Trabajo Practico
Tribunales

Estructura de Datos

Grupo:
N° 13

Integrantes:
- Miguel Angel Robledo
- Adrian Yaniri

Informe:

Modulo 'Enum':

Este modulo codifica los distintos condiciones que puede llegar a tener un TDA expediente, mediante la clase “enum”.
Las condiciones son
*Fuero: civil, comercial, penal, laboral, familiar
class Fuero devuelve los códigos en enteros del 1 al 5 par a determinar el tipo de fuero que corresponde a cada expediente

civil = 1
penal = 2
laboral= 3
familiar= 4
comercial = 5

*Prioridad de cada expediente
class Prioridad devuelve los códigos entre 1 – 2 para determinar la prioridad de cada expediente

Normal = 1
Urgente= 2

*Estado de la causa
class Estado devuelve en código entre 1-2 el estado de la causa

Investigación = 1
En juicio =2

TDA Stack

El TDA modela un pila que luego se utilizara para apilar datos del tipo expedientes.
En la pila el primer dato en entra es el primer dato en salir.

def __inti__(self,)
Función constructor de la pila. Crea una pila vaciá.

def vaciar(self)
Función que elimina todos los elemento que hay en la misma.

def push(self,elemento)
Función que recibe por parámetro un elemento y lo agrega al principio de la pila.

def pop(self,elemento)
Función que elimina el primer elemento que se encuentra el la primer posición de la pila. Si la pila esta vaciá la función devuelve una excepción.

def top(self)
Función que devuelve el elemento de la pila. Si la pila esta vacía devuelve un Exception .

def clonar(self)
Función que clona la misma pila.

def tamanio(self)
Función que retorna la cantidad de elemento que posee la pila.

delf estaVacia(self)
Función que indica de la pila no tiene ningún elemento.

def __repr__(self)
Función para poder imprimir por pantalla una pila.

TDA Cola:

El TDA moldea una cola de elemento. Que sera utilizada para encolar los TDA expedientes
El primer dato que entra en la cola es el primero en salir

def __init__(self)
Función constructor de la cola. Crea una cola vaciá.

def empty(self)
Función que elimina todo los elemento de la cola.

def queue(self)
Función que agrega un elemento al final de la cola.

def dequeue(self)
Función que elimina el primer elemento de la cola.

def top(self)
Muetra el elemento que se encuentra en la primera posición de la cola

def clonar(self)
Clona la todos los elemento de la cola

def lenQueue(self)
función que retorna la cantidad de elemento que tiene la cola

def inEmpty(self)
retorna si la cola True si la cola no tiene elementos, caso contrario False

TDA Expedientes

Modela un expediente del juzgado. Los datos que debe mostrar el expediente son:

*numero de expediente

- el fuero al que pertenece
- la prioridad
- el estado del expediente

def __inti__(self,nro, fuero,prioridad,estado)

Función constructor. La función recibe por parámetro

nro = indica el numero de expediente
fuero = indica el fuero al que pertenece. El fuero esta representado entre 1 y 5
prioridad = indica la prioridad en que se encuentra el expediente. Puede ser normal = 1
urgente = 2
estado = indica el estado en que esta la causa. Puede ser investigación =1 o enjuicio =2

def getPrioridad(self)
Función que obtiene la prioridad del expediente.

def fuero(self)
Función que obtiene el fuero del expediente.

def esNomarl(self)
Función que indica True si el expediente tiene una prioridad = Normal.

def getNro(self)
Función que obtiene el numero del expediente.

def estaEnJuicio(self):
Función que retorna True si el estado del expediente esta enjuicio o no.

def cambiarPrioridad(self)
Función que cambia el tipo de prioridad de un expediente. Si el expediente tiene prioridad Normal lo cambia a prioridad Urgente y viceversa.

