PRG - ETSInf. TEORIA. Curs 2016-17. Parcial 1.
10 d'abril de 2017. Duració: 2 hores.

Nota: L'examen s'avalua sobre 10 punts, però el seu pes específic en la nota final de PRG és de **3 punts**.

1. $\boxed{\text{4 punts}}$ Donat un array `a` de `int` i un enter `x`, escriu un mètode **recursiu** que torne quants múltiples de `x` hi ha a l'array `a`.

Es demana:

a) (0.75 punts) Perfil del mètode, amb els paràmetres adequats per a resoldre recursivament el problema.

b) (1.25 punts) Cas base i cas general.

c) (1.50 punts) Implementació en Java.

d) (0.50 punts) Crida inicial perquè es realitze el càlcul sobre tot l'array.

Solució:

a) Una possible solució consisteix en definir un mètode amb el següent perfil:

```
/** Precondició: 0 <= pos */
public static int multiplesX(int[] a, int x, int pos)
```

de manera que torne quants múltiples de `x` hi ha a l'array `a[pos..a.length - 1]`, sent $0 \leq$ `pos`.

b)
- Cas base, `pos` $\geq$ `a.length`: Subarray buit. Torna 0.
- Cas general, `pos` $<$ `a.length`: Subarray d'un o més elements. Si `a[pos] % x == 0`, torna 1 més el número de múltiples de `x` en `a[pos + 1..a.length - 1]`; si no, torna el número de múltiples de `x` en `a[pos + 1..a.length - 1]`.

c)
```
/** Torna el número de múltiples de x en a[pos..a.length - 1].
 *  Precondició: 0 <= pos */
public static int multiplesX(int[] a, int x, int pos) {
    if (pos >= a.length) { return 0; }
    else if (a[pos] % x == 0) { return 1 + multiplesX(a, x, pos + 1); }
    else { return multiplesX(a, x, pos + 1); }
}
```

d) Per un array `a`, la crida `multiplesX(a, x, 0)` resol el problema enunciat.

2. $\boxed{\text{3 punts}}$ Donat un array de caràcters `a` i un caràcter `c` qualsevol, el següent mètode escriu en l'eixida estàndard, línia a línia, tots els prefixes de la seqüència de caràcters en `a`, de longitud `1` en endavant, que no acaben en el caràcter `c`.

```
public static void prefixes(char[] a, char c) {
    for (int i = 0; i < a.length; i++) {
        if (a[i] != c) {
            for (int j = 0; j <= i; j++) {
                System.out.print(a[j]);
            }
            System.out.println();
        }
    }
}
```

Per exemple, si `a = {'g', 't', 'a', 't', 'c'}`, els prefixes de longituds succesives són `g`, `gt`, `gta`, `gtat` i `gtatc`. Per a `a` i `c = 't'`, el mètode escriu:

```
g
gta
gtatc
```

Es demana:

a) (0.25 punts) Indicar quina és la grandària o talla del problema, així com l'expressió que la representa.

b) (0.75 punts) Indicar si existeixen diferents instàncies significatives per al cost temporal de l'algorisme i identificar-les si és el cas.

c) (1.50 punts) Triar una unitat de mesura per a l'estimació del cost (passos de programa, instrucció crítica) i d'acord amb ella obtenir una expressió matemàtica, el més precisa possible, del cost temporal del mètode, distingint el cost de les instàncies més significatives en cas d'haver-les.

d) (0.50 punts) Expressar el resultat anterior utilitzant notació asimptòtica.

Solució:

a) La talla del problema és el nombre d'elements de l'array `a` i l'expressió que la representa és `a.length`. D'ara endavant, anomenarem a aquest número n. Açò és, n = `a.length`.

b) Sí que existeixen diferents instàncies. El cas millor es dóna quan tots els caràcters de l'array `a` són el caràcter `c`. El cas pitjor es dóna quan tots són distints del caràcter `c`.

c) Triant com a unitat de mesura el pas de programa, es té:

- En el cas millor: $T^m(n) = 1 + \sum_{i=0}^{n-1} 1 = n + 1 \; p.p.$
- En el cas pitjor: $T^p(n) = 1 + \sum_{i=0}^{n-1}(1 + \sum_{j=0}^{i} 1) = 1 + \sum_{i=0}^{n-1}(2 + i) = 1 + 2n + \sum_{i=0}^{n-1} i = 1 + n + \dfrac{n(n+1)}{2} = 1 + \dfrac{3n}{2} + \dfrac{n^2}{2} \; p.p.$

Triant com a unitat de mesura la instrucció crítica i considerant com tal:

- la condició `a[i] != c` de la instrucció `if` (de cost unitari), en el cas millor es té: $T^m(n) = \sum_{i=0}^{n-1} 1 = n \; i.c.$
- la instrucció del cos del bucle intern `System.out.print(a[j])` (de cost unitari), en el cas pitjor es té: $T^p(n) = \sum_{i=0}^{n-1} \sum_{j=0}^{i} 1 = \sum_{i=0}^{n-1}(1 + i) = n + \sum_{i=0}^{n-1} i = \dfrac{n(n+1)}{2} = \dfrac{n^2}{2} + \dfrac{n}{2} \; i.c.$

d) En notació asimptòtica: $T^m(n) \in \Theta(n)$ i $T^p(n) \in \Theta(n^2)$. Per tant, $T(n) \in \Omega(n)$ i $T(n) \in O(n^2)$.

