  " + canario1.equals(canario2));
        System.out.print("  " + ((Canario) canario1).equals((Canario) canario2));
    }
}
```

Listagem C

```
class Teste2 {
    static void f() throws Exception {
        try {
            System.out.print(" f1");
            int[] x = new int[4];
            x[10] = 9;
            System.out.print(" f2");
        } catch (ArrayIndexOutOfBoundsException e) {
            System.out.print(" f3");
            throw new IOException();
        } catch (Exception e) {
            System.out.print(" f4");
        }finally {
            System.out.print(" f5");
        }
        System.out.print(" f6");
    }
    public static void main(String[] args) {
        try {
            System.out.print(" m1");
            f();
            System.out.print(" m2");
        } catch (ArrayIndexOutOfBoundsException e) {
            System.out.print(" m3");
        } catch (Exception e) {
            System.out.print(" m4");
        }
        System.out.print(" m5");
    }
}
```

Listagem D

```java
interface ISerVivo{
    void alimentar();
}
interface IDeslocar{
    void deslocar();
}
interface IAnimal extends ISerVivo, IDeslocar{
    void respirar();
}
interface INadar extends IAnimal{
    void nadar();
}
abstract class AnimalAquatico implements IAnimal, INadar {
}
class Peixe extends AnimalAquatico {
    private String nome;
    public Peixe( String s){
        nome = s;
    }
    public void respirar(){}
    public void nadar(){}

    public void setNome(String s){
        nome = s;
    }
}
class Salmao extends Peixe {
    public Salmao(){
        setNome("Salmao");
    }
    public void alimentar(){}
    public void deslocar(){}
    public void nadar(){}
    public void passearCom( ISerVivo sv){
        deslocar();
        sv.deslocar();
    }
}
public class Teste4 {
    static public void main(String[] args) {
        ISerVivo  a =  new AnimalAquatico();
        ISerVivo  b =  new Salmao();
        IAnimal [] c =  new IAnimal[4];
        c[0] = b;
        c[0] = new Salmao();
        c[0].alimentar();
        c[0].respirar();
        c[0].deslocar();
        c[0].nadar();
        b.alimentar();
        b.respirar();
        b.deslocar();
        b.nadar();
    }
}
```

Listagem E

```
public class SelfCheckoutCashdesk {

    public static enum Enums {IDLE, PAY, WAIT_PAYMENT, START, WAIT_ITEM_TO_BE_SCANNED,
                            ITEM_SCANNED, ITEM_PUT_IN_BAG, TIMEOUT, BAG_WEIGHT_CHANGED,
                            SOLVED, ABORTED, WAIT_ITEM_TO_BE_IN_BAG,
                            WAIT_WEIGHT_TO_MATCH_SCANNED_ITEMS, WAIT_CASHIER_ACTION};

    private Enums situation;

    public SelfCheckoutCashdesk () {
        situation = Enums.IDLE;
    }

    public Enums getSituation() {
        return situation;
    }

    public void setSituation(Enums situation) {
        this.situation = situation;
    }

    public void processEvent(Enums event, Info info) {

        switch(situation){

            case IDLE:

                if(event == Enums.START){
                    situation = Enums.WAIT_ITEM_TO_BE_SCANNED;
                }
                break;

            case WAIT_ITEM_TO_BE_SCANNED:

                if(event == Enums.ITEM_SCANNED){
                    situation = Enums.WAIT_ITEM_TO_BE_IN_BAG;
                }else if(event == Enums.BAG_WEIGHT_CHANGED){
                    situation = Enums.WAIT_WEIGHT_TO_MATCH_SCANNED_ITEMS;
                }else if(event == Enums.PAY){
                    situation = Enums.WAIT_PAYMENT;
                }
                break;

            case WAIT_PAYMENT:

                if(event == Enums.PAY){
                    if(info.paymentSucceeded()){
                        situation = Enums.IDLE;
                    }
                }
                break;

            case WAIT_CASHIER_ACTION:

                if(event == Enums.SOLVED){
                    situation = Enums.WAIT_ITEM_TO_BE_SCANNED;
                }else if(event == Enums.ABORTED){
                    situation = Enums.IDLE;
                }
                break;

            case WAIT_WEIGHT_TO_MATCH_SCANNED_ITEMS:

                if(event == Enums.BAG_WEIGHT_CHANGED){
                    if(info.getBagWeight() == info.getScannedItemsWeight()){
                        situation = Enums.WAIT_ITEM_TO_BE_SCANNED;
                    }else{
                        situation = Enums.WAIT_CASHIER_ACTION;
                    }
                }else if(event == Enums.TIMEOUT){
                    situation = Enums.WAIT_CASHIER_ACTION;
                }
                break;

            case WAIT_ITEM_TO_BE_IN_BAG:

                if(event == Enums.BAG_WEIGHT_CHANGED){
                    if(info.getBagWeight() == info.getScannedItemsWeight()){
                        situation = Enums.WAIT_ITEM_TO_BE_SCANNED;
                    }else{
                        situation = Enums.WAIT_CASHIER_ACTION;
                    }
                }else if(event == Enums.TIMEOUT){
                    situation = Enums.WAIT_CASHIER_ACTION;
                }
                break;

            default:
                break;

