Universidad de San Carlos de Guatemala
Escuela de Ingeniería en Ciencias y Sistemas, Facultad de Ingeniería
Introducción a la programación y computación 2, 2do. Semestre 2022.

SOLUCIONES GUATEMALTECAS, SA

202001482 – DIEGO PAOLO MEDINA GARCIA

Resumen

El programa calculará el tiempo de espera para cada cliente, el tiempo promedio de atención para cada escritorio, así como el tiempo mínimo y máximo. Para resolver el programa se implementó TDA's listas simples y colas en el caso de los clientes. puede contener elementos repetidos. cada elemento de la lista tiene un índice que lo ubica dentro de la misma.

Una cola por otra parte es una lista de elementos en donde siempre se insertan nuevos elementos al final de la lista y se extraen elementos desde el inicio de la lista. Al momento de atender a clientes se generará una grafica hecha con la herramienta Graphviz para cada escritorio, la información que se genera en dicha grafica es la siguiente, el id del escritorio, el tiempo promedio de atención, el tiempo máximo y el tiempo mínimo.

The program will calculate the waiting time for each client, the average service time for each desk, as well as the minimum and maximum time. To solve the program, TDA's simple lists and queues were implemented in the case of clients. can contain repeated elements. each item in the list has an index that places it within the list.

A queue on the other hand is a list of elements where new elements are always inserted at the end of the list and elements are extracted from the beginning of the list. When serving customers, a graph made with the Graphviz tool will be generated for each desk, the information generated in said graph is the following: the desk id, the average service time, the maximum time and the minimum time.

Introducción

En este documento se tratará sobre el desarrollo del programa incursionado por Soluciones Guatemaltecas S.A.

Desde el código fuente hasta el flujo del programa.

Desarrollo del tema

Soluciones Guatemaltecas, S.A proporciona un software creado en el lenguaje de programación Python. El programa se maneja enteramente desde consola, introduciendo dos archivos con extensión xml, el primero corresponde a un listado de todas las empresas a quienes se les brinda el servicio y el segundo es la configuración de cada empresa y sucursal, con datos como los escritorios que estarán atendiendo en un inicio, el listado o cola de clientes que están en espera y sus respectivas transacciones

El programa calculará el tiempo de espera para cada cliente, el tiempo promedio de atención para cada escritorio, así como el tiempo mínimo y máximo.

No solo se puede manejar el programa por medio de carga de archivos si no , que también se pueden ingresar empresas y clientes manualmente, únicamente se debe conocer el id de la empresa, el nombre y las sucursales disponibles y para los clientes; dpi, nombre, y sus transacciones.

Para resolver el programa se implementó TDA's listas simples y colas en el caso de los clientes.

Siendo una lista, una estructura de datos muy importante en los lenguajes de programación donde: representa una colección de elementos ordenados. puede contener elementos repetidos. cada elemento de la lista tiene un índice que lo ubica dentro de la misma.

Una cola por otra parte es una lista de elementos en donde siempre se insertan nuevos elementos al final de la lista y se extraen elementos desde el inicio de la lista. También se conoce a las colas como listas FIFO (FIRST IN - FIRST OUT: el primero que entra es el primero que sale).

Al momento de atender a clientes se generará una grafica hecha con la herramienta Graphviz para cada escritorio, la información que se genera en dicha grafica es la siguiente, el id del escritorio, el tiempo promedio de atención, el tiempo máximo y el tiempo mínimo.

La estructura de los tda's son las siguientes:

Un lista de empresas, dentro de cada nodo empresa se almacena una lista para las sucursales y otra para las transacciones que puede realizar la empresa, dentro de la lista de sucursales, se almacenan nodos los cuales contiene una lista de clientes y otra de escritorios, dentro de la lista escritorios se almacenan nodos que contine una lista para los clientes atendidos donde se almacena el nodo cliente en el escritorio que fue atendido.

De tal manera que el flujo de la aplicación queda de la siguiente manera:

Primero se cargan los archivos de entrada.

Luego se listan las empresas, introduciendo el id de la empresa y posteriormente se listan los puntos de atención e introduciendo el id del punto de atención se desplegará toda la información como los

escritorios activos, desactivados, y el numero de clientes en espera.

