Trabajo Práctico 2
Pilas y colas

Estructuras de datos
Comision A (Turno Mañana)
Comision B (Turno Noche)
Segundo cuatrimestre de 2020

Docentes	Sergio Gonzalez Román García Ariel Clocchiatti
Email	sergio.gonzalez@unahur.edu.ar roman.garcia.uni@gmail.com aclocchi@gmail.com
Fecha de entrega	09 / 10 / 2020

Índice

1. Introducción

En este trabajo práctico vamos a integrar los temas de la primer parte de la materia, centrándonos en las estructuras de datos Pila (*Stack*) y Cola (*Queue*). Van a tener que utilizar los conocimientos que fueron adquiriendo hasta ahora, para resolver un problema e implementar esta solución en el lenguaje de programación *Python*.

No se pide modelar este ejercicio usando objetos. No se compliquen buscando herencias y clases abstractas. No está mal si necesitan poner un *"if"* en algunos métodos donde normalmente tratarían de usar herencia y polimorfismo.

1.1. Presentación del problema

En una empresa de remolque y reparación de vehículos deciden informatizar el manejo de los auxilios pedidos por los clientes, desde el momento del llamado hasta que termina el proceso. Para eso contratan a un/una estudiante de la UNAHUR para que modele el problema y diseñe e implemente una solución que le permita automatizar el proceso.

Para el modelado de los auxilios pedidos por los clientes, se deben tener en cuenta estas condiciones:

- Cada auxilio se identifica con el número de patente del vehículo (Formato de Patente: 3 Letras 3 Números).
- Cada auxilio tiene una zona de partida y una zona de llegada. Como la empresa trabaja en el AMBA, las zonas pueden ser las siguientes: *Sur*, *Norte*, *Este*, *Oeste* o *CABA*.
- Los auxilios pueden ser de 2 tipos: *Remolque* o *Reparación*. En los auxilios de *Reparación* la zona de partida y de llegada es la misma (ubicación del vehículo).
- Cada auxilio tiene dos estados: *Espera* o *Aprobado*. Según la aprobación de la compañía de seguros correspondiente.

Durante este trabajo van a implementar tipos de datos y operaciones para trabajar con los pedidos de auxilio de los clientes. Para resolver el problema de organización de la empresa, vamos a modelar los auxilios, las oficinas de atención telefónica y finalmente el edificio de la empresa con su central telefónica.

Figura 1: Una de las últimas unidades adquiridas por la empresa.

2. Objetivos

2.1. Oficinas de atención

Definir un TDA que represente a una oficina donde se reciben las llamadas de los clientes, que maneja dos colas de auxilios, una de auxilios de "Remolque" y otra de auxilios de "Reparación". Cada oficina se identifica con el interno en la central telefónica (Números entre 1 y 999). Las oficinas tienen una cantidad crítica de auxilios de cada tipo (por defecto 50 pedidos).

El TDA ***OficinaAtencion*** debe incluir las siguientes operaciones:

