Universidad de Granada

DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN E INTELIGENCIA ARTIFICIA

ETSII Informática y de Telecomunicación, C/ Periodista Daniel Saucedo Aranda s/n- 18071- Granada (España)

Estructuras de Datos
Curso 2015-2016. Convocatoria de Septiembre
Grado en Ingeniería Informática.
Doble Grado en Ingeniería Informática y Matemáticas

1. (2 puntos) Un almacén está desarrollando una aplicación que permita un acceso rápido a los productos de los que disponen y no se tiene claro qué estructura de datos elegir para almacenar los registros (nombre, código, año...). Las operaciones que son críticas para el almacén son: a) Inserción; b) búsqueda, que se hace por el nombre del producto; c) borrado, a partir del código único asociado a cada producto; d) imprimir de forma ordenada todos los productos. Analizar la eficiencia de las 4 operaciones si se usa como estructura de datos: Un vector ordenado, una lista ordenada, un ABB, un AVL y una Tabla hash.
¿Qué representación utilizarías para garantizar la mejor solución al problema?.

2. (2 puntos) Se dispone de un conjunto de claves de gran tamaño, que se almacenan de forma ordenada. La particularidad de este conjunto es que las claves pueden aparecer de forma consecutiva un numero indefinido de veces. Por ej:

{ 1,1,1,1,1,3,3,3,6,6,6,6,6,6,6,9,9,9,9,9,.....}

a) Para el acceso a las claves se pretende diseñar un TDA con la misma especificación del vector de la STL, sin embargo se busca una representación que almacene los datos con un espacio de memoria proporcional al número de claves distintas existentes. ¿Qué representación propondrías?
b) Considerando dicha representación, construye una función que dada una posición i del vector original devuelva el valor almacenado en esa posición.

3. (2 puntos) Usando la clase list<T>, construye la siguiente función que permite "duplicar" una lista (intercalando alternativamente tras cada elemento en la posición i, el elemento que está en la posición n-i-1 (i=0,1,...,n-1).

```
template <typename T>
void duplicar(const list<T> & inicial, list<T> & final)
```

Ejemplo 1:

Lista inicial: (a,b,c,d)
Lista final: (a,d,b,c,c,b,d,a)

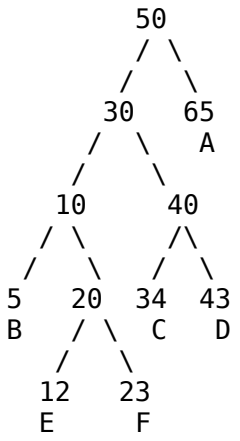
Ejemplo 2:

Lista inicial: (1,2,3,4,5)
Lista final: (1,5,2,4,3,3,4,2,5,1)

4. (2 puntos) Usando el TDA bintree, diseña una función para calcular la potencia externa PE de un ABB definida como:

PE = Σ 2^{profundidad(n)}
n ε {hojas}

P.ej. El árbol mostrado a continuación tiene una potencia externa de 58.



PE= A+B+C+D+E+F=
2^1+2^3+2^3+2^3+2^4+2^4=58

5.- (2 puntos) Implementa un iterador que itere sobre las claves que sean números impares en una clase diccionario definida como:

```
class diccionario{
    private:
       map<int, list<string> > datos;
    ……..
    ……..
}
```

Además de los métodos de la clase iterador, debes implementar begin y end de la clase diccionario.

Tiempo: 3 horas