Algoritmos 2, Curso Mendez ~ 2do Recuperatorio, 1er Cuatrimestre 2024 ~ 2024-07-04

Apellido y nombre: _____

Padrón: _____

Nota final:				

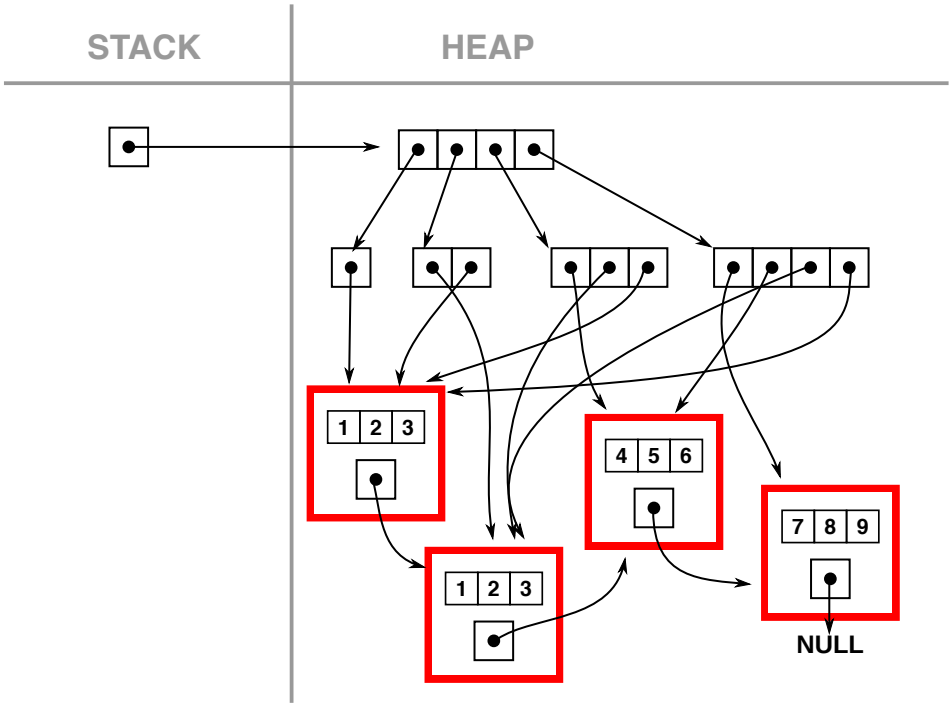
1) Escriba en pseudocódigo (o en **C** si le parece mas fácil) una implementación de las operaciones de **inserción** y **eliminación** de un **heap binario**. Explique cómo funcionan. Justifique en base a estas implementaciones la complejidad de las operaciