Universidad Simón Bolívar
Departamento de Computación y T.I
CI-3825, Sistemas de Operación I
Septiembre-Diciembre 2022

Tarea # 1 (6%)

Objetivos Generales

Familiarizarse con los aspectos básicos de programación en lenguaje C.

Objetivos Específicos

- Adquirir destrezas en el lenguaje C
- Adquirir destrezas en la definición de estructuras compuestas y tipos.
- Adquirir destrezas en el uso y manejo eficiente de memoria dinámica.
- Utilizar las herramientas libres básicas de desarrollo de programas en ANSI C (editor de texto, make, compilador GNU, depurador)

Definición del problema

Ud debe desarrollar un programa que permita medir el tiempo de realizar búsquedas en arreglos. A continuación se muestra un esquema de cómo debe funcionar su programa:

1. Lee el archivo con los registros
    1. Crea un arreglo dinámico con los registros,
       Cada registro estará identificado con una clave única

    2. Crea un arreglo dinámico en la que cada posición tiene
        1. una clave y
        2. la ubicación de registro correspondiente en el arreglo anterior

2. Por cada clave en el archivo de claves
    1. Busca el registro en forma secuencial en el primer arreglo
    2. Imprime el registro en el archivo de salida secuencial con el tiempo que tomó la búsqueda,
    3. Busca, usando búsqueda binaria, el registro usando el arreglo de claves la posición.
    4. Imprime el registro en el archivo de salida índice con el tiempo que tomó la búsqueda,

3. Imprimir el tiempo promedio de los dos tipos de búsqueda

La sintaxis del comando es:

busca <tipo> <archivo datos> <archivo claves> <archivo salida índice> <archivo salida secuencial>

donde

- <tipo>: tendrá dos posibles valores: a o e. Indica si la clave será alfanumérica (a) o entera (e).
- <archivo datos>: Cada línea del archivo tendrá los datos de un registro. Puede asumir que los registros tendrán siempre los siguientes 3 campos:
  - **clave**: tendrá una longitud de 6 caracteres, éste podrá representar un entero o una cadena de caracteres pero será consistente con tipo.
  - **Nombre**: una cadena de caracteres y tendrá una longitud de 20 caracteres
  - **edad**: tendrá una longitud de 2 caracteres y será de tipo numérico
- <archivo claves>: Cada línea tendrá una cadena de caracteres que corresponda con una clave que debe ser buscada. Puede asumir que será consistente con tipo.
- <archivo salida índice>: será creado por su programa y se colocará el registro correspondiente a la clave buscada usando el arreglo índice.
- <archivo salida secuencial>: será creado por su programa y se colocará el registro correspondiente a la clave buscada en forma secuencial.

El programa deberá utilizar arreglos dinámicos para almacenar los datos. Ningún archivo será leído más de una vez.

Ejemplos

- Dos archivos de datos:

datos1.txt	**datos2.txt**

`Eleme1El nombre del elem 120`
`Eleme2El nombre del elem 222`
`Eleme3El nombre del elem 340`
`Eleme4El nombre del elem 467`
	`  1Pedro Perez          25`
	`  12Luiz Mendoza Alcanta44`
	`  234María Ana Perdomo   24`
	`121212Mariano de la Santos77`

  Note que en cada caso el tipo de la clave es diferente pero será consistente: en datos1, todas las claves son alfanuméricas y en datos2, todas las claves son enteras.

- Archivos de claves

claves1.txt	**claves2.txt**

`Eleme1`
`Eleme4`
`Eleme2`
	`1`
	`12`
	`1`
	`66`

Ejecución:

Observe los comandos siguientes y sus salidas esperadas:

```
# buscar a datos1.txt claves1.txt salida1Indice.txt  salida1Secuencial.txt
```

salidas1Indice.txt

```
Eleme1: (Eleme1, El nombre del elem 1, 20) 10ms
Eleme4: (Eleme4, El nombre del elem 4, 67) 12ms
Eleme2: (Eleme2, El nombre del elem 2, 22) 08ms
```

salidas1Secuencial.txt

```
Eleme1: (Eleme1, El nombre del elem 1, 20) 50ms
Eleme4: (Eleme4, El nombre del elem 4, 67) 92ms
Eleme2: (Eleme2, El nombre del elem 2, 22) 28ms
```

 # buscar n datos2.txt claves2.txt salida2Indice.txt salida2Secuencial.txt

salidas2Indice.txt

```
 1: (1, Pedro Perez, 25)           5ms
12: (12, Luiz Mendoza Alcanta, 44) 8ms
 1: (1, Pedro Perez, 25)           5ms
66: (No Existe)                   14ms
```

salidas2Secuencial.txt

```
 1: (1, Pedro Perez, 25)           25ms
12: (12, Luiz Mendoza Alcanta, 44) 38ms
 1: (1, Pedro Perez, 25)           56ms
66: (No Existe)                   245ms
```

Note que:

- Su implementación debe ser *case sensitive*.
- Ambos archivos de datos tienen claves de tipos diferentes, su programa debe almacenarlos según el caso.
- Los espacios al final del nombre fueron eliminados
- No está definido el número máximo de registros. Su aplicación debe funcionar independientemente del tamaño de los archivos,

No realice otros cambios en el contenido de los archivos. La verificación de la ejecución será realizada en forma binaria.

Recomendaciones

1. Diseñe su solución completa antes de proceder a implementar.
2. Trabaje en forma ordenada e incremental
3. Pruebe que cada una de sus funciones opera correctamente
4. Estructure bien su código.
5. El código debe tener una cantidad adecuada de comentarios.
6. Realice la documentación de su código a medida que vaya programando, dejarlo para el final se traduce en invertir más tiempo para hacerlo.

Entrega de la tarea:

Realización: Individual

Digital: Hasta las 11:59pm del Jueves semana 4.

Cada estudiante debe colocar en un archivo que con nombre carnet.tar.gz:

- los fuentes de su tarea (archivos .c y .h). Estos deben estar bien identificados, documentados y estructurados. Este conjunto será usado para compilar y correr su tarea.
- Un archivo de nombre *codigo.pdf* con los fuentes de su tarea. Exactamente la misma versión que los .c y .h. Éste será usado para evaluar el código de su tarea.
- El Makefile que pueda compilar/enlazar su aplicación

Note que debe estar subscrito al espacio canvas para poder optar a esta opción, ***no espere al día de la entrega para notificar que tiene problemas o que no se ha registrado***.

NOTAS IMPORTANTES:

1. Tarea que no cumpla con algunas de las especificaciones establecidas en este enunciado corre el riesgo de no ser corregida.
2. No use librerías que no hayan sido dadas en clases o en este enunciado sin autorización.

Material útil (leer las páginas del manual):

1. Búsqueda Binaria
    1. https://edukativos.com/apuntes/archives/10565
    2. https://es.wikipedia.org/wiki/B%C3%BAsqueda_binaria
2. Algunas funciones de la librería strings.h
    - strlen, strcat, strcpy, strcmp, strcasecmp
3. Algunas funciones de la librería stdio.h
    - fopen
    - fscanf, scanf, sscanf
    - printf, fprinf, sprintf
4. Algunas funciones de la librería stdlib.h
    - malloc, free
5. https://overiq.com/c-programming-101/fscanf-function-in-c/
6. https://fresh2refresh.com/c-programming/c-printf-and-scanf/