Primer examen integrador - Programación II
Jueves Noche - 1C 2025

Profesor Monzón, Nicolás Alberto

11 de abril 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 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 (50 %)

Crear el TDA `FlatDequeue`, imitando (en lo que se pueda) la implementación de `Queue` desarrollada en clase. Será una estructura lineal destructiva.

Una `FlatQueue` es una estructura que utiliza un arreglo virtual por dentro. Es decir, utiliza índices de inicio y fin en lugar de tener un `count` para indicar únicamente el fin. Para neustro caso, se busca implementar la estructura de tal forma que no haya desperdicio de memoria ni errores ténicos fuera de los habituales (los vistos en clase).

Por otro lado una `Dequeue` es una estructura que permite acceder al primer y al último elemento de una la cola.

El TDA `FlatDequeue` es una estructura que mezcla estos dos conceptos.

Se espera:

- Definir un conjunto de métodos que permita resolver la mayor cantidad de algoritmos.
- Si $\mathcal{P}(\texttt{a})$, considerar la estructura mutable. Si $\mathcal{P}(\texttt{b})$, considerar la estructura inmutable. Si $\mathcal{P}(\texttt{c})$, a elección.
- Agregar a la definición de los métodos cuáles son las precondiciones, post-condiciones y estrategia de implementación.
- Indicar además en un archivo .txt cuales son los invariantes que consideran para la estructura.
- Crear una implementación estática del TDA, considerando que el primer elemneto de la cola está en la posición `init`.
- Crear una implementación estática del TDA, considerando que el primer elemneto de la cola está en la posición `end`.

2. Ejercicio 2 (10 %)

Modificar la implementación estática de `Stack`, para que en lugar de usar un arreglo nativo, se utilice el TDA `FlatDequeue`.

3. Ejercicio 3 (10 %)

Desarrolle un método `map` que permita convertir una instancia de `FlatDequeue` a una instancia de `A`, y otro método `map` que permita convertir una instancia de `A` a una `FlatDequeue`.

- Si $\mathcal{P}(\texttt{a})$, considerar `A` = `Stack`.
- Si $\mathcal{P}(\texttt{b})$, considerar `A` = `Stack` o `A` = `Queue`.
- Si $\mathcal{P}(\texttt{c})$, considerar `A` = `Queue`.

4. Ejercicio 4 (10 %)

A. Crear un método que aplique el ordenamiento de burbuja a una instancia de `FlatDequeue`.

B. Crear un método que aplique convierta la estructura en un acumulado parcial. Ejemplo si esto fuese un arreglo.
Input
$[1, 2, 3, 4, 5]$
Output
$[1, 3, 6, 10, 15]$

- Si $\mathcal{P}(\texttt{a})$, elegir una de las dos opciones.
- Si $\mathcal{P}(\texttt{b})$, realizar sólo el ítem A.
- Si $\mathcal{P}(\texttt{c})$, realizar sólo el ítem B.

5. Ejercicio 5 (10 %)

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

6. Ejercicio 6 (10 %)

Crear el TDA `Tuple` que cumpla las propiedades usuales de una Tupla de n componentes ($n \in \mathbb{N}$). Dar una implementación, invariantes, precondiciones, postcondiciones y definir una clase utilitaria con una función que proyecte la i-ésima componente. Explique con sus palabras qué sucede cuando $n = 1$ y qué sucede cuando $n = 2$.