Recuperació del Segon Parcial de IIP - ETSInf
Data: 28 de gener de 2014. Duració: 2:30 hores.

1. 6.5 punts Es disposa de la classe `Sensor` que representa un sensor de temperatura conectat a un dispositiu tèrmic de manera que, a més de mesurar la temperatura actual, pot també augmentar i reduir la temperatura (mesurada en graus centrígrads) del dispositiu. A més, és possible definir la temperatura ideal a la qual l'usuari desitja que estiga la zona on es troba el sensor. Tot seguit es mostra un resum de la seua documentació:

Constructor Summary

Constructors

Constructor and Description
`Sensor(java.lang.String nom, int tempIdeal)` Crea un Sensor amb un nom i una temperatura ideal.

Method Summary

Methods

Modifier and Type	Method and Description
`void`	`augmentaTempActual()` Augmenta la temperatura actual en una quantitat determinada (aleatòria) de graus.
`boolean`	`equals(java.lang.Object o)` Comprova si el Sensor en curs es igual a un altre donat; és a dir, si coincideixen en el nom.
`java.lang.String`	`getNom()`
`double`	`getTempActual()`
`double`	`getTempIdeal()`
`void`	`redueixTempActual()` Redueix la temperatura actual en una quantitat determinada (aleatòria) de graus.
`java.lang.String`	`toString()` Torna un String amb la informació del Sensor en curs.

Es demana: implementar la classe `CasaDomotica` què, com indica el seu nom, representa una casa domòtica en la qual poden haver-hi múltiples sensors ubicats a diferents llocs. Es representa mitjançant els components (atributs i mètodes) que s'indiquen a continuació.

a) (0.5 punts) Atributs:
   - `MAX_SENSORS`, constant que representa el nombre màxim de sensors d'una casa domòtica: 100.
   - `nombreSensors`, un enter en l'interval `[0..MAX_SENSORS]` que representa el nombre de sensors que té en cada moment la casa domòtica.
   - `sensors`, un array amb tipus base `Sensor`, de capacitat `MAX_SENSORS` i on els sensors s'emmagatzemen seqüencialment, en posicions consecutives des de la `0` fins a la `nombreSensors-1`.
   - `maxGrausDesviacio`, un valor decimal que representa la màxima desviació en graus que l'usuari desitja per als sensors de la casa domòtica.
   - `nombreSensorsAmbDesviacio`, que representa el nombre de sensors de la casa domòtica que tenen actualment una desviació en graus, en valor absolut, superior a `maxGrausDesviacio`.

b) (0.5 punts) Un constructor general que, donat un valor de màxima desviació en graus, crea una casa domòtica buida, amb 0 sensors, i el valor de màxima desviació en graus indicat.

c) (1 punt) Un mètode amb perfil: `private boolean existeixSensor(Sensor s)`
que, donat un `Sensor s`, indica si existeix o no aquest `Sensor` en la casa domòtica.

d) (0.5 punts) Un mètode amb perfil: `private boolean sensorAmbDesviacio(Sensor s)`
que, donat un `Sensor s`, torna `true` si el `Sensor` presenta una desviació de la seua temperatura actual front a la seua temperatura ideal superior a la màxima desviació en graus definida per a la casa domòtica. En cas contrari torna `false`.

e) (1 punt) Un mètode amb perfil: `public int registraSensor(Sensor s)`
que torna la posició en la que s'ha afegit el `Sensor s` donat en l'array de la casa domòtica. Si `s` no cap o ja està en l'array, el mètode torna `-1` per advertir que no s'ha pogut afegir a l'array. Cal fer ús del mètode privat `existeixSensor(Sensor)` per cercar el `Sensor`. També cal fer ús del mètode privat `sensorAmbDesviacio(Sensor)` per actualitzar, si cal, l'atribut `nombreSensorsAmbDesviacio`.

f) (1 punt) Un mètode amb perfil: `public Sensor[] sensorsAmbDesviacio()`
que torna un array de `Sensor` amb els sensors que presenten desviació de la seua temperatura actual front a la seua temperatura ideal, considerant la màxima desviació en graus definida per a la casa domòtica. La longitud d'aquest array serà igual al nombre de sensors que hi presenten desviació, o 0 si no hi ha cap. Cal fer ús del mètode `sensorAmbDesviacio(Sensor)`.

g) (1 punt) Un mètode amb perfil: `public void ajustaTemperatura()`
que regula la temperatura de tots els `Sensor` que presenten desviació en la seua temperatura actual front a la seua temperatura ideal d'acord amb la desviació definida per a la casa domòtica. La regulació es farà per a cada `Sensor` que presente desviació, s'han d'utilitzar els corresponents mètodes de la classe `Sensor` per ajustar la temperatura actual, fins que no hi presente desviació.

h) (1 punt) Un mètode amb perfil: `public Sensor sensorAmbMajorTemperatura()`
que torna el `Sensor` que tinga la major temperatura actual al moment d'invocar aquest mètode, o `null` si la casa domòtica no té cap `Sensor` registrat.

