INF-253 Lenguajes de Programación
Tarea 2: C

Profesor: José Luis Martí Lara
Ayudante de Cátedras: Lucio Fondon Rebolledo
Ayudantes de Tareas: Sebastián Campos Muñoz - Gabriel Carmona Tabja

10 de mayo de 2021

1. Don Fondon y los autos que hablan

Después de tener la aplicación funcional, Don Fondon estaba trabajando tranquilamente en su bodega cuando sin previo aviso cae un meteorito fuera de su casa, a lo que él asustado se dirige a revisar. Al llegar a la zona de impacto se encuentra con un alienígena con forma de automóvil llamado Optimus C, esta entidad apenas se da cuenta sobre la presencia de Don Fondón le comienza a explicar rápidamente la procedencia de su especie y el terrible problema en el cual están inmersos. Don Fondon no entendiendo nada y asustado, le pregunta en que puede ayudar. Optimus C le dice que podría ayudarlos en crear un programa que permita realizar funciones de un Transimperativer y luchar mejor contra los Objecticons. Don Fondon no sabe nada de C y tiene miedo de lo que pueden hacer, por eso les dice a ustedes que lo ayuden.

2. Transimperativer

Los Transimperativer serán representados mediante una lista enlazada la cual guardará características. Estas características corresponderán a tipos de datos distintos los cuales tendrán un peso en los atributos del Transimperativer. Además, cada Transimperativer tendrá la capacidad de ejecutar acciones las cuales variarán de Transimperativer en Transimperativer. Finalmente, a partir de los atributos, los Transimperativer podrán batallar para saber qué Transimperativer es superior.

2.1. Definición de un Transimperativer

Un Transimperativer corresponderá a una estructura compuesta por una lista de características y dos punteros a funciones. Esta lista de características será una lista enlazada en la cual cada nodo alberga un struct compuesto por un char, un puntero a void y un puntero al nodo siguiente. El puntero a void guardará la información de la característica (esta información puede estar representada por un entero, arreglo de caracteres, carácter y arreglo de enteros), mientras que el char representará el tipo de la característica ('i' para enteros, 'c' para char, 'h' para arreglos de char y 'a' para arreglos de enteros). En resumen, los Transimperativer tendrán la siguiente forma:

```
typedef struct Transimperativer{
```

```
    void (*mezclar)(Transimperativer*, Transimperativer*);
    void (*separar)(Transimperativer*, Transimperativer*, Transimperativer*);
    Cuerpo cuerpo;
} Transimperativer;

typedef struct Cuerpo{
    nodo* curr;
    nodo* head;
    nodo* tail;
    int length;
    int posCurr;
} Cuerpo;

typedef struct nodo{
    void* caracteristica;
    int tipo;
    nodo* next;
} nodo;
```

Nota: Los arreglos de enteros y de char nunca estarán vacíos. El primer elemento del arreglo siempre representará el largo del arreglo, en el caso del arreglo de enteros el primer valor puede tomar cualquier número entero mayor o igual a 0 (si el valor es 0 significa que el arreglo está vacío). Para el caso del arreglo del char, el primer valor sólo puede ir del '0' al '9' (si el primero valor es '0', significa que el arreglo está vacío). Finalmente, el largo del arreglo no se debe utilizar para los cálculos relacionados al arreglo.

3. Funciones a implementar

Para la lista enlazada, toda función que sea importante para un correcto funcionamiento de lista enlazada debe ser implementada (Por ejemplo: size, append, erase, clear, etc).

Las siguientes funciones serán las que se deben implementar para la tarea.

- CrearTransimperativer: debe recibir la variable donde se creará el Transimperativer (puntero) y dos punteros a funciones. La función debe crear un Transimperativer y guardarlo en la variable de entrada.
- MezclarTransimperativer: debe recibir dos Transimperativer t1 y t2, donde ellos se intercambiarán características según la función mezclar que tenga el t1.
- SepararTransimperativer: debe recibir el Transimperativer a separar y los dos Transimperativer que nacerán a partir de la función seperar que tenga el Transimperativer a separar. El Transimperativer separado deja de existir.
- MostrarTransimperativer: debe recibir un Transimperativer y mostrar cada una de sus características.
- BorrarTransimperativer: debe recibir un Transimperativer y eliminarlo.
- Pelear: debe enfrentar en un combate frenético a dos Transimperativer y decir cual es el ganador. Para calcular esto debe realizar una suma de las características donde: los enteros se multiplican por 2, los caracteres se dividen por 2, los arreglos de caracteres se saca el promedio de los caracteres y los arreglos de enteros se saca el promedio de sus enteros.
- AnadirAtributo: debe recibir un Transimperativer y una característica, debe añadirle esa característica al Transimperativer recibido.

- QuitarAtributo: debe recibir un Transimperativer y el índice de la característica, debe quitarle esa característica al Transimperativer recibido.

Las siguientes funciones corresponden a funciones que los Transimperativer deben tener para mezclarse y separarse como se indico más arriba. Estas dos funciones son para que vean que su programa funciona, teóricamente hablando se le podría pasar cualquier función para mezclar o separar.

