Parcial 2 - Pràctiques - PRG - ETSInf - Curs 2013/14
10 de juny de 2014 - Duració: 1 hora i 15 minuts

1. 2.5 punts El mètode `llegirInt`, que figura a continuació, llig des d'un `Scanner` i torna un valor enter en l'interval `[lInf..lSup]`. Aquest mètode s'utilitza des del programa principal per tal de llegir un número de compte de 5 dígits (entre 10000 i 99999).

```
public static int llegirInt(Scanner t, String missatge, int lInf, int lSup) {
    System.out.print(missatge);
    int res = t.nextInt();
    return res;
}

public static void main(String[] args) {
    Scanner tec = new Scanner(System.in);
    int numC = llegirInt(tec,"Introdueix un número de compte (de 5 dígits): ",10000,99999);
    System.out.println("El número de compte llegit és: " + numC);
}
```

Es demana:

a) [1.5 punts]: Modificar el mètode `llegirInt` perquè, si el valor introduït no està en el rang `[lInf..lSup]`,
- **llance** l'excepció `IllegalArgumentException`, amb un missatge que indique que el valor llegit no està en aquest rang,
- i la **propague explícitament**.

Solució:

```
public static int llegirInt(Scanner t, String missatge, int lInf, int lSup)
                                                        throws IllegalArgumentException {
    System.out.print(missatge);
    int res = t.nextInt();
    if (res>lSup || res<lInf)
        throw new IllegalArgumentException(res + " no està en [" + lInf + ".." + lSup + "]");
    return res;
}
```

b) [1 punt]: Modificar també el mètode `main` perquè **capture** aquesta excepció, mostrant el missatge de la mateixa mitjançant el mètode `getMessage()`.

Solució:

```
public static void main(String[] args) {
    Scanner tec = new Scanner(System.in);
    try {
        int numC = llegirInt(tec,"Introdueix un número de compte (de 5 dígits): ",10000,99999);
        System.out.println("El número de compte llegit és: " + numC);
    } catch(IllegalArgumentException e) {
        System.out.println(e.getMessage());
    }
}
```

2. 2.5 punts **Es demana:** Escriure un mètode que, donat el nom d'un **fitxer de text** amb la informació dels comptes d'un banc, torne la suma dels saldos de tots els comptes. Cada línia del fitxer té dos elements d'informació, un número de compte de tipus `int` seguit d'un saldo de tipus `double`. La capçalera del mètode és la que figura a continuació. **No has de tractar cap excepció**, fixa't que es propaga qualsevol excepció que es puga produir.

```
public static double sumaSaldos(String nomFitx) throws Exception
```

Solució:

```
public static double sumaSaldos(String nomFitx) throws Exception {
    double suma = 0;
    Scanner ent = new Scanner(new File(nomFitx)).useLocale(Locale.US);
    while(ent.hasNext()) {
        int numCompte = ent.nextInt();
        double saldo = ent.nextDouble();
        suma += saldo;
    }
    ent.close();
    return suma;
}
```

3. [3 punts] Donades les estructures de dades `Concordanca` i `NodeCnc`, com les vistes en pràctiques, amb atributs segons les declaracions següents:

```
Concordanca                          NodeCnc
-----------                          -------
private NodeCnc prim;                String pal;
private int talla;                   CuaIntEnla numLins;
private boolean esOrd;               NodeCnc seguent;
private String separadors;
```

Es demana: Escriure un mètode dintre de la classe `Concordanca` amb perfil:

```
// PRECONDICIÓ: n >= 1
public boolean mesAparicionsN(int n)
```

que determine si existeix en el text amb el que s'ha construït la `Concordanca` una paraula que aparega **al menys** `n` vegades.

Solució:

```
// PRECONDICIÓ n >= 1
public boolean mesAparicionsN(int n) {
    NodeCnc aux = prim;
    while(aux!=null && aux.numLins.talla()<n)
        aux = aux.seguent;
    return aux!=null;
}
```

4. [1 punt] Els mètodes públics de la classe `CuaIntEnla` són el constructor `CuaIntEnla()`, els modificadors `encuar(int)` i `desencuar()`, i els consultors `primer()`, `esBuida()`, `talla()` i `toString()`.

Es demana: Enumerar quins d'aquests mètodes s'han fet servir en:

a) la classe `NodeCnc`

Solució: Els mètodes `CuaIntEnla()` i `encuar(int)`.

b) la classe `Concordanca`

Solució: Els mètodes `encuar(int)`, `talla()` i `toString()`.

5. [1 punt] **Es demana:** Justificar breument quin és el cost assimptòtic (constant, lineal, quadràtic, logarísmic, etc.) del mètode:

```
public boolean esOrdenada()
```

definit en la classe `Concordanca`, que torna si es tracta d'una `Concordanca` ordenada o no.

Solució: El mètode `esOrdenada()` té cost constant ja que és un mètode consultor que només executa un `return` de l'atribut `esOrd`.