AVL FILE & Random File

Organización de Archivos

Profesor Heider Sanchez

P1: AVL File: Usando como base los algoritmos de búsqueda e inserción del AVL, se le pide adaptarlos para trabaja con archivos de registros de longitud fija.

a) Implemente la función de búsqueda dado una search-key.
b) Implemente la función para agregar un nuevo registro.
a) Implemente una función para leer todos los registros ordenados.

```
struct Record
{
    //Defina el registro y los campos adicionales
};

class AVLFile
{
private:
    string filename;
    long pos_root;
public:
    AVLFile(string filename);
    Record find(TK key);
    void insert(Record record);
    vector<Record> inorder();
};
```

P2. Random File: Usando como base la implementación de archivos con registros de longitud fija y la estructura Registro, se le pide implementar lo siguiente:

c) Construir un índice (diccionario) para mantener asociado el search-key con las ubicaciones de los registros en el archivo de datos. El **índice debe ser cargado a memoria principal** al iniciar el programa.
d) Implementar la función de búsqueda dado una search-key.
e) Implemente la función para agregar un nuevo registro.
b) Cuando finalice el programa, deben **regresar el index al disco duro**.

```
class RandomFile
{
private:
    string fileName;
    string indexName;
    RandomIndex index; //diccionario en memoria principal
```

```
public:
    RandomFile(string _fileName){
        ...
        index = leerIndice();//desde disco duro
        ...
    }

    ~RandomFile(){
      ...
      guardarIndice(index);
      delete index;
      ...
    }
}
```

Entregable:

- **En ambos programas debe contener las pruebas de funcionalidad en la función main.**
- **Adjunte solo el código fuente**