DEPARTAMENTO
INGENIERÍA INFORMÁTICA
Y CIENCIAS DE LA COMPUTACIÓN
FACULTAD DE INGENIERÍA UNIVERSIDAD DE CONCEPCIÓN

Estructuras de Datos (2021-1)

Laboratorio 3

Profesor: Diego Seco

Ayudantes: Alexis Espinoza, Catalina Pezo, Alexander Irribarra, Lucas Kraemer

Objetivos

Los objetivos del laboratorio son:

- Mejorar la programación, compilación y ejecución de programas escritos en lenguaje *C++* u otros.
- Implementar distintas estructuras de datos.
- Analizar y comparar diferentes implementaciones de un mismo tipo de dato abstracto.

Ejercicios

1. Crear el **ADT** (tipo de dato abstracto) **List** en una clase que tenga el mismo nombre. Esta interfaz debe contener los siguiente métodos como mínimo:
   - **Insertar elemento al principio:** `virtual void insert(int)=0;`
   - **Eliminar al final:** `virtual void pop()=0;`
   - **Acceder al i-ésimo elemento:** `virtual int at(int)=0;`
   - **Obtener la cantidad de elementos almacenados:** `virtual int size()=0;`
2. Implementar la estructura de datos **ArrayList**, que debe heredar de la clase **List** y contener sus métodos implementados (Se deben crear los ficheros ArrayList.h y ArrayList.cpp).
3. Implementar la estructura de datos **LinkedList**, que debe heredar de la clase **List** y contener sus métodos implementados (Se deben crear los ficheros LinkedList.h y LinkedList.cpp).

4. Se debe realizar un análisis experimental de las estructuras de datos implementadas, midiendo los tiempos promedio de cada uno de sus métodos. Para probar los métodos *insert* y *pop* se debe tomar el tiempo promedio de **insertar/remover un elemento**, ejecutando n veces el método, y para probar el método at, se debe **buscar** un elemento cuando ya hay n elementos insertados en la lista. Escribir sus resultados experimentales en una tabla y realizar gráficos comparativos por cada método. Recuerde tomar valores de entrada equidistantes entre sí para una mejor apreciación de la complejidad.

5. ¿Cuál crees que es la mejor implementación para **ADT List**? ¿Por qué?

Observación

Los estudiantes pertenecientes al minor son libres de implementar las soluciones en el lenguaje de programación *C++*, *Java* o *Python*, en este caso pueden tomar el código proporcionado como una base para empezar a realizar los ejercicios.

Para el caso de entregas en *Python*: En la implementación de **LinkedList** deben crear una clase **nodo** auxiliar, mientras que para **ArrayList** deben utilizar **array**[1] de *Python*.

Normas de entrega

Antes del lunes 19 de abril, se deben enviar todos los ejercicios resueltos a los ayudantes mediante la plataforma CANVAS.

La entrega debe consistir de dos archivos:

- Archivo PDF con el nombre completo, número de matrícula, las respuestas a las preguntas que correspondan y capturas de pantalla de la ejecución de sus códigos.
- Un archivo comprimido (.zip o .tar.gz) que contenga los ficheros de código fuente correspondiente (archivos .h y .cpp para *C++* u otras extensiones para los otros lenguajes).
- **IMPORTANTE:** Los archivos debe llamarse *apellido1_nombre_01.formato*

[1] `https://www.geeksforgeeks.org/python-arrays/`