TDA Juzgado
Moldea un juzgado. El mismo tiene un nombre que es el del Juez que esta a cargo del juzgado, tiene 2 colas de expedientes. Una con expedientes Normales y otra con expedientes Urgente.

def __init__()
función constructor recibe por parámetros el nombre de el juez (string) que esta a cargo del juzgado
La función crea 2 TDA colas que representa la colas de los expedientes:
cola Normal, es una cola vaciá donde se guardaran lo expediente que tenga Prioridad Normal
cola Urgente es un cola vaciá donde se guardaran los expedientes Prioridad Urgente.

def getNombre(self)
Función que obtiene el nombre del juez a cargo del juzgado.

def tieneNormal(self)
Determina se la cola de expedientes Normales tiene o no elemento. Devuelve un booleano.

def tieneUrgente(self)
Determina si la cola de expedientes Urgentes tiene o no elementos. Devuelve un booleano.

def recibirExpedidente(self, expediente)
La función recibe por parámetro un expediente, Compara el la prioridad del expedientes y lo guardad en la cola correspondiente. La variable cantCritica indica la cantidad cuando una cola llego a la cantidad critica de expedientes e informa por pantalla si la misma alcanzo dicha cantidad.

def primerExpedienteATratar(self)
La función recorre las 2 colas de expedientes y compara la prioridad de los expedientes.
Primero recorre la lista de urgentes si encuentra algún expedientes retorna el expediente. No encuentra ningún expediente en la cola Urgentes, pasa a la cola Normal y retorna el primer expedientes de la cola. Si en caso de no encuentra ningún expedientes lanza una Exception.
No elimina el expediente de la cola.

def tratarExpediente(self)
La función retorna el primer expediente con prioridad Urgente de la cola
Primero pregunta si hay algún expediente en la cola urgente, usando la funcion tieneUrgentes()
Si encuentra algún expediente lo elimina de la cola
Si no encuentra expedientes en la cola Urgente pasa a al cola Normal y elimina el primer expediente de la cola Normal.

def cantidadDeExpedientesTotal(self)
Suma el total de los expedientes que hay en la cola urgente y normal, usando las función lenQueue() de la cola
Crea 2 variables sumNormal y sumUrgente y luego retorna la suma de la mismas

def expedientePorTipo(self)
La función devuelve la cantidad de expedientes que tiene cada cola usando la función lenQeuue() de la cola.
expNormal = suma todos los expedientes de la cola
expUrgente= suma todos los expedientes de la cola
y retorna los 2 valores obtenidos.

def esCritico(self)

Retorna True si el juzgado tiene alguna cola que haya pasado el limite critico de expedientes.
La variable cantCritica() indica la cantidad que es considerada critica.

def enJucio(self)
Función que retorna la cantidad de expedientes que tiene estado “enJuicio” de las 2 colas de expedientes
Las variables auxNormal y auxUrgente clonas las colas de Normal y Urgente del juzgado
La variable cant indica la cantidad de expedientes que tiene estado enJuicio
La función recorre al lista mediante dos while
El primero recorre la cola auxUrgente, valida que no esta vacía y luego suma en la variable cant 1 si el expediente tiene prioridad enJuicio.Al salir del if desencola el expediente de la lista auxUrgente
Realiza este recorrido hasta que la cola esta vacía,después hace lo mismo con la cola auxNormal
Al finalizar los recorridos retorna en la variable “cant” la cantidad de expedientes que tienen estado enJuicio.

def buscarExpedienteEn(nro,cola)
Función recibe por parámetro un numero de expediente un una cola.
Recorre la cola con while y va comparando el numero del primer expediente de la cola con el nro pasado por parámetro. Si el numero es igual al nro pasado por parámetro lo desencola de la lista y termina el ciclo, sino va desencolando los elemento hasta q la cola este vaciá y retorna el expediente, sino encuentra el expediente retorna None.

def buscarExpediente(nroExp)
Usando la función bucarExpendienEn(nro,cola) Recorre las 2 colas del juzgado para encontrar el el expediente pasado por parámetro, si no encuentra el expediente retorna None.

def eliminarDe(nro,cola)
Función que recibe por parámetro un nro de expediente y una cola.
Crea una cola auxiliar y vacía la cola que recibe por parámetro.
Recorre la cola auxiliar y va comparando los nro de los expedientes. Si los números de los expedientes son distintos, va encolando los expediente en la cola que recibe por parámetro. Al salir de la comparación va desencolando el expediente.
Retorna la lista que recibe por parámetro modificada sin el expediente que recibe por parámetro.