3. $\boxed{\text{3 punts}}$ El següent mètode determina si, donat un nombre enter no negatiu `num`, el seu literal pot estar expressat en una base determinada `b` ($2 \leq$ `b` ≤ 10), comprovant que tots els dígits del nombre tenen un valor estrictament menor que la base `b`. Per exemple, el nombre 453123 pot representar un valor en base 6, 7, 8, 9 i 10 ja que tots els seus dígits són estrictament inferiors als valors d'aquestes possibles bases.

```
/** Precondició: 2 <= b <= 10 i num >= 0 */
public static boolean basePossible(int num, int b) {
    if (num == 0) { return true; }
    else {
        int ultDig = num % 10;
        if (ultDig < b) { return basePossible(num / 10, b); }
        else { return false; }
    }
}
```

Es demana:

a) (0.25 punts) Indicar quina és la grandària o talla del problema, així com l'expressió que la representa.

b) (0.75 punts) Indicar si existeixen diferents instàncies significatives per al cost temporal de l'algorisme i identificar-les si és el cas.

c) (1.50 punts) Escriure l'equació de recurrència del cost temporal en funció de la talla per a cada un dels casos si n'hi ha diversos, o una única equació si només hi hagués un cas. Cal resoldre-la per substitució.

d) (0.50 punts) Expressar el resultat anterior utilitzant notació asimptòtica.

Solució:

a) La talla del problema pot ser:

(1) el valor del primer argument del mètode, açò és, `num`; anomenarem a aquest número m.

(2) el nombre de xifres de `num`; anomenarem a aquest número n.

b) Sí que hi ha instàncies significatives, ja que és una cerca. En el millor cas, la xifra de les unitats de `num` és major o igual que la base `b` i en el cas pitjor totes les xifres de `num` són menors que `b`, açò és, `num` es pot representar en base `b`.

c) Plantegem l'equació de recurrència per a cadascuna de les dues instàncies significatives, en passos de programa, considerant cadascuna de les talles possibles.

(1) Per a talla m (valor de `num`) s'obté:

- En el cas millor: $T^m(m) = 1$ *p.p.*
- En el cas pitjor:

$$T^p(m) = \begin{cases} T^p(m/10) + 1 & \text{si } m > 0 \\ 1 & \text{si } m = 0 \end{cases}$$

Resolent-la per substitució:
$T^p(m) = T^p(m/10) + 1 = T^p(m/10^2) + 2 = \ldots = T^p(m/10^i) + i$. Si $1 \leq m/10^i < 10 \rightarrow i = \lfloor log_{10}m \rfloor$, amb el que $T^p(m) = T^p(m/10^{\lfloor log_{10}m \rfloor}) + \lfloor log_{10}m \rfloor = T^p(0) + 1 + \lfloor log_{10}m \rfloor$. S'arriba al cas base en el que $T^p(0) = 1$. Amb el que $T^p(m) = 2 + \lfloor log_{10}m \rfloor$ *p.p.*

(2) Per a talla n (nombre de xifres de `num`) s'obté:

- En el cas millor: $T^m(n) = 1$ *p.p.*
- En el cas pitjor:

$$T^p(n) = \begin{cases} T^p(n-1) + 1 & \text{si } n > 0 \\ 1 & \text{si } n = 0 \end{cases}$$

Resolent-la per substitució:
$T^p(n) = T^p(n-1) + 1 = T^p(n-2) + 2 = \ldots = T^p(n-i) + i$. S'arriba al cas base (talla 0) quan $n - i = 0 \rightarrow i = n$. Amb el que $T^p(n) = 1 + n$ *p.p.*

d) En notació asimptòtica:

(1) Per a talla m (valor de `num`), el cost temporal serà: $T^m(m) \in \Theta(1)$ i $T^p(m) \in \Theta(\log_{10} m)$, és a dir, les fites per al cost són $T(m) \in \Omega(1)$ i $T(m) \in O(\log_{10} m)$.

(2) Per a talla n (nombre de xifres de `num`), el cost temporal serà: $T^m(n) \in \Theta(1)$ i $T^p(n) \in \Theta(n)$, és a dir, les fites per al cost són $T(n) \in \Omega(1)$ i $T(n) \in O(n)$.

Com pots observar el cost temporal del mètode és el mateix només que expressat en funció de talles distintes. Recorda que el nombre de xifres d'un número enter m és $1 + \lfloor log_{10}m \rfloor$, açò és, n.