EI1013/MT1013 Estructuras de datos

Examen final - Segunda convocatoria

Viernes, 22 de Junio de 2012

1. (**1,5 puntos**) Dos listas son *equivalentes* si contienen los mismos elementos aunque no necesariamente en el mismo orden. Implementa un método estático que reciba dos listas y devuelva **true** si son equivalentes y **false** en caso contrario.

2. (**2,5 puntos**) Un polinomio $P(x)$ de grado $n \geq 0$ de una sola variable x es un sumatorio de monomios del la forma

$$P(x) = \sum_{i=0}^{n} a_i x^i$$

donde $a_i, i = 0 \ldots n$ son números reales. Cada uno de los sumandos $a_i x^i$ se denomina *monomio*, donde a_i es el coeficiente e i es un entero que indica el grado del monomio.

Vas a implementar una clase para manipular polinomios de una variable. Por eficiencia, la clase sólo almacenará los monomios cuyo coeficiente a_i sea distinto de cero.

   a) Define la parte privada de la clase que almacenará los datos.

   b) Implementa un constructor que tome como entrada un vector **float [v]** con los coeficientes del polinomio. El elemento **v[i]** almacenará el coeficiente a_i.

   c) Incluye un método que tome como parámetro otro polinomio y lo sume al almacenado en el objeto.

3. **(1 punto)** Dada la siguiente secuencia de números: 1, 15, 2, 14, 3, 13, 4, 12, construir un montículo (min-heap) a partir de montículo vacío. Dibujar los sucesivos montículos que se van obteniendo tras cada inserción. Finalmente muestra los montículos que se obtienen tras realizar dos borrados.

4. **(2 puntos)** La siguiente clase genérica implementa un árbol general, es decir un árbol en el que cada nodo puede tener 0 o más hijos. La clase que implementa este tipo de árboles guarda el dato contenido en su raíz y una lista de subárboles del mismo tipo. Por simplicidad, asumimos que no puede haber árboles generales vacíos, y que un árbol sin hijos guarda un lista vacía de tamaño cero.

```
public class ArbolGeneral<T> {
  public T raiz;
  public EDList<ArbolGeneral<T>> hijos;
  ...
}
```

Implementar una función estática recursiva **gradoMaximo** que devuelva el valor del grado máximo de entre todos los nodos de un árbol.

5. **(2 puntos)** Los ficheros con extensión *.pgm* almacenan imágenes en niveles de gris en formato de texto. Un ejemplo de este tipo de ficheros sería éste

```
P5
# nombre.pgm
4 4
15
 0 0 15 15
 0 0 15 7
 0 0 15 7
 15 15 0 0
```

La primera línea contiene un *valor mágico* (normalmente 2 caracteres), seguido de una línea con comentarios que empieza con el carácter `'#'`. A continuación están el número de filas y de columnas de la imagen y el máximo nivel de gris usado en la imagen. Finalmente, escritos por filas están los datos de la imagen. Cada línea del fichero contiene los píxeles de una fila de la imagen. Cada píxel es un un valor entero indicando el nivel de gris correspondiente.

Deseamos crear un clase que almacene y gestiones imágenes *pgm*. Hemos definido los siguiente campos de la clase.

```
public class ImagenPGM {
  private String nombreFichero;
  private int maxNivelGris;
  private int[][] pixelImagen;

  //Constructor
  public ImagenPGM (String nombre);

}
```

Implementa el constructor. Éste tomará el nombre del fichero que contiene la imagen, leerá los datos y los almacenará en los campos privados.

6. **(1 punto)** ¿ Cual es el propósito de la función de dispersión en las tablas de dispersión? ¿ Qué condiciones ha de cumplir una buena función de dispersión?