Programación 3 - Curso 2015
Tarea A: Grafos

1. Objetivos

- Diseñar e implementar un tipo abstracto de datos de manera modular
- Manejo de grafos y diseño de algunas operaciones

2. Conocimientos previos

- Metodología de programación estructurada
- Tipos Abstractos de Datos (TAD)
- Conocimientos básicos de C*
- Grafos

3. Descripción del problema

La implementación que se solicita se basa en la realidad planteada a partir de las tareas 1.3 y 1.4 (trabajadas en monitoreos).

3.1. Estructura de datos

Se pide implementar una estructura de datos llamada *Deposito* para almacenar tanto los artículos como las colecciones que *El Indio* y sus colaboradores han recolectado a lo largo de sus travesías. La implementación de esta estructura corresponde al archivo `Deposito.h`.

3.2. Operaciones

Para realizar las operaciones que se han ido mencionando a lo largo del planteo de la realidad, se pide implementar los procedimientos y funciones incluidos en los siguientes archivos:

- `Cola.h`
- `ListaOrd.h`
- `OperacionesDeposito.h`
- `Pila.h`
- `tipoT.h`

Se deberá prestar atención a las **operaciones nuevas** en los módulos implementados en la Tarea 0.

Se recuerda que la resolución de los ejercicios planteados a lo largo de las entregas anteriores debe ayudar y facilitar de forma importante a la implementación de las operaciones solicitadas.

Se les brindará un módulo `Principal.cpp` que funcionará como intérprete de comandos para las operaciones que implementarán. Se recomienda que se utilicen archivos de entrada y salida para la ejecución del intérprete. Es decir, se realice de la siguiente forma:

```
$ ./Principal < entrada.in > salida.sal
```

4. Lenguaje a utilizar

El lenguaje a utilizar en este trabajo será `C` con las siguientes extensiones:

- Operadores `new` y `delete`.
- Pasaje de parámetros por referencia (uso de `&`).
- Declaración de tipos como en `C++` para registros y enumerados.
- Uso del tipo `bool` predefinido en `C++`.
- Comentarios en línea (//).

5. Sobre lo que se espera

Para cada módulo de cabecera `.h` con los prototipos de las operaciones y estructuras solicitadas debe entregarse un módulo `.cpp` con la implementación. Deben respetarse estrictamente los prototipos especificados. Esto es: nombre de la operación, tipo, orden y forma de pasaje de los parámetros y tipo de retorno.

Los módulos de cabecera pueden bajarse de la sección *Grafos* del sitio en el EVA del curso. Estos módulos no forman parte de la entrega y, por lo tanto, **NO DEBEN SER MODIFICADOS**.

Se brindará un archivo `test.sh`, al igual que en la Tarea 0, de forma que les permita probar la tarea fácilmente y generar el archivo de entrega.

Los módulos deben funcionar en el ambiente de las salas Unix de facultad. Las mismas están en los salones 314, 401 y 402. Se espera que todos los módulos compilen sin errores (utilizando las flags `-Wall` y `-Werror`), se ejecuten sin colgarse y den los resultados correctos.

6. Consulta y respuesta de dudas

Las dudas específicas sobre esta parte de la tarea se responderán de forma exclusiva en los foros del EVA del curso. Cuando el cuerpo docente lo considere pertinente se realizarán aclaraciones a través del foro *Novedades*, pero incitamos a los estudiantes estar atentos a lo que se responda en el foro *Consultas sobre implementación*.

7. Fecha de entrega

La fecha de entrega del laboratorio será el **jueves 24 de setiembre a las 18:00**. Se deberá entregar un archivo `EntregaA.tar.gz` que contenga, **respetando el formato de los nombres**, los siguientes archivos:

- `Cola.cpp`
- `Deposito.cpp`
- `ListaOrd.cpp`
- `OperacionesDeposito.cpp`
- `Pila.cpp`
- `tipoT.cpp`

La primera línea de cada uno de los archivos debe contener un comentario con la cédula de los integrantes del grupo, sin puntos ni dígito de verificación. Por ejemplo, si las cédulas son `1.234.567-8`, `4.254.566-2` y `3.339.717-0`, la primera línea de cada archivo deberá ser exactamente:

```
/* 1234567 4254566 3339717 */
```

8. Sobre la individualidad del trabajo

Para este obligatorio rige el reglamento de No Individualidad publicado en la página web del curso.

9. Defensa

El grupo cuya entrega no se considere satisfactoria, podrá hacer una defensa. Esta defensa consiste en hacer una nueva entrega junto con un archivo de texto explicando los cambios realizados. Esos cambios deben consistir en la corrección de **pequeños errores** que se detectan al examinar nuevos casos de prueba. No pueden implicar un cambio sustancial.