Enunciado Entrega Evaluación Continua 2020-2021 **Estructura de Datos y Algoritmos II**

Ejercicios sobre MEMORIA SECUNDARIA

Ejercicio 1. Organización Indexada.

Teniendo en cuenta las especificaciones que se detallan en el **Enunciado 1** de las prácticas de Memoria Secundaria, correspondientes al Tema 6. Organización de Archivos, se deben implementar las siguientes funciones:

1. Una función que genere un fichero índice asociado al fichero secuencial *alumnos.dat* siguiendo una organización no secuencial indexada. El prototipo de la función será:

   int generaIndice(char *ficheroDatos, char *ficheroIndice)

   donde el primer parámetro indica el nombre del fichero de datos (*alumnos.dat*) y el segundo el nombre del nuevo fichero índice que se quiere generar (*alumnos.idx*). El fichero índice a generar será un fichero secuencial con registros de tipoIndice tal como se definen en indice.h donde, el primer campo, *dni*, se utiliza como clave de búsqueda y el segundo campo, *NRR*, proporcionará el número relativo del registro en el fichero de datos asociado. La función debe devolver un entero con el siguiente valor:

   - el número de registros finales en el fichero índice
   - -2 si hay problemas con el fichero de datos
   - -3 si hay problemas con el fichero de índice
   - -4 si ocurre algún otro error en el proceso

2. Una función que, utilizando el fichero indice generado por la función previa, busque un registro en el fichero de datos partiendo de su clave. El prototipo de la función será:

   int busquedaIndice(char *dni, char *ficheroDatos,char *ficheroIndice, tipoAlumno *alumno)

   donde el primer parámetro indica el valor del campo de búsqueda *dni*, el segundo el nombre del fichero de datos (*alumnos.dat*), el tercero el nombre del fichero indice(*alumnos.idx*) y el cuarto es el parámetro en el que se devuelve el registro buscado, si existe. La función devuelve un entero con el siguiente valor:

   - el NRR si el registro existe
   - -1 si el registro no existe
   - -2 si hay problemas con el fichero de datos
   - -3 si hay problemas con el fichero de índice
   - -4 si ocurre algún otro error en el proceso

Ejercicio 2. Organización Directa. Dispersión.

Teniendo en cuenta las especificaciones que se detallan en el **Enunciado 2** de las Prácticas de Memoria Secundaria, correspondientes al Tema 6. Organización de Archivos y el fichero (*alumnos.hash*) que se obtiene como resultado en el método de dispersión, se deben implementar las siguientes funciones:

1. Una nueva función de búsqueda con el siguiente prototipo:

   tipoAlumno *busquedaHash(FILE *f, char *dni, int *nCubo, int *ncuboDes, int *posReg, int *error)

   que busca en el archivo creado un registro a partir de su clave, parámetro *dni* de entrada a la función. Esta función devuelve el registro si lo encuentra o el valor NULL si no existe. Además devuelve información sobre la situación del registro en el fichero en los últimos tres parámetros, información que puede utilizarse en procesos posteriores de modificación y/o eliminación:

   - **nCubo:** número de cubo en el que se encuentra el registro si no está desbordado o en el que debería estar si está desbordado.
   - **nCuboDes:** si el registro está desbordado el número de cubo de desborde en el que se encuentra, considerando los cubos desbordados numerados secuencialmente a continuación de CUBOS. Si el registro no está en el área de desborde se le asigna a este parámetro el valor -1.
   - **posReg:** posición del registro en el cubo en el que se encuentra (inicial o desbordado)
   - **error:** este parámetro sirve para detectar si ha habido algún error en el proceso, la función debe asignarle alguno de los siguientes valores:
     - 0 si el proceso acaba correctamente y el registro exite
     - -1 si el proceso acaba correctamente pero el registro no existe
     - -2 si hay problemas con el fichero de datos
     - -4 si ocurre algún otro error en el proceso

2. Una función que, utilizando la información que aporta la función previa, permita modificar el campo provincia a los registros del fichero que se obtiene como resultado en el método de dispersión (*alumnos.hash*). La función a implementar debe seguir el prototipo:

   int modificarReg(char *fichero, char *dni, char *provincia)

   donde el primer parámetro indica el nombre del fichero *hash*, el segundo el *dni* del alumno a modificar y el último, el nuevo valor del campo provincia que se desea asignar al alumno. La función devuelve un entero con el siguiente valor:

   - el número de cubo en el que se encuentra el registro modificado si existe
   - -1 si el registro no existe
   - -2 si hay problemas con el fichero de datos
   - -4 si ocurre algún otro error en el proceso

Condiciones de la ENTREGA:

- Se debe subir un **único fichero comprimido** que incluya:
  - un fichero con el código C de las funciones del ejercicio 1 (indice.c)
  - un fichero con el código C de las funciones del ejercicio 2 (dispersion.c)
  - un **fichero de prueba** que permita verificar el correcto funcionamiento de las funciones implementadas
  - un fichero **makefile** que permita la perfecta compilación de estos ficheros junto con los proporcionados en la práctica
- Todos los ficheros deben incluir una línea inicial con el **nombre, dni y grupo de prácticas** del alumno. El fichero comprimido debe subirse a la tarea antes de las **23:55** horas del **4 de Junio de 2021**. No obstante, la tarea permitirá entregas retrasadas hasta el día **15 de Junio a las 14:00**.