Parcial 1 - Prácticas - PRG - ETSInf - Curso 2013/14
23 de junio de 2014 - Duración: 50 minutos

1. [2 puntos] Completar el siguiente método para que resuelva el problema de las Torres de Hanoi:

```
public static void hanoi(int discos, String origen, String destino, String auxiliar) {
   if( /* COMPLETAR */ )           // Caso base
      moverDisco( origen, destino );
   else {                          // Caso general
      hanoi( /* COMPLETAR */ );
      moverDisco( /* COMPLETAR */ );
      hanoi( /* COMPLETAR */ );
   }
}
```

Solución:

```
public static void hanoi(int discos, String origen, String destino, String auxiliar) {
   if( discos == 1 )           // Caso base
      moverDisco( origen, destino );
   else {                          // Caso general
      hanoi( discos-1, origen, auxiliar, destino );
      moverDisco( origen, destino );
      hanoi( discos-1, auxiliar, destino, origen );
   }
}
```

2. [3 puntos] Implementar un método **RECURSIVO** cuyo perfil debe ser:

```
public static int apariciones(String a, String b)
```

que devuelve la cantidad de apariciones de la cadena `a` en la cadena `b`. Por ejemplo, si `a`=*coc* y `b`=*coca de cocochas cocinadas con coco*, `a` aparece en `b` cinco veces. Se supondrá que la cadena `a` no es la cadena vacía.

Se **debe** resolver utilizando el método implementado en la *práctica 2*, cuyo perfil es:

```
public static boolean esPrefijo(String a, String b)
```

Solución:

```
public static int apariciones(String a, String b) {
   if ( a.length() > b.length() ) return 0;
   else { int c = esPrefijo(a,b) ? 1 : 0;
          return c + apariciones(a, b.substring(1));}
}
```

3. [5 puntos] Se dispone de un método con perfil `public static void algoritmo(int n)` que implementa cierto algoritmo cuyo coste tiene como talla el parámetro `n`. Se pide completar el método:

```
public static void medidaAlgoritmo(int tallaIni, int tallaFin, int tallaInc, int numRep) {
    System.out.printf("# Talla    Tiempo promedio (sg.)\n");
    System.out.printf("#------------------------------\n");

                    /* COMPLETAR */

}
```

para que mida de forma empírica (práctica) el coste de `algoritmo(n)` invocándolo para tallas comprendidas entre `tallaIni` y `tallaFin` con incrementos de `tallaInc`. Para mejorar la estimación del tiempo medido, la medición se repetirá `numRep` veces para cada talla mostrando por pantalla el valor de cada una de las tallas y el promedio de los tiempos medidos en **segundos**.

Se puede utilizar el método `public static long nanoTime()`, de la clase `java.lang.System`, que devuelve el valor actual del temporizador del sistema en **nanosegundos** (1 nanosegundo = 10^{-9}segundos).

NOTA: se considera que los métodos `algoritmo` y `medidaAlgoritmo` se encuentran en la misma clase.

Por ejemplo, una salida para la invocación `medidaAlgoritmo(10000, 20000, 1000, 10)` es:

```
# Talla    Tiempo promedio (sg.)
#------------------------------
  10000         5.32
  11000         6.28
  12000         8.61
   ...          ...
  20000         30.45
```

Solución:

```
public static void medidaAlgoritmo(int tallaIni, int tallaFin, int tallaInc, int numRep) {
    System.out.printf("# Talla    Tiempo promedio (sg.)\n");
    System.out.printf("#------------------------------\n");
    long t1 = 0, t2 = 0, tt = 0; double tmedt = 0;   // Tiempos
    for (int t=tallaIni; t<=tallaFin; t+=tallaInc) {
      tt = 0;                          // Tiempo acumulado inicial a 0
      for (int r=0;r<numRep;r++) {
        t1 = System.nanoTime();        // Tiempo inicial
        algoritmo(t);
        t2 = System.nanoTime();        // Tiempo final
        tt += (t2-t1);                 // Actualizar tiempo acumulado
      }
      tmedt = (double)tt/numRep;       // Tiempo promedio del caso promedio
      System.out.printf("%8d  %8d\n", t, tmedt*1.0e-9);
    }
}
```