**TALLER FINAL.**

1. (10) Las listas enlazadas son por definición estructuras de datos homogéneas (es decir, todos sus nodos almacenan elementos del mismo tipo y estructura). ¿Eso es totalmente cierto?, es decir, ¿Sería posible tener una lista que almacene elementos diferentes en cada uno de sus nodos (tanto en tipo como en estructura)? Justifique su respuesta.

**R:** Bien, si es por definición, estrictamente la lista enlazada es eso y cualquier otra cosa debería tener otro nombre, por lo que sería totalmente cierto.

Sin embargo, es posible crear una lista con una estructura similar a las listas enlazadas para almacenar elementos diferentes en sus nodos, pues al final, son los nodos los que están conectados entre sí y no los datos. Claro, esto afectaría algunas operaciones como el ordenamiento y con él, la búsqueda, pero sigue siendo útil para almacenar datos de diferente tipo.

Un ejemplo puede ser un árbol binario, donde los valores pueden ser numéricos a categóricos y se trata a los nodos de manera distinta, según el número de hijos que tengan, pudiendo almacenar más de un dato.

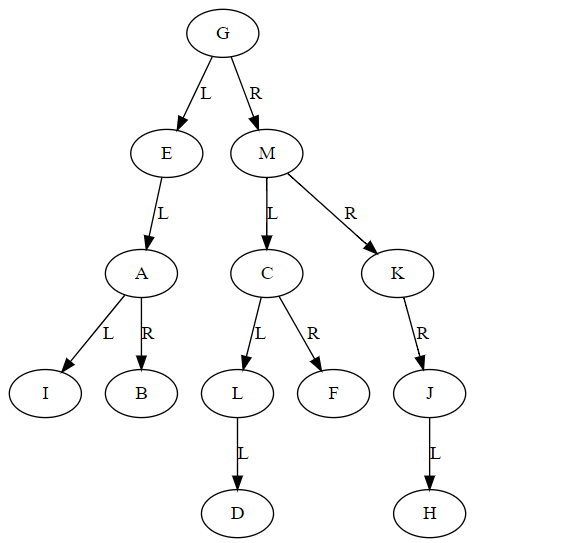
1. (10) Dada una pila, se desea conocer el promedio de los elementos que ella almacena. Como restricción la pila puede ser recorrida una sola vez.

**R:** Revisar el archivo pilapromedio.py

1. (15)El recorrido en preorden de un determinado árbol binario es: GEAIBMCLDFKJH y en inorden IABEGLDCFMKHJ .

Resolver:

3.1 Dibujar el árbol binario.(al crear el archivo en githun deberá mostrar la figura que quedaría puede ser en Word o excel)

**R: **

3.2 Dar el recorrido en postorden.

**R:** IBAEDLFCHJKMG

3.3 Diseñar una función para dar el recorrido en postorden dado el recorrido en preorden e inorden y escribir un programa para comprobar el resultado del apartado anterior

**R:** Revisar arbolorden.py

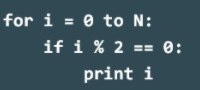
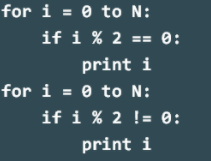
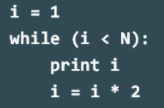
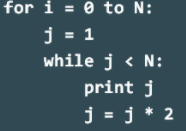
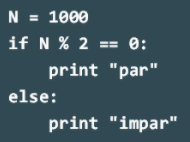
1. (10)En McDonald’s se pueden comprar nuggets de pollo en paquetes de 6, 9 o 20 unidades.

Escribir un algoritmo en Python que reciba un parámetro entero y decida si es posible o no comprar ese numero de nuggets.

Ejemplos

* es posible comprar 15 nuggets( un paquete de 6 y uno de 9)
* es posible comprar 18 nuggets( 3paquetes de 6 o 2 paquetes de 9)
* **Es imposible** compara 14 nuggets+

**R:** Revisar Nuggets.py

1. (5) Indicar con notación “Big -O”, la complejidad algorítmica(respecto al tiempo) de cada fragmento de código a continuación
   1.  O(n) porque hace n comparaciones.
   2.  O(n) porque hace 2 veces n comparaciones.
   3.  O(log n) porque el incremento de la variable a comparar es exponencial, lo que hace que el total de comparaciones termine rápidamente, específicamente, se realizan log2 n + 1 comparaciones.
   4.  O(n log n) porque el primer ciclo realiza n asignaciones y entra n veces a un ciclo de complejidad O(log n) caracterizado por el crecimiento exponencial de la variable de comparación del while (explicado en 5.3).
   5.  O(1) porque se tiene una cantidad definida de operaciones, en este caso una asignación, una comparación y una impresión.
2. (10) Escribe un programa que calcule la altura máxima de un árbol binario. Si está vacío, la altura se considera 0, y si solo hay la raíz se considera 1.

**R:** Revisar la función get\_nivel() de harbol.py

1. (10)En <https://github.com/gsepulv1/ST0245-08/blob/master/taller_Final/estudiantes.py> se encentra un algoritmo que almacena información de estudiantes.
   1. que estructura se utilizó para almacenar los datos de los estudiantes?

**R:** Se utilizó una lista simplemente enlazada.

* 1. que complejidad tiene este algoritmo?

**R:** Como utiliza un número definido de operaciones, sin contar la impresión, sería de O(1). Ahora suponiendo, que deba hacer el mismo método para añadir n estudiantes, las complejidad sería **O(n).**

* 1. Realice las optimizaciones de código a este algoritmo

**R:** Revisar estudiantes.py

* 1. Que complejidad tiene después de la mejora realizada en 7.3

**R:** Continúa con una complejidad de **O(n)**, sin embargo, ahora solo se debe introducir una lista para que haga el procedimiento a todos los alumnos.

1. (30) Implemente las modificaciones necesarias al algoritmo anterior y almacene e imprima la información de sus compañeros de curso empleando las presentaciones que se encuentran en el grupo de teams.

**R:** Revisar estudiantes2.py