FCEFQyN, Universidad Nacional de Río Cuarto
Asignatura: Estructuras de Datos y Algoritmos - Algoritmos I
Segundo Cuatrimestre de 2022

Práctica 2

Esta práctica consta de ejercicios obligatorios y ejercicios recomendados. El/los ejercicios obligatorios deberán ser resueltos, en grupos de no más de tres estudiantes, antes del comienzo de la siguiente práctica.

1. Lea las secciones desde la 3.1 a la 3.5 del capítulo 3 de Data Structures and Problem Solving Using Java (Mark Allen Weiss 4th Edition).

2. Acceder al directorio `algoritmos/java/biblioteca/` del repositorio de la materia, que incluye la implementación parcial de una clase Libro, una clase Catálogo de libros y clases que contienen el método main que permiten testear las clases definidas.
    a. Analice las diferentes Clases y la documentación que incluyen (comentarios).
    b. Completar el método toString() de la clase Libro.
    c. Crear algunos libros adicionales y mostrar su contenido.
    d. Experimente la diferencia entre usar **==** o usar el método **`equals()`** para comparar libros. Interprete los resultados de ejecutar el método main de la clase MainLibro.
    e. Complete la implementación de la clase Catálogo de libros, definiendo los métodos: ***toString*** (representación de un catálogo), ***agregarLibro*** (toma un nuevo Libro y lo agrega al final del Catálogo, si el mismo no se encuentra lleno) y ***buscarPorTitulo*** (busca un Libro de acuerdo a un título dado).

3. Escribir una clase para el manejo de números racionales. Sus atributos son dos variables de tipo long. Puede suponerse que el denominador siempre es positivo. La clase debe proveer funcionalidades para sumar y restar racionales, y un método equals para comparar dos racionales por igualdad. Implementar también un método toString que convierte un racional en un String. Hacer un método main que cree varios racionales y les aplique diversas operaciones, mostrando sus resultados por pantalla.

4. Considere la Interface Lista provista en el repositorio de la materia. Se pide dar dos implementaciones diferentes de la Interface y definir una clase Main que permita el uso de Listas.

5. Acceder al directorio `algoritmos/java/colecciones/cola/` del repositorio de la materia, que incluye clases que Implementan el TAD Cola (Cola.java) y realice los siguientes ejercicios:
    a. Completar la implementación de los métodos de la clase ColaArregloFijo.java.
    b. Utilizando alguna de las implementaciones del TAD Cola, defina un programa que decida si una cadena, ingresada por línea de comandos, es un palíndromo o no.
    c. Resuelva el mismo problema planteado en el inciso b, pero utilizando una implementación diferente del TAD Cola a la que usó para resolverlo en el inciso anterior y compare los resultados.
    d. Defina en el lenguaje C++ una implementación del TAD Cola y luego compare las dos implementaciones del TAD.

6. Implemente en Java el TAD Pila genérico.
    a. Implemente un algoritmo que tome como argumento una secuencia de números y los imprima en orden inverso. Ayuda: Utilice una pila.
    b. Agregue una implementación de Pilas, diferente a la definida al comienzo de este ejercicio, es decir, si implementó una pila con arreglos cree una nueva pila con listas enlazadas y viceversa. Para las pilas con arreglos lance una excepción en caso de que la pila se llene.
    c. Cambie la implementación de pilas utilizada en el main, modificando lo menos posible el código.

7. Compare la implementación del TAD Pila definido en el ejercicio anterior con la implementación de Pila definido en el lenguaje C, en la práctica de repaso.

8. Modifique la implementación de pilas con arreglos para que cambie de tamaño dinámicamente, de manera que nunca se llene (hasta agotar la memoria).

9. Acceda al directorio `algoritmos/java/ejemploArchivos` del repositorio de la materia. Este contiene un método *mostrarUltimasNLineas* que toma el nombre de un archivo de texto y un número n, modifique este algoritmo, de manera tal que imprima sólo las últimas ***n*** líneas de los archivos. ¿Qué estructura de datos debería utilizar?

10. (Opcional) Implemente una lista que posea una operación de concatenación de tiempo constante (sin usar ciclos). Ayuda: use una lista circular, manteniendo un puntero al último elemento de la lista.

11. El directorio `algoritmos/compras/main` del repositorio de la materia, contiene una clase ListaDeCompras, con la implementación parcial de una aplicación simple para administrar una lista de compras.
    a. Complete la implementación de manera tal que almacene los artículos ingresados por el usuario de a uno por vez. El programa debe, además de almacenar los artículos, imprimirlos en el orden original al finalizar la confección de la lista. Utilizar una lista simplemente encadenada genérica.

    b. Modifique el programa anterior para que el usuario pueda ingresar la cantidad de artículos a comprar por cada ítem en la lista.

12. Provea una implementación de vectores de números reales en JAVA (ayuda: utilice arreglos). Las operaciones definidas sobre los vectores son: multiplicación por un escalar, suma y multiplicación (producto punto) de vectores del mismo tamaño. También deben poder compararse por igualdad. Cree algunos vectores y utilice sus operaciones.

Ejercicio Obligatorio:

- ➢ Implementar, en el lenguaje Java, un algoritmo que tome como argumentos una secuencia de paréntesis, corchetes y llaves y determine si la secuencia está balanceada. Por ejemplo, su algoritmo deberá retornar verdadero para [()][()()]() y falso para [(]). (Ayuda: Usar el TAD Pila.)

- ➢ Este ejercicio debe resolverse de forma grupal. Para esto pueden reutilizar los grupos creados para la actividad de la práctica anterior o crear un nuevo grupo (informar en estos casos). Los integrantes del grupo deben trabajar colaborativamente para resolver el ejercicio. Esto incluye la realización de commits que reflejen la contribución de cada integrante, con **mensajes representativos**.

- ➢ Para acceder a este ejercicio y unirse/crear los grupos de trabajo debe ingresar al siguiente enlace: https://classroom.github.com/a/E8sGQQIp

- La solución al ejercicio debe estar disponible, para su corrección, en el repositorio antes del comienzo de la práctica Nº 3.