Registros de Longitud Fija y Variable

Profesor Heider Sanchez

P1: Se les proporciona un ejemplo de **archivo de texto** con registros de longitud fija (datos1.txt). Cada registro contiene cuatro campos de tamaño fijo. Note usted que si el tamaño del dato real no calza con la longitud establecida para dicho campo, entonces **se completa con espacios en blanco**. Usted debe desarrollar un programa que manipule dicho archivo usando la siguiente estructura base:

```
struct Alumno
{
  char codigo [5];
  char nombre [11];
  char apellidos [20];
  char carrera [15];
};
```

Encapsular las siguientes operaciones de manipulación del archivo en una clase llamada **FixedRecord**:

a) El constructor reciba el nombre del archivo.
b) Implemente la función ***vector<Alumno> load()*** para leer todos los registros del archivo.
c) Implemente la función ***void add(Alumno record)*** para agregar un nuevo registro al archivo.
a) Implemente la función ***Alumno readRecord(int pos)*** para obtener el registro de la posición "pos".
d) Realice pruebas de cada función en el programa principal.

P2: Ahora modifiquemos la esructura del registro agregando dos atributos no textuales: el ciclo y el costo de mensualidad. Implemente las siguientes operaciones con **archivo binario** ¿Por qué conviene usar archivo binario?.

```
struct Alumno
{
  char codigo [5];
  char nombre [11];
  char apellidos [20];
  char carrera [15];

  int ciclo;
  float mensualidad;
};
```

Encapsular las siguientes operaciones de manipulación de archivo con estrategia de eliminación FreeList en una clase llamada **FixedRecord**:

a) El constructor reciba el nombre del archivo.
b) Implemente la función ***vector<Alumno> load()*** para leer todos los registros del archivo.
c) Implemente la función ***void add(Alumno record)*** para agregar un nuevo registro al archivo.
a) Implemente la función ***Alumno readRecord(int pos)*** para obtener el registro de la posición "pos".
d) Implemente la función ***bool delete(int pos)*** para eliminar un registro al archivo utilizando la estrategia de **FreeList**.

e) Realice pruebas de cada función en el programa principal.

P3: Asuma que tiene un **archivo de texto** de registros de longitud variable. En donde, cada registro contiene cuatro campos de tamaño variable, los campos están delimitados por el caracter **|,** y los registros están delimitados por el salto de línea **\n**. La primera línea del archivo indica los nombres de cada campo por lo tanto ignore esta línea en la lectura.

```
Nombre|Apellidos|Carrera|Mensualidad
Howard|Paredes Zegarra|Computacion|1500.00
Penny|Vargas Cordero|Industrial|2550.50
Sheldon|Cooper Quizpe|Mecatronica|1850.00
```

```
struct Alumno
{
  string Nombre;
  string Apellidos;
  string Carrera;
  float mensualidad;
};
```

Encapsular las siguientes operaciones de manipulación del archivo en una clase llamada **VariableRecord**:

a) El constructor reciba el nombre del archivo.
b) Implemente la función ***vector<Alumno> load()*** para leer todos los registros del archivo.
c) Implemente la función ***void add(Alumno record)*** para agregar un nuevo registro al archivo.
d) Implemente la función ***Alumno readRecord(int pos)*** para obtener el registro de la posición "pos".

P4: Implemente otro programa para leer y escribir registros de longitud variable en un **archivo binario usando el tamaño del dato como separador.** El registro debe tener los campos que se indican en la siguiente figura.

```
struct Matricula
{
  string codigo;
  int ciclo;
  float mensualidad;
  string observaciones;
};
```

Se le pide implementar la estructura adecuada para acceder directamente a los registros considerando lo siguiente:

a) Maneje un archivo adicional (metadata) para indicar la posición inicial de cada registro. Evalúe si es necesario guardar también el tamaño del registro.
b) Implemente adecuadamente la función ***vector<Alumno> load()*** para leer todos los registros del archivo.

c) Implemente adecuadamente la función ***void add(Alumno record)*** para agregar un nuevo registro al archivo.
d) Implemente la función ***Matricula readRecord(int pos)*** para obtener el registro de la posición "pos".

Entregable:

- **Subir al canvas solo el código fuente de cada solución (p1.cpp, p2.cpp, p3.cpp, p4.cpp).**
- **Los cuatro programas deben contener pruebas de funcionalidad en el main (debe correr).**