def eliminarExpediente(nroExp).
Usando la función eliminarDe(nro,cola) elimina el expediente que recibe por parámetro.
Retorna las colas sin el elemento eliminado, si el expediente no esta retorna la lista originales .

def cambiarDeEstado()
La funcion obtine por parametro un nro de expediente y lo cambia de estado
variable exp utiliza la funcion buscarExpediente(nro) para ver si encuentra el expediente que le pasan por parametro
Luego utiliza la funcion cambiarPrioridad() para cambiar la prioridad del expediente. Si esta normal la pasa a urgente y viceversa
Compra la variable exp con el nuevo estado. Si la prioridad es Normal, la encola en la cola de expedientes Normal, luego utiliza la funcion eliminarDe(nro) para eliminar el expediente de la cola anterior.

TDA Tribunales
Moldea el edificio de un tribunal.

El tribunal esta representado por un arreglo de matriz. Por N cantidad de pisos y M cantidad de oficinas.
Esta formado por una numero de oficinas y piso. Las oficinas de cada piso pueden estar ocupadas por un juzgado. La oficinas vaciás se representa con un None.
Se importa la numpy para poder moldar el arreglo.

def __init__(piso,oficinas):
Función que recibe por parámetro la cantidad de piso que tiene le edifico y la cantidad de oficinas que tiene cada piso.
Se usa el modulo numpy para crear el array vacío de tipos juzgados.

def getPisos()
Función para obtener la cantidad de piso del edificio tribunales.

def getOficinas()
Función para obtener la cantidad de oficinas que hay en un piso.

def oficinaActual(piso,oficina)
Función que retorna el juzgado que se encuentra en la ubicación pasadas por parámetro piso y oficinas.

def oficinaVacia(oficina)
Se le paso por parámetro una oficina del tribunal y retorna si la misma esta vacía o no.
Para saber si esta vaciá compara la oficina con None.

def establecerJuzgado(piso, oficina)
Toma los valores de piso y oficina y ubica el juzgado pasado por parámetro en el edificio de tribunales en el piso y oficina que les pasa por parámetro.

def juzgadoMenosRecargado()
La función buscan en todo el tribunal el juzgado que tiene menos expedientes críticos,en la variable menosCargado se guardan la el juzgado que tiene menos expedientes, con 2 ciclos for recorremos todo el edificio. En la variable actual se guarda la oficina actual que esta recorriendo el bucle
En cada iteración compra esa variable con None para saber si no esta vaciá la oficina, después compara la cola de expediente Urgente con el menosCargado. Si es el juzgadoMenosCargado() retorna la ubicación.
La variable i indica el piso y la variable j indica la oficina donde se encuentra el menosCargado.

def buscarJuez(juez)
Recorre toda el edificio buscando el juez que se pasa por parámetro. Con un ciclo for va recorriendo el edificio. La variable actual determina el juzgado actual.
Compara si no esta vacío, luego obtiene el nombre del juez mediante la función getNombre() del TDA juzgado dentro de la matriz tribunales.
Con la variable i y j retorna la ubicación del juez en el edificio

def mesaDeEntrada(pilaExp,juez)
Recibe por parámetro un TDA pila con expediente
Va des-apilando la pila los expedientes y los coloca en el juzgado
Primero cargamos los resultados de buscarJuez() con el str como parámetro a las variables piso y oficina para encontrar la oficina donde trabaja el juez ,después preguntamos si la pila de

expedientes esta NO esta vacía y si NO es critico para darles todos los expedientes a ese juez, sino realizamos el mismo procedimiento que buzcarJuez() pero con juzgadoMenosCargado() para saber que juez tiene menos urgentes, lo localizamos y le damos todos los expedientes.

Def moverExpediente(nroExp,juezOrigen,juezDestino)
la función pregunta si el juezDestino que pasamos por parámetro , si esCritico() se imprime un mensaje o se podría usar una excepción, sino el juezDestino recibe el expedientes con la función recibirExpediente() ,dentro de recibir buscamos con el juezOrigen el expediente que va a recibir el juezDestino con buscarExpediente(nroExp) y al final eliminamos el expediente del juezOrigen.