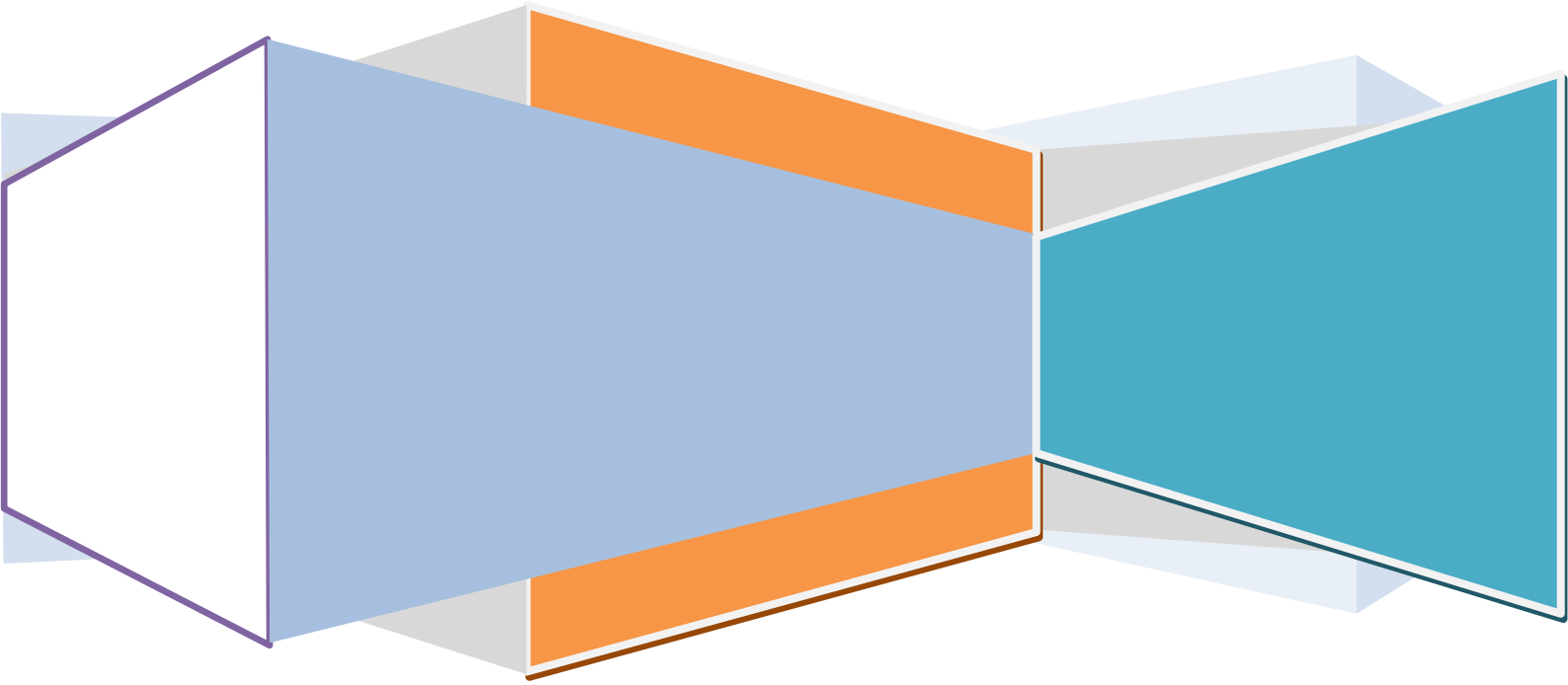
POLITÉCNICA

Mantenimiento de archivo de datos indexado con Arbol B

Estructura de datos II

Roberto Maestre Martínez

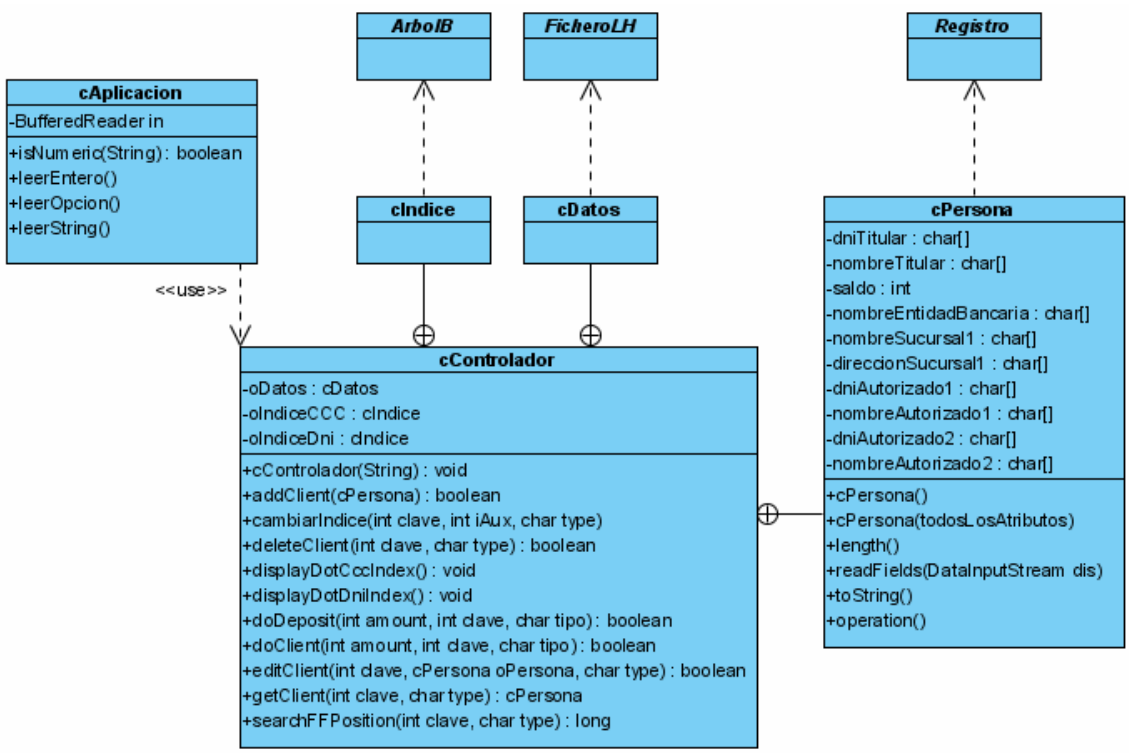


Diseño y arquitectura del sistema

- La nomenclatura utilizada es la siguiente .

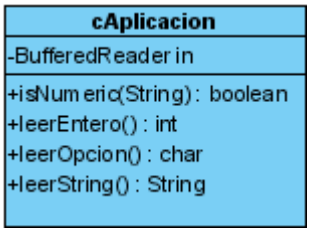
cVar – Indica nombre de clase Var
oVar – Indica un objeto de la clase Var

- El diagrama de clases utilizado es el mostrado a continuación, en el se especifican atributos, métodos, visibilidad, herencia y usos de las clases.



Implementación del sistema

- ***Clases diseñadas***



Se componee -> BufferedReader

Usa -> cControlador

La clase cAplicacion proporciona la interface para que el usuario interaccione con el sistema. Se compone de un objeto de la clase BufferedReader para poder leer por teclado las operaciones a realizar.

Procedimientos, funciones y algoritmos a destacar

1. **isNumeric(String) y leerEntero()**

   leerEntero() es un procedimiento que devuelve un entero Integer.parseInt(strAux) una vez convertido el String del teclado. Se apoya del método isNumeric(String) con el que implementa un while hasta que el usuario introduzca un enterro por teclado.