- **recibirAuxilio(auxilio):** Agrega el auxilio que recibe por parámetro a la cola que corresponde a su **tipo**. Si se excede la cantidad **crítica** en alguna cola, se almacena el auxilio, pero se debe informar por pantalla la situación.
- **primerAuxilioAEnviar():** Retorna el primer auxilio a enviar a los conductores de las grúas. Si hay auxilios de **Remolque**, devuelve el primer pedido de auxilio a extraer de la cola correspondiente, caso contrario, el auxilio a extraer de la cola de **Reparaciones**. No desencola el pedido, solo lo muestra.
- **enviarAuxilio(zonaDeGrua):** Recibe por parámetro la zona en donde se encuentra una grúa y desencola y retorna el primer auxilio que se le puede enviar. El auxilio debe tener como zona de partida la zona en la que esta la grúa (**zonaDeGrua**). Los pedidos de **Remolque** se tratan primero, si no hay ninguno de **Remolque** en la zona, se tratan los de **Reparación**.
- **auxiliosPorTipo():** Cantidad de auxilios de cada tipo (*Remolque* y *Reparación* por separado) y **retorna ambos valores**.
- **cantidadTotalAuxilios():** Cantidad **total** de auxilios en la oficina de atención, sumando los de las dos colas.
- **esCritica():** Retorna *True* si alguna de las dos colas tiene una cantidad de auxilios mayor que la cantidad **crítica** o *False* en caso contrario.
- **auxiliosEnEspera():** Retorna la cantidad total de auxilios (*Remolque* y *Reparación*) que estan en estado de *Espera*.
- **buscarAuxilio(nroPatente):** Recibe un número de patente (**nroPatente**) y si en alguna de las colas (cualquierade las dos) hay un auxilio pedido para ese vehiculo, lo retorna. **El auxilio no debe ser eliminado de la cola**.
- **eliminarAuxilio(nroPatente):** Recibe un número de patente (**nroPatente**) y si hay un pedido de auxilio para ese vehículo en alguna de las colas de la oficina de atención, **lo elimina de ella**.
- **cambiaDeTipo(nroPatente):** Cambia el tipo del auxilio del vehículo con la patente *nroPatente* (de *Reparación* a *Remolque* o viceversa), en consecuencia, **debe cambiarlo de cola**. La operación debe **verificar** previamente que exista un pedido de auxilio para ese vehiculo en la oficina.

Aclaración: Pueden usar los TDAs Pila (*Stack*) y Cola (*Queue*) que programaron en las guías de ejercicios correspondientes.

2.2. Edificio de Empresa

En el edificio de la empresa tenemos M pisos con N habitáculos cada uno (un habitáculo es un lugar físico). En cada habitáculo se ubica una oficina de atención telefónica con su número de interno. Modelar con un TDA el Edificio de la Empresa, teniendo en cuenta que puede haber habitáculos vacios, cuando un habitáculo esta vacío, se representa como *None* en la estructura. Implementar las siguientes operaciones para el TDA del ***EdificioEmpresa***:

- **establecerOficina(numeroPiso, numeroHabitaculo, oficinaAtencion):** Pone la **oficinaAtencion** en habitáculo correspondiente.
- **cantidadDeOficinasCriticas(piso):** Operación **recursiva** que retorna la cantidad de oficinas en situación **crítica** en el piso que recibe por parámetro.
- **oficinaMenosRecargada():** Retorna la ubicación (número de piso y número de habitáculo) de la oficina que tiene el **menor número** de auxilios de tipo *Remolque* pendientes en todo el edificio.
- **buscaOficina(nroInterno):** Recibe el número de interno y retorna la **ubicación** de la oficina en el edificio (número de piso y número de habitáculo).
- **centralTelefónica(pilaDeAuxilios):** Operación que actua como la **Central Telefónica** del edificio de la empresa. Recibe como entrada una pila de auxilios y debe enviar los auxilios de a uno a la **oficina menos recargada**.
- **moverAuxilio(nroPatente, internoOficinaOrigen, internoOficinaDestino):** Saca el auxilio del vehículo con patente *nroPatente* de la oficina **internoOficinaOrigen** y lo pasa a la oficina de **internoOficinaDestino**.

3. Datos de prueba

Previo a la entrega del trabajo práctico, deben controlar que su implementación funcione con un lote de datos de prueba que nosotros les vamos a entregar. El lunes 5 de octubre, subiremos al ***webcampus*** el lote de datos de prueba, junto con un programa en *Python* para ejecutarlo. Por favor, respetar los nombres de las operaciones y de los TDAs (***Auxilio***, ***OficinaAtencion*** y ***EdificioEmpresa***).

4. Entrega

La entrega del trabajo práctico debe ser:

- Un informe escrito (doc, pdf. etc), incluyendo:
  - Descripción de cada una de las **estructuras de datos** diseñadas e implementadas. Incluir una descripción escrita de los algoritmos. Pueden incluir diagramas de flujo.
  - Descripción de la implementación en *Python*. Explicar **claramente** que hace cada función y procedimiento implementados.
- Código completo y comentado de la implementación.
- Opcional: Video explicando cómo funciona el algoritmo.

Luego de la entrega, les vamos a hacer preguntas a las/los integrantes del grupo sobre el trabajo, asi que les aconsejamos no copiar código que encuentren en internet.