Recuperació Segon Parcial d'IIP - ETSInf
Data: 30 de gener de 2012. Duració: 2:30 hores.

1. 2.0 punts Els *nombres triangulars* es defineixen mitjançant la següent recurrència:

$$T_0 = 0$$
$$T_i = T_{i-1} + i \qquad i > 0$$

i els *nombres quadrats* es poden definir en funció dels triangulars com:

$$C_0 = 0$$
$$C_i = T_i + T_{i-1} \qquad i > 0$$

Es desitja un mètode estàtic iteratiu que donat un enter $n \geq 0$, escriga en la sortida els $n+1$ primers nombres triangulars i quadrats (en la i-èsima línia, els i-èsims nombres triangular i quadrat). Per exemple, per a $n = 4$, el mètode hauria d'escriure:

```
 0    0
 1    1
 3    4
 6    9
10   16
```

No es podrà usar cap altra operació aritmètica més que la suma.

Solució:

```
/** Escriu en la sortida els n+1 primers nombres triangulars i quadrats */
public static void nombresTC(int n) {
    int t = 0, c = 0, i = 0;
    System.out.println(t + "  " + c);
    while (i<n) {
        i++;
        int tAnt = t;
        t = tAnt + i;
        c = t + tAnt;
        System.out.println(t + "  " + c);
    }
}
```

2. 2.0 punts Escriure un mètode estàtic que donat un array de nombres enters i dos índex de l'array, inverteixca les components de l'array entre aquestes posicions inclusivament.

Per exemple, l'array {1,23,2,3,284,5,6} invertit entre els índex 1 i 4 ha de quedar {1,284,3,2,23,5,6}.

Solució:

```
/** Inverteix els elements d'a entre els índex i, j, éssent
  * ambdós índex del rang de l'array */
public static void invertPos(int[] a, int i, int j) {
    int aux;
    while (i<j) {
        aux = a[i];
        a[i] = a[j];
        a[j] = aux;
        i++;
        j--;
    }
}
```

3. 6.0 punts Un periòdic digital organitza les seues notícies mitjançant l'ús d'una classe `Noticia` que inclou la data (enter en el format *aaaammdd*, per exemple, *20120130* és *30 de gener de 2012*), l'hora (enter en el format *hhmm*, per exemple, *1005* és *10:05*), el text de la notícia (`String`), i el nombre de voltes que la notícia ha segut llegida (enter). El codi següent mostra part de la classe `Noticia` ja implementada:

```
/**La classe Noticia representa una notícia d'un periòdic digital.
 * @author (IIP - ETSINF - DSIC - UPV)
 * @version (Curs 2011-12)
 */
public class Noticia {
    private int data;      // Data en el format: aaaammdd
    private int hora;      // Hora en el format: hhmm
    private String text;   // Text de la notícia
    private int lectures;  // Nombre de vegades que s'ha llegit la notícia

    /** Crea una Noticia amb la data, hora i text donats. */
    public Noticia(int da, int h, String t) {
        data = da;
        hora = h;
        text = new String(t);
        lectures = 0;
    }

    /** Retorna la data en el format aaaammdd. */
    public int getData() { return data; }

    /** Retorna el nombre de lectures. */
    public int getLectures() { return lectures; }

    /** Incrementa en un el nombre de lectures. */
    public void incLectures() { lectures++; }

    /** Comprova si una data donada per d, m i a és igual
      * a la data de la Noticia actual. */
    public boolean igualData(int d, int m, int a) {
        return (a*10000 + m*100 + d) == data;
    }
```

```
    /** Retorna un String amb la informació de la Noticia. */
    public String toString() {
        String s = "";
        s+=data%100 + "/" + (data/100)%100+ "/" + (data/10000) + " - ";
        s+=(hora/100) + ":" + (hora%100) + "\n";
        s+=text + "\n";
        s+="Llegida " + lectures + " vegades\n";
        return s;
    }
}
```

El periòdic es composa d'un conjunt de notícies (se suposa que no hi ha més de 1000 notícies disponibles), i proporciona funcionalitats tals com obtindre la primera notícia d'un dia determinat, mostrar les notícies més populars (les més llegides), i esborrar les notícies velles.