Para atender a un cliente la lógica que se siguió fue el siguiente.

```
def atender_cliente(self, id_empresa, id_punto_atencion):
    empresa = self.lst_empresa.buscar_empresa(id_empresa)
    punto_atencion = empresa.lst_punto_atencion.buscar_punto_atencion(id_punto_atencion)
    lst_escritorios_activos = punto_atencion.lst_escritorio.lista_escritorios_activos()
    punto_atencion.lst_clientes.listar()

    if punto_atencion.lst_escritorio.escritorios_activos() != 0:



        for i in range(0, len(lst_escritorios_activos)):
            cliente = punto_atencion.lst_clientes.atender()
            if cliente != None:
                print('-----------------------------------------')
                print(f'Cliente atendido: {cliente.nombre_cliente}')
                print(f'Por escritorio: {lst_escritorios_activos[i].id_escritorio}')
                print('-----------------------------------------')
                """ nodo_escritorio = punto_atencion.lst_escritorio.buscar_escritorio(escritorio.id_escritorio)
                nodo_escritorio.lst_clientes_atendidos.agregar(cliente) """
                lst_escritorios_activos[i].lst_clientes_atendidos.agregar(cliente)
                lst_escritorios_activos[i].lst_clientes_atendidos.graficar(lst_escritorios_activos[i].id_escritorio)
                #escritorio.lst_clientes_atendidos.imprimir()

            else: print('No hay más clientes por atender!')

    else:
        print('Por el momento no hay escritorios activos para atender a clientes!')
```

Se toma el id de la empresa, el id del punto de atención y se recorre con ayuda de una estructura repetitiva for la lista de escritorios activos, se desencola el cliente más próximo y se retorna para posteriormente ser almacenado en la lista de clientes atendidos ubicada en cada nodo de escritorio.

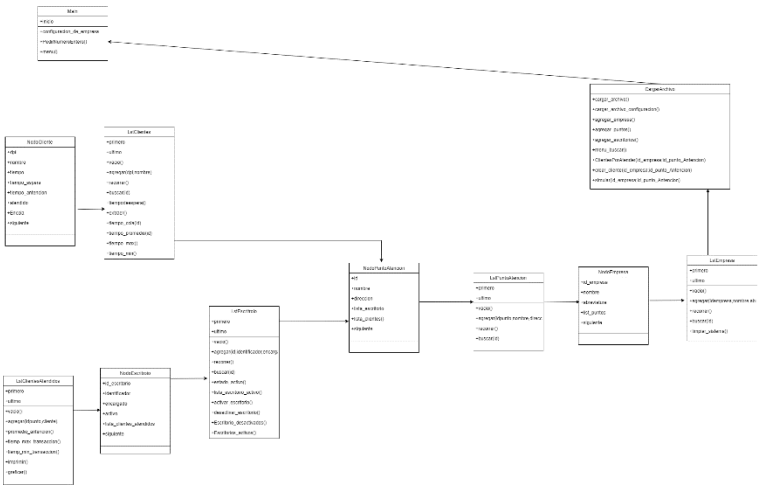
Cabe resaltar que también se podrá activar o desactivar escritorios a voluntad, únicamente se debe conocer el id de cada escritorio.

Graphviz es un programa de visualización gráfica de fuente abierta. La visualización de gráficos es una forma de representar información estructural como diagramas de redes y gráficos abstractos. Tiene aplicaciones importantes en redes, bioinformática, ingeniería de software, diseño web y de bases de datos, aprendizaje automático y en interfaces visuales para otros dominios técnicos.

Los programas de diseño Graphviz toman descripciones de gráficos en un lenguaje de texto simple y hacen diagramas en formatos útiles, como imágenes y SVG para páginas web; PDF o Postscript para incluir en otros documentos; o mostrar en un navegador gráfico interactivo. Graphviz tiene muchas funciones útiles para diagramas concretos, como opciones de colores, fuentes, diseños de nodos tabulares, estilos de línea, hipervínculos y formas personalizadas.

Conclusiones

a implementación de listas tiene mejores beneficios y flexibilidad para el funcionamiento de un programa a comparación de programarlo con arrays o tuplas, los datos pueden ser manipulados de una manera mucho más sencilla y al ser en esencia una clase puede tratársele como un objeto.



Referencias bibliográficas

Máximo 5 referencias en orden alfabético.

C. J. Date, (1991). *An introduction to Database Systems.* Addison-Wesley Publishing Company, Inc.