EDUCA2 – ¡Domina las Matemáticas y Programación!

Curso Intermedio de Programación en Python

Preparador: Alexanyer Naranjo

Guía Práctica N°4

Estructuras de Datos Avanzadas

1. Realice una función que reciba una matriz NxM como argumento y retorne la suma de todos sus elementos.
2. Realice una función que reciba una matriz NxM como argumento y retorne en una lista con la suma de cada fila. Cada valor en la lista es el resultado de sumar cada fila, es decir, en la posición 0 estará la suma de la fila 0, en la posición 1 estará la suma de la fila 1 y así sucesivamente.
3. Realice una función que reciba una matriz NxM como argumento y retorne en una lista con la suma de cada columna.
4. Realice una función que reciba una matriz NxM como argumento y retorne el número de filas que más ceros (0´s) contenga.
5. Realice una función que reciba una matriz NxM como argumento y retorne en un diccionario el número de fila (clave) con la mayor suma de toda la matriz (valor).
6. Escribir una clase TorreDeControl que modele el trabajo de una torre de control de un aeropuerto, con una pista de aterrizaje. La torre trabaja en dos etapas: *reconocimiento* y *acción*.
    - Escribir un método reconocimiento, que verifique si hay algún nuevo avión esperando para aterrizar y/o despegar, y de ser así los encole en la cola correspondiente. Para ello, utilizar el módulo random con la función **randrange(2)**.
    - Escribir un método acción, que haga aterrizar o bien despegar, al primero de los aviones que esté esperando (los que esperan para aterrizar tienen prioridad). Debe desencolar el avión de su cola y devolver la información correspondiente.

- Escribir un método __str__ que imprima el estado actual de ambas colas.

- Escribir un programa que inicialice la torre de control, y luego llame continuamente a los métodos reconocimiento y acción, imprimiendo la acción tomada y el estado de la torre de control cada vez.

7. **Atención a los pacientes de un consultorio médico:**
   - Escribir una clase ColaDePacientes, con los métodos nuevo_paciente, que reciba el nombre del paciente y lo encole, y un método proximo_paciente que devuelva el primer paciente en la cola y lo desencole.
   - Escribir una clase Recepción, que contenga un diccionario con las colas correspondientes a cada doctor o doctora, y el método nuevo_paciente que reciba el nombre del paciente y del especialista, y proximo_paciente que reciba el nombre de la persona liberada y devuelva el próximo paciente en espera.
   - Escribir un programa que permita al usuario ingresar nuevos pacientes o indicar que un consultorio se ha liberado y en ese caso imprima el próximo paciente en espera.
8. Implemente la primitiva Eliminar del Árbol Binario desarrollado en las clases teóricas del curso.
9. Implemente una primitiva que sea capaz de retornar el número más grande almacenado en todo el árbol.
10. Implemente un Árbol Binario que sea capaz de almacenar los nombres de personas por orden alfabético y desarrolle una primitiva que, dada una letra como argumento de entrada, retorne una lista con los nombres de las personas que empiecen por dicha letra.
11. Investigue e implemente los algoritmos de Búsqueda por Profundidad (DFS) y Búsqueda por Anchura (BFS) para Árboles Binarios. ¿Cuáles son las aplicaciones en un proyecto real para cada uno de estos algoritmos?