- MezclarMitad: debe recibir dos Transimperativer t1 y t2, donde t1 debe recibir la mitad de la derecha de t2 y t2 debe recibir la mitad de t1. Por ejemplo: si t1 es [1, 2, 3] y t2 es [4, 5, 6, 7, 8, 9]. El resultado será t1 igual a [1, 2, 7, 8, 9] y t2 igual a [4, 5, 6, 3].
- SepararMitad: debe recibir el Transimperativer a separar y los dos Transimperativer que nacerán de él. La separación debe ser realizada de tal forma que el hijo 1 tenga la mitad redondeada hacia arriba del padre y el hijo 2 tenga la otra mitad.

4. Definición de funciones a realizar

A continuación, se definen los headers de las funciones a realizar:

```
void CrearTransimperativer(Transimperativer* t1,
    void (*mezclar)(Transimperativer*, Transimperativer*),
    void (*separar)(Transimperativer*, Transimperativer*, Transimperativer*));
void MezclarTransimperativer(Transimperativer* t1, Transimperativer* t2);
void SepararTransimperativer(Transimperativer* padre, Transimperativer* hijo1,
                             Transimperativer* hijo2);
void MostrarTransimperativer(Transimperativer* t);
void BorrarTransimperativer(Transimperativer* t1);
void Pelear(Transimperativer* t1, Transimperativer* t2);
void AnadirAtributo(Transimperativer* t1, char tipo, void* caracteristica);
void QuitarAtributo(Transimperativer* t1, int indice);
void MezclarMitad(Transimperativer* t1, Transimperativer* t2);
void SepararMitad(Transimperativer* padre, Transimperativer* hijo1,
                  Transimperativer* hijo2);
```

5. Programa General

Su programa general debe generar un menú en el cual uno pueda usar todas las funciones implementadas (menos las dos ultimas, ya que estarán asociadas a cero o más Transimperativer). También los Transimperativer que se creen deben irse guardando en un arreglo, el cual permita mantener control de los Transimperativer creados.

También suponga que van a existir dos arreglos de tamaño 4, donde un arreglo contendrá los punteros a las funciones para mezclar y otro los punteros a las funciones para separar. Entonces cuando se vaya a crear un Transimperativer se debe pedir el índice para saber que función se le asignará al Transimperativer. El nombre del arreglo de funciones para mezclar se llamará funcionesMezclar y el arreglo de funciones para separar se llamará funcionesSeparar.

También debe existir una opción para terminar el programa y al seleccionar esta opción debe liberar toda la memoria pedida.

6. Archivos a Entregar

Resumiendo lo anterior, los archivos mínimos a entregar son:

- README.txt
- listaEnlazada.c
- listaEnlazada.h
- Transimperativer.c
- Transimperativer.h
- main.c
- MAKEFILE

El archivo listaEnlazadas.c debe contener las funciones implementadas relacionadas a la listaEnlazada.

El archivo Transimperativer.c debe contener las funciones implementadas relacionadas al manejo y control de Transimperativer

El archivo main.c debe ejecutar el menu y llamar a las funciones necesarias, este archivo contendrá los arreglos de funciones explicados en el punto 5.

7. Sobre Entrega

- El código debe venir **indentado y ordenado**.
- Las funciones deberán ir comentadas, explicando clara y brevemente lo que realiza, los parámetros que recibe y los que devuelve (en caso de que devuelva algo). Se deja libertad al formato del comentario.
- Debe estar presente el archivo MAKEFILE para que se efectúe la revisión, este debe compilar **TODOS** los archivos.
- Se debe trabajar de forma individual.
- **La entrega debe realizarse en tar.gz y debe llevar el nombre: Tarea2LP_RolIntegrante.tar.gz**
- **El archivo README.txt debe contener nombre y rol del alumno e instrucciones detalladas para la compilación y utilización de su programa.**
- La entrega será vía aula y el plazo máximo de entrega es hasta el **12 de Mayo a las 23:55 hora aula**.
- Las copias serán evaluadas con nota 0 y se informarán a las respectivas autoridades.
- Si el makefile no está bien realizado, la tarea no se revisará.
- Se utilizará Valgrind para detectar los leak de memoria.

8. Calificación

Todos comienzan con nota máxima y se les irá descontando por cada punto omitido (el puntaje que aparece al lado es el máximo de puntos que se les descontará en caso de incumplimiento):

- Transimperativer

    - CrearTransimperativer (10 pts)
    - MezclarTransimperativer (5 pts)
    - SepararTransimperativer (5 pts)
    - MostrarTransimperativer (10 pts)
    - BorrarTransimperativer (10 pts)
    - Pelear (10 pts)
    - AnadirAtributo (5 pts)
    - QuitarAtributo (5 pts)
    - MezclarMitad (10 pts)
    - SepararMitad (10 pts)
- Main (20 puntos)
- Descuentos
    - Código no ordenado (-10 puntos)
    - Código no compila (-100 puntos)
    - Warning (c/u -5 puntos)
    - Falta de comentarios (-30 puntos MAX)
    - Por cada día de atraso se descontarán 20 puntos (La primera hora de atraso serán solo 10 puntos).
    - Porcentaje de leak de memoria ((1-5) % -5 puntos (6- 50 %) -15 puntos (51-100 %) -25 puntos)

En caso de existir nota negativa esta será reemplazada por un 0.