Examen Final del Taller - Regulares

Algoritmos y Estructuras de Datos II

Tarea

El alumno deberá implementar el tipo abstracto de dato `set` de enteros (i.e. conjunto) en el lenguaje de programación
C, utilizando la técnica de ocultamiento de información visto en el taller de la materia.

Es requisito mínimo para aprobar es el siguiente:

- Implementar en C el TAD set (utilizando punteros a estructuras y manejo dinámico de memoria).
- Implementar en C una interfaz interactiva de línea de comando para que usuarios finales puedan usarlo (ver detalles más abajo).
- El programa resultado no debe tener memory leaks ni accesos inválidos a memoria, se chequeará tal condición usando valgrind (`make mem`).

Especificaciones

Se define el TAD set, que representa los conjuntos finitos de números enteros. Ejemplos:

```
{1; 3; 5; 6; 9; 11}
{23; 5; 3}
{}
```

El TAD deber a tener 2 constructores:

- uno para crear el conjunto vacío (i.e. `set_empty()`)
- otro para agregar un entero a un conjunto existente (i.e. `set_add()`). Este último constructor tiene por argumento un entero a y un set H.

El objetivo del examen es implementar el TAD set con una lista enlazada **sin repeticiones**. La implementación de la operación agregar debe modificar el conjunto que recibe como argumento, mientras que las de las operaciones unión e intersección deben dejar sus dos argumentos intactos y **generar un nuevo conjunto** que devuelven como resultado.
Para implementarlas puede convenirte implementar también una operación para clonar (es decir, generar una copia de) un conjunto.

Implementación

Los siguientes son los módulos a implementar:

- set.h y set.c, archivos de cabecera e implementación del TAD set.
- main.c, interfaz de línea de comandos.

La interfaz de línea de comandos (solo hacer en caso de rendir libre)

En el archivo main.c se debe implementar la interfaz de línea de comando para manipular conjuntos. La interfaz tiene que ser capaz de mantener 2 conjuntos a la vez para que el usuario final pueda hacer intersecciones y uniones de los mismos.

Cada opción del menú implícitamente debe trabajar sobre el “conjunto activo” el cual se cambia con la opción t.

Esta interfaz debe proveer las siguientes operaciones:

```
e -> Destruye el set actual y crea uno vacío
a -> Agrega un número al set actual
b -> Prueba si un elemento pertenece al set
l -> Imprime la cantidad de elementos
t -> Cambia el set actual entre el 0 y el 1
u -> Muestra la unión
i -> Muestra la intersección
c -> Corre una serie de operaciones de prueba
q -> Salir
```

A modo de ejemplo, se muestra a continuación la interacción con un usuario final esperada de la interfaz de línea de comandos para, por ejemplo, agregar un número al conjunto 1:

- El usuario aprieta t para cambiar al conjunto 1 (el conjunto default es 0)
- Usuario elige opción a
- La interfaz le pide al usuario que ingrese el elemento a agregar
- El usuario ingresa el numero entero

Recordar

- Se debe resolver el ejercicio en 4 horas (o menos).
- Se debe compilar pasando todos los argumentos usados en los proyectos.
- Comentar e indentar el código apropiadamente, siguiendo el estilo de código ya indicado por la cátedra (indentar con 4 espacios, no pasarse de las 80 columnas, inicializar todas las variables, etc).
- Todo el código tiene que usar la librería estándar de C, y no se puede usar extensiones GNU de la misma.

Criterio para aprobar

1. El programa resultante no debe tener memory leaks ni accesos (read o write) inválidos a la memoria.
2. Tiene que implementar todas las funciones pedidas en `set.c` y `main.c`
3. Tiene que pasar los tests de la opción ‘c’
4. Las funciones union e interseccion tiene que devolver una copia del set