        }
    }

}
```

Listagem F

```
public interface IView {
    void update(Observable o);
}

-------------------

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

public class Observable {

    private List<IView> views = new ArrayList<>();

    public void addView(IView view){
        views.add(view);
    }

    public boolean removeView(IView view){
        return views.remove(view);
    }

    public void notifyViews(){
        for(IView v:views){
            v.update(this);
        }
    }

}
```

Listagem G

```
public class CashDeskBasicView /* A */ {

    SelfCheckoutCashdesk cashdesk;

    public CashDeskBasicView(SelfCheckoutCashdesk cashdesk) {

        /* B */
        /* C */

    }

    /* D */

}
```

Listagem H

```
public class ProcessEventCommand implements ICommand {

    private /* A */; //command receiver
    private Enums event;
    private Info info;

    private Enums previousSituation;
    private boolean executed;

    public ProcessEventCommand(/* A */, Enums event, Info info){

        /* B */;
        this.event = event;
        this.info = info;
        executed = false;
    }

    public boolean execute() {

        if(executed)
            return false;

        /* C */;
        /* D */;
        return true;
    }

    public boolean undo() {

        if(!executed)
            return false;

        /* E */;
        return true;
    }

}
```

Listagem I

```
public abstract class Proposal implements Comparable<Proposal> {

    private String id;
    private int academicYear;
    private int semester;
    private String title;

    public Proposal(String id, int academicYear, int semestre, String title){
        this.id = id;
        this.academicYear = academicYear;
        this.semester = semestre;
        this.title = title;
    }

    public int hashCode() {
        /* A */
    }

    public boolean equals(Object obj) {
        /* B */
    }

    public String toString() {
        /* C */
    }

    public int compareTo(Proposal o) {
        /* D */
    }
}

---------------

public class Project extends Proposal {

   String researchGroup;

    public Project(String id, int academicYear, int semester, String title, String researchGroup){
        /* E */
    }

    public String toString(){
        /* F */
    }
}

---------------

public class Internship extends Proposal {

   String company;

    public Internship(String id, int academicYear, int semester, String title, String company){
        /* G */
    }

   @Override
    public String toString(){
        /* H */
    }
}
```

```
---------------

public class Factory {

    public static final int PROJECT = 0, INTERNSHIP = 1;

    public static Proposal createProposal(String id, int academicYear, int semester,
                                String title, String companyOrProject, int type){

        /* I */

        switch(type){
            /* J */
        }
    }
}

---------------

import java.util.*;

public class UseFactory {

    public static void main(String args[]){
        List <Proposal> proposals = new ArrayList<>();

        proposals.add(Factory.createProposal("P01", 1920, 1, "aaa", "X", Factory.INTERNSHIP));
        proposals.add(Factory.createProposal("P01", 1819, 2, "bbb", "RSD", Factory.PROJECT));
        proposals.add(Factory.createProposal("P01", 1819, 1, "ccc", "SI", Factory.PROJECT));
        proposals.add(Factory.createProposal("P02", 1718, 1, "ddd", "Y", Factory.INTERNSHIP));
        proposals.add(Factory.createProposal("P01", 1718, 1, "eee", "DA", Factory.PROJECT));

        for(Proposal p:proposals)
            System.out.println(p);

        System.out.println();

        Collections.sort(proposals);

        for(Proposal p:proposals)
            System.out.println(p);

    }

}
```

Expected output:

```
{Year: 1920 ; Semester: 1 ; id: P01} "aaa"(company: X)
{Year: 1819 ; Semester: 2 ; id: P01} "bbb"(research group: RSD)
{Year: 1819 ; Semester: 1 ; id: P01} "ccc"(research group: SI)
{Year: 1718 ; Semester: 1 ; id: P02} "ddd"(company: Y)
{Year: 1718 ; Semester: 1 ; id: P01} "eee"(research group: DA)

{Year: 1718 ; Semester: 1 ; id: P01} "eee"(research group: DA)
{Year: 1718 ; Semester: 1 ; id: P02} "ddd"(company: Y)
{Year: 1819 ; Semester: 1 ; id: P01} "ccc"(research group: SI)
{Year: 1819 ; Semester: 2 ; id: P01} "bbb"(research group: RSD)
{Year: 1920 ; Semester: 1 ; id: P01} "aaa"(company: X)
```