2. **leerOpcion()**

   Devuelve el char que es introducido por teclado

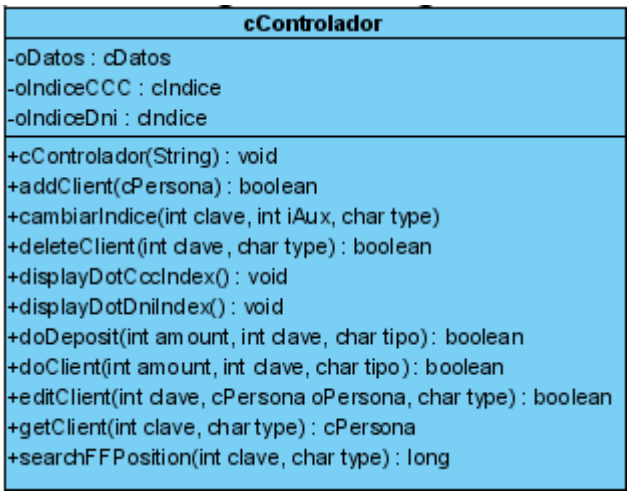
3. **leerString()**

   Devuelve como String la línea que se ha introducido por teclado

4. **public static void main (String args[])**

   Cuando instanciamos esta clase, creamos el objeto oControlador al cual le pasamos el nombre de la aplicación.

   Mediante un while pedimos por pantalla al usuario las distintas operaciones que están disponibles, utilizando el objecto oControlador y los datos introducidos por teclado para realizarlas.



Se componee -> cDatos, cIndice

La clase cControlador es la encargada de realizar las operaciones sobre los índices y el fichero de datos, es la espina dorsal de la aplicación.

Cuando llamamos al constructor este crea 3 estructuras en disco: dos índices (arboles B) y un fichero de datos para guardar los registros.

Las demás operaciones mostradas en la figura trabajan sobre estos índices y fichero de datos para gestionar las claves en el índice y en fichero de datos.

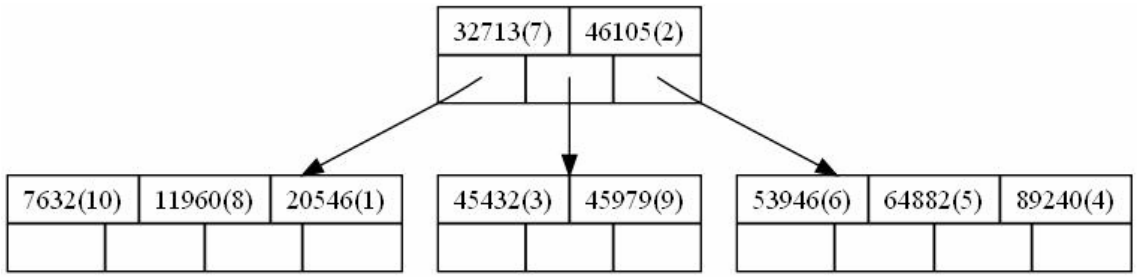
Cabe comentar la clase Clave

Clave.java Clave posicionFicheroFisico valor Clave() Clave(Clave) Clave(int) Clave(int, long) compareTo(Object) copyFrom(Clave) equals(Object) getPosicionFF() getValor() length() readFields(DataInputStream) setPosicionFicheroFisico(long) toString() writeFields(DataOutputStream)	Vamos a guardar dos valores en la clave, uno que es valor para poder indexarlo en el índice y luego un campo posicionFicheroFisico indicando en que posición se aloja la información.

Resumen y análisis

Los tiempos de búsqueda sobre el índice (con una estructura de árbol B), son mínimos, una búsqueda en un árbol B no visitara mas de 1+logd(n) (donde d es el orden del árbol) [1] . Gracias a esto, podemos indexar un gran cantidad de registros.

Un posible índice puede ser el siguiente



Donde el numero entre paréntesis , indica la posición dentro del fichero de datos donde se almacena el registro.