Solució:

CasaDomotica.java

```
/**
 * Representa una CasaDomotica, que inclou un array de sensors
 * que registren la temperatura de certes zones de la casa.
 */
public class CasaDomotica {
    private Sensor[] sensors;
    private int nombreSensors;
    private int nombreSensorsAmbDesviacio;
    private double maxGrausDesviacio;
    public static final int MAX_SENSORS = 100;

    public CasaDomotica(double maxGrausDesviacio) {
      sensors = new Sensor[MAX_SENSORS];
      nombreSensors = nombreSensorsAmbDesviacio = 0;
      this.maxGrausDesviacio = maxGrausDesviacio;
    }

    private boolean existeixSensor(Sensor s) {
      int i;
      for (i=0; i<nombreSensors && !sensors[i].equals(s); i++);
      return i<nombreSensors;
    }

    private boolean sensorAmbDesviacio(Sensor s) {
      return Math.abs(s.getTempActual() - s.getTempIdeal()) > maxGrausDesviacio;
    }

    public int registraSensor(Sensor s) {
      int res = -1;
      if (!existeixSensor(s) && nombreSensors<MAX_SENSORS) {
        res = nombreSensors; sensors[nombreSensors++] = s;
        if (sensorAmbDesviacio(s)) nombreSensorsAmbDesviacio++;
      }
      return res;
    }
```

```
    public Sensor[] sensorsAmbDesviacio() {
      Sensor[] aux = new Sensor[nombreSensorsAmbDesviacio];
      int k = 0;
      for (int i=0; i<nombreSensors; i++)
         if (sensorAmbDesviacio(sensors[i])) {
            aux[k] = sensors[i];
            k++;
         }
      return aux;
    }

    public void ajustaTemperatura() {
      for (int i=0; i<nombreSensors; i++) {
         Sensor s = sensors[i];
         while (sensorAmbDesviacio(s))
           if (s.getTempActual() > s.getTempIdeal()) s.redueixTempActual();
           else s.augmentaTempActual();
      }
      nombreSensorsAmbDesviacio = 0;
    }

    public Sensor sensorAmbMajorTemperatura() {
      Sensor max = sensors[0];
      for (int i=0; i<nombreSensors; i++)
         if (sensors[i].getTempActual() > max.getTempActual()) max = sensors[i];
      return max;
    }
}
```
CasaDomotica.java

2. 1.5 punts Donat un enter h dins del rang [0..23], es demana un mètode públic i estàtic que escriga en l'eixida estàndard hora a hora i minut a minut, totes les representacions horàries en el format de cinc caràcters `hh:mm`, des de la `00:00` fins l'últim minut de l'hora h. Cada minut s'ha de mostrar en una línia diferent. Per exemple, si $h = 13$, s'ha de mostrar el següent:

```
00:00
00:01
00:02
...
00:59
01:00
01:01
01:02
...
01:59
...
13:00
13:01
13:02
...
13:59
```

Nota: Per motius d'espai en aquest enunciat, l'exemple anterior s'ha abreujat substituint per punts suspensius bona part de les representacions que el mètode ha de mostrar.

Solució:

Primera versió:

```
/** h pertany al rang [0..23] */
public static void displayHores(int h) {
   for (int i=0; i<=h; i++) {
      String display = i<10 ? "0"+i+":" : i+":";
      for (int j=0; j<=9; j++) System.out.println(display+"0"+j);
      for (int j=10; j<=59; j++) System.out.println(display+j);
   }
}
```

Segona versió:

```
/** h pertany al rang [0..23] */
public static void displayHores(int h) {
    for (int i=0; i<=h; i++)
        for (int j=0; j<60; j++)
            System.out.printf("%02d:%02d\n", i, j);
}
```

3. 2 punts **Es demana:**

a) Indicar què mostra per pantalla el següent codi i explicar què fa en dues o tres línies com a molt.

b) Indicar què passaria si la condició del bucle intern al mètode `funcio(int[][])` fóra `j<m.length`.

```
public class Traza {
    public static void main(String[] args ) {
        int[][] a = { { 1, 2, 2 }, { 0, 1, 2 }, { 0, 0, 1 } };
        funcio( a );
        for (int i=0; i<a.length; i++) {
            for (int j=0; j<a.length; j++)
                System.out.printf( " %3d ", a[i][j] );
            System.out.println();
        }
    }
    public static void funcio(int[][] m) {
        for (int i=0; i<m.length; i++)
            for (int j=0; j<i; j++)
                intercanvia( m, i, j );
    }
    public static void intercanvia(int[][] m, int a, int b) {
        int aux = m[a][b]; m[a][b] = m[b][a]; m[b][a] = aux;
    }
}
```

Solució:

a) Per pantalla es mostra:

```
1    0    0
2    1    0
2    2    1
```

Calcula la transposada de la matriu `a`.

b) Deixaria la matriu com estava originalment, doncs fa l'operació de transposar dues vegades.