PRG - ETSInf. TEORIA. Curs 2014-15. Parcial 2.
11 de juny de 2015. Duració: 2 hores.

1. [2.5 punts] **Es demana:** implementar un mètode estàtic tal que:

   - Ha de rebre com argument un `String`, que es considera que conté la ruta i nom d'un fitxer de text.
   - S'haurà de suposar que cada línia del fitxer rebut conté una representació vàlida d'un nombre enter o alguna cosa que no ho siga. Es desconeix quantes línies hi ha al fitxer.
   - Haurà de propagar l'excepció `FileNotFoundException` si no puguera obrir el fitxer.
   - Haurà de sumar tots els números enters continguts al fitxer i tornar aquesta suma.
   - Si es llegeix una representació vàlida d'un nombre enter, aquest nombre se sumarà al resultat a retornar.
   - Si es llegeix una representació no vàlida d'un nombre enter, l'excepció `InputMismatchException` produïda s'haurà de capturar, mostrant a l'eixida d'error un missatge que incloga el nom de l'excepció i el valor d'aquesta representació no vàlida. Aquesta circumstància no ha d'impedir que continue la lectura del fitxer.

Solució:

```
public static int sumar(String f) throws FileNotFoundException {
    int suma = 0;
    Scanner sc = new Scanner(new File(f));
    while (sc.hasNextLine())
        try {
            suma += sc.nextInt();
        } catch (InputMismatchException e) {
            System.err.println(e + "::" + sc.nextLine());
        }
    sc.close();
    return suma;
}
```

2. [3 punts] **Es demana:** afegir a la classe `CuaIntEnla` un mètode amb perfil:

```
public void colar(int x)
```

   tal que:

   - Cerque la primera ocurrència de l'element `x` dins de la cua i en cas d'èxit en la cerca, faça que aquest element es "cole" davant del tot i, per tant, es quede com el primer de la cua.
   - En cas de fracàs en la cerca, la cua es queda com estava.

   NOTA: Només es permet accedir als atributs de la classe, quedant terminantment prohibit l'accés als seus mètodes.

Solució:

```
/** Si x està en la cua, el fica el primer en la cua. */
public void colar(int x) {
    NodeInt aux = primer, ant = null;
    while (aux != null && aux.dada != x) {
        ant = aux;
        aux = aux.seguent;
    }
```

```
    if (aux != null && aux != primer) {
        ant.seguent = aux.seguent;
        aux.seguent = primer;
        primer = aux;
        if (aux == ultim) ultim = ant;
   }
}
```

3. 2 punts Considereu la classe `LlistaPIIntEnla`, amb tots els mètodes coneguts i, a més, el mètode seguent, que es suposa també implementat:

```
/** Torna true si n es troba en algun node de la llista,
 *  false en cas contrari. No modifica el PI.
 */
public boolean conte(int n)
```

Es demana: implementar un mètode estàtic (en una classe distinta de `LlistaPIIntEnla`) tal que:

- Reba com arguments dos objectes de la classe `LlistaPIIntEnla`, anomenats `a` i `b`.
- Ha d'inserir en la llista `a` només les dades emmagatzemades en la llista `b` que no es troben prèviament emmagatzemades en la llista `a`.
- La inserció en la llista `a` es farà davant de l'element assenyalat pel `PI`, mantenint-se la posició del `PI`.
- En la llista `b` es pot modificar la posició del seu `PI`.
- En la implementació, s'ha d'usar el mètode `conte`.

Solució:

```
public static void inserir_nous(LlistaPIIntEnla a, LlistaPIIntEnla b) {
    b.inici();
    while (!b.esFi()) {
        int i = b.recuperar();
        if (!a.conte(i)) a.inserir(i);
        b.seguent();
    }
}
```

4. 2.5 punts Donada una `PilaIntEnla p` i un enter `x`, **es demana:** escriure un mètode estàtic (en una classe distinta de `PilaIntEnla`), tal que:

- Calcule i torne el número d'aparicions de `x` en `p`.
- Ha de deixar la pila `p` en l'estat en què estava inicialment.

Solució:

```
public static int numAparicionsEnPila(PilaIntEnla p, int x) {
    int n = 0;
    if (!p.esBuida()) {
        int aux = p.desempilar();
        n = numAparicionsEnPila(p, x);
        if (aux == x) n++;
        p.empilar(aux);
```

```
    }
    return n;
}
```

Alternativament:

```
public static int numAparicionsEnPila(PilaIntEnla p, int x) {
    int[] aux = new int[p.talla()];
    int n = 0, i = 0;
    while(!p.esBuida()) {
        aux[i] = p.desempilar();
        if (aux[i] == x) n++;
        i++;
    }
    for (i = aux.length - 1; i >= 0; i--) p.empilar(aux[i]);
    return n;
}
```