Parcial 1 - PRÀCTIQUES - PRG - ETSInf. Curs 2016-17
10 d'abril de 2017. Duració: 1 hora

Nota: L'examen s'avalua sobre 10 punts, però el seu pes específic en la nota final de PRG és de **0,8 punts**.

NOM:	GRUP DE PRÀCTIQUES:

1. 4 punts El mètode `esPrefixe(String, String)`, implementat en la pràctica 2, torna `true` si el primer paràmetre és prefixe del segon i `false` en cas contrari.

   Es demana: completar el mètode recursiu `comptaSubcadena(String, String)` que segueix per tal que, fent ús del mètode anterior, torne el número de vegades que una cadena no buida `a` és subcadena d'un altra cadena `b`. Per exemple, `comptaSubcadena("ab", "aaba")` ha de tornar `1`, `comptaSubcadena("aa", "abac")` ha de tornar `0` i `comptaSubcadena("aa", "aaaa")` ha de tornar `3`.

```
/** Precondició: a.length() > 0 */
public static int comptaSubcadena(String a, String b) {
    if (a.length() <= b.length()) {
         if (esPrefixe(a, b)) { return /* COMPLETAR */; }
         else { return /* COMPLETAR */; }
    }
    else { return 0; }
}
```

   Recorda que `s.substring(i)` és un mètode de la llibreria de Java que torna un objecte `String` que representa la substring de `s` formada pels caràcters compresos entre el `i` i el `s.length() - 1`.

Solució:

```
/** Precondició: a.length() > 0 */
public static int comptaSubcadena(String a, String b) {
    if (a.length() <= b.length()) {
        if (esPrefixe(a, b)) { return 1 + comptaSubcadena(a, b.substring(1)); }
        else { return comptaSubcadena(a, b.substring(1)); }
    }
    else { return 0; }
}
```

2. 2 punts **Es demana:** completar el mètode que segueix perquè torne un array d'enters de talla `t` que continga valors per al **cas pitjor** del mètode d'ordenació per **inserció directa**, però tenint en compte que la diferència entre dos valors consecutius qualssevol de l'array resultant ha de ser com a mínim de 2.

```
private static int[] casPitjorInsercio(int t) {
    int[] a = new int[t];
    for ( /* COMPLETAR */ ) {
        a[i] = /* COMPLETAR */;
    }
    return a;
}
```

Solució:

El cas pitjor del mètode d'ordenació per inserció directa es dóna quan els elements de l'array a ordenar estan ordenats decreixentment. A continuació, es presenten 2 formes diferents (d'entre moltes possibles solucions) de completar el bucle `for` del mètode `casPitjorInsercio(int)` per tal que, sent `a` l' array resultant, es complisca que $\forall$`i`, `0` $\leq$ `i` $<$ `a.length - 1`, `a[i] - a[i + 1]` ≥ 2.

```
        for (int i = 0; i < a.length; i++) {
           a[i] = t - (2 * i) - 1;
        }

        for (int i = 0, j = t - 1; i < a.length; i++, j -= 2) {
            a[i] = j;
        }
```

3. 4 punts En la classe `AlgorismesMesurables` està definit el mètode d'ordenació per inserció directa amb el següent perfil:

```
public static void insercio(int[] a)
```

En la classe `MesuraOrdenacio` estan definits els següents mètodes per a inicialitzar un array:

- `private static int[] crearArrayAleatori(int t)` que torna un array d'enters de talla `t` amb valors compresos entre `0` i `t - 1`.
- `private static int[] crearArrayOrdCreixent(int t)` que torna un array d'enters de talla `t` ordenat de forma creixent.
- `private static int[] crearArrayOrdDecreixent(int t)` que torna un array d'enters de talla `t` ordenat de forma decreixent.

Donat el següent fragment de codi on el mètode `mesuraInsercioCasPitjor()`, definit també en la classe `MesuraOrdenacio`, està incomplet:

```
// Constants que defineixen els parametres de mesura
public static final int MAXTALLA = 10000, INITALLA = 1000;
public static final int INCRTALLA = 1000, REPETICIONS = 200;
public static final double NMS = 1e3;  // relacio micro - nanosegons

public static void mesuraInsercioCasPitjor() {
    System.out.printf("# Talla    Pitjor\n");
    System.out.printf("#------------------\n");
    long tp1, tp2;
    int[] a;
    double tpitjor;
    for (   /* COMPLETAR */   ) {

                /* COMPLETAR */

        tpitjor /= REPETICIONS;
        System.out.printf(Locale.US, "%8d   %8.2f\n", t, tpitjor / NMS);
    }
}
```

Es demana: completar el mètode `mesuraInsercioCasPitjor` per tal d'obtenir les mesures de temps (en microsegons) del **cas pitjor** del mètode d'ordenació per **inserció directa**.

Recorda que el mètode `static long nanoTime()`, en `java.lang.System`, torna el valor actual del temporitzador més precís del sistema en nanosegons.

Solució:

```
public static void mesuraInsercioCasPitjor() {
    System.out.printf("# Talla    Pitjor\n");
    System.out.printf("#------------------\n");
    long tp1, tp2;
    int[] a;
    double tpitjor;
    for (int t = INITALLA; t <= MAXTALLA; t += INCRTALLA) {
        tpitjor = 0;
        for (int r = 0; r < REPETICIONS; r++) {
            a = crearArrayOrdDecreixent(t);
            tp1 = System.nanoTime();
            AlgorismesMesurables.insercio(a);
            tp2 = System.nanoTime();
            tpitjor += tp2 - tp1;
        }
        tpitjor /= REPETICIONS;
        System.out.printf(Locale.US, "%8d   %8.2f\n", t, tpitjor / NMS);
    }
}
```