Modelo de segundo examen integrador - Programación II Lunes Noche - 1C 2025

Profesor Monzón, Nicolás Alberto

18 de mayo de 2025

Requisitos para aprobar el integrador

A. Por actos de deshonestidad académica será sancionado.

B. Para poder aprobar este examen Ud. deberá estar administrativamente en condiciones de poder rendir el examen. En caso de no estarlo y rendir el examen, este no va a ser corregido y quedará anulado.

C. Deberá comprimir la carpeta `src` de su proyecto junto a los `txt` utilizados (también se admiten `pdf` y `MD`). El archivo comprimido deberá tener como nombre el número de grupo. Deberá enviarlo por mail a `nimonzon@uade.edu.ar`.

D. Deberá desarrollar un set de prueba para demostrar que funciona su código (no se piden test unitarios, pero si alguna prueba en el `main` del proyecto).

E. Solo podrá utilizar técnicas vistas en este curso. Está prohibido el uso de librerías, estructuras de datos abstractas que vienen por defecto en la JRE y el uso de genéricos. No respetar este punto es suficiente para desaprobar el integrador.

F. Deberá ser entregado dentro de las 3 horas y 30 minutos. a partir del horario de inicio.

G. CONDICIONES PARA APROBAR: el examen se mide con una escala logarítmica donde obtener al menos 60 puntos corresponderá a un 4 y se deberá cumplir con todos los puntos anteriores. En caso de no cumplir con alguno de los puntos anteriores, el examen queda desaprobado sin excepción.

Considere los siguientes predicados sobre los legajos de los integrantes.

$\mathcal{P}(\texttt{a})$ = la suma de los legajos módulo 3 es 0

$\mathcal{P}(\texttt{b})$ = la suma de los legajos módulo 3 es 1

$\mathcal{P}(\texttt{c})$ = la suma de los legajos módulo 3 es 2

1. Ejercicio 1 (30 %)

Crear el TDA `RepeatedStack`, imitando (en lo que se pueda) la implementación de `Stack` desarrollada en clase. Será una estructura lineal destructiva, que, aparentará tener apiladas duplas donde la primera componente es el elemento, y la segunda cuántas veces se apiló el elemento de forma consecutiva.

Es decir, si apilamos los elementos 1, 6, 8, 1, 1, 3, 3, 3, 3, 6, de izquierda a derecha, entonces deberá parecer que apilamos las duplas (1, 1), (6, 1), (8, 1), (1, 2), (3, 4), (6, 1), de izquierda a derecha.

Se espera:

- Definir un conjunto de métodos que permita resolver la mayor cantidad de algoritmos.
- Utilizar una cola con prioridad en lugar de un arreglo.
- Agregar a la definición de los métodos cuáles son las precondiciones, postcondiciones y estrategia de implementación.
- Indicar además en un archivo `txt` (`pdf` y `MD` también están permitidos) cuáles son los invariantes que consideran para la estructura.

Desapilar los elementos debería decrementar, y si el decremento hace que el total de veces apilador sea 0, entonces se deberá eliminar la dupla.

2. Ejercicio 2 (20 %)

Modificar la estructura del ejercicio 1, tal que

- Ahora se permitan ingresar duplas en lugar de números enteros. Es decir, si se agrega (1, 4), (1, 4), (2, 2), (3, 8), debería entenderse como que se agregaron las duplas ((1, 4), 2), ((2, 2), 1), ((3, 8), 1).
- Utilizar para esto una cola con prioridad de tres niveles. Se puede modificar la implementación estática o dinámica. La prioridad será el primer elemento de la dupla contenida como primer elemento, la segunda prioridad será el segundo elemento de la dupla contenida como primer elemento y el valor será el segundo elemento de la dupla.

3. Ejercicio 3 (20 %)

Crear una estructura para representar conjuntos de pilas, sin utilizar genéricos. Crear un Util que contenga las operaciones usuales: unión, intersección, diferencia, diferencia simétrica, pertenece, contiene, contiene o es igual, igualdad y cardinalidad.

4. Ejercicio 4 (10 %)

De la versión creada en el ejercicio 1, crear un map a `Stack`, y otro map para el sentido inverso.

5. Ejercicio 5 (10 %)

Crear el TDA `Tripleta` que cumpla las propiedades usuales de una 3-*upla*. Dar una implementación, invariantes, precondiciones, postcondiciones y definir una clase utilitaria con una función que proyecte la i-ésima componente (en este caso $i = 0$, $i = 1$, o $i = 2$).

6. Ejercicio 6 (10 %)

Explicar como se podrían crear mapas entre la estructura del ejercicio 2 y una pila de tripletas.