Per a representar aquest conjunt de notícies es demana escriure una classe `Periodic` que ha d'incloure:

a) Atributs (0.5 punts):

- Un atribut `noticies`, array de `Noticia`, amb capacitat per a 1000 elements.
- Un atribut enter `numNoticies`, que ha de representar el nombre d'objectes `Noticia` emmagatzemats en cada moment en l'array.
  Nota: Els objectes `Noticia` s'han de trobar sempre contigus al començament de l'array i **no** es mantenen ordenats per data. Així doncs, l'atribut `numNoticies` indica també la primera posició lliure de l'array.

b) Un constructor (0.5 punts) que inicialitze els atributs, creant l'array de 1000 posicions, sense cap `Noticia` inicial.

c) Un mètode (0.5 punts) amb perfil:

```
public void inserir(Noticia n)
```

que donada certa `Noticia`, l'afegeixca a l'array, si cap. En cas contrari, no fa res.

d) Un mètode (1 punt) amb perfil:

```
public Noticia primeraNoticia(int d, int m, int a)
```

que retorne la primera notícia que es trobe en sentit ascendent en l'array amb la data donada per `d`, `m` i `a`. Si no existeix cap notícia d'eixa data o no existeixen notícies en l'array s'ha de retornar `null` per tal d'indicar-ho.

e) Un mètode (1.5 punts) amb perfil:

```
public void mesPopulars()
```

que mostre per pantalla tota la informació de les notícies més populars (és a dir, de les notícies més llegides). En cas de que n'hi haja vàries amb el mateix nombre de lectures, s'han de mostrar totes. En cas de que no hi hagen notícies, s'ha de mostrar un missatge que ho indique.

f) Un mètode (2 punts) amb perfil:

```
public void esborrarAnteriors(int d, int m, int a)
```

que elimine de l'array les notícies anteriors a la data donada per `d`, `m` i `a`. Després de l'esborrat, els elements en l'array han de permanèixer contigus (és a dir, no poden quedar forats).

Solució:

```
/**
 * @author (IIP - ETSINF - DSIC - UPV)
 * @version (Curs 2011-12)
 */
public class Periodic {
    public static final int NUM_NOT = 1000;
    private Noticia[] noticies;
    private int numNoticies;

    public Periodic() {
       noticies = new Noticia[NUM_NOT];
       numNoticies = 0;
    }

    public void inserir(Noticia n) {
        if (numNoticies!=NUM_NOT) noticies[numNoticies++] = n;
    }

    public Noticia primeraNoticia(int d, int m, int a) {
        int i = 0;
        while(i<numNoticies && !noticies[i].igualData(d,m,a)) i++;
        if (i<numNoticies) return noticies[i];
        else return null;
    }

    public void mesPopulars() {
        if (numNoticies==0) System.out.println("Sense notícies");
        else {
               Noticia mesP = noticies[0];
               int indMesP=0;
               for (int i=1; i<numNoticies; i++)
                  if (noticies[i].getLectures()>mesP.getLectures()) {
                      mesP = noticies[i];
                      indMesP=i;
                  }
               // mesP és la notícia més popular de menor índex,
               // si hi ha més notícies igual de populars que mesP
               // estaran a partir de l'índex indMesP
               System.out.println(mesP);
               for (int j=indMesP+1; j<numNoticies; j++)
                  if (noticies[j].getLectures()==masP.getLectures())
                      System.out.println(noticies[j]);
            }
    }

    public void esborrarAnteriors(int d, int m, int a) {
        int data = a*10000 + m*100 + d;
        int i = 0;
        while(i<numNoticies) {
            if (noticies[i].getData()<data) {
                noticies[i] = noticies[numNoticies-1];
                numNoticies--;
            }
            else i++;
        }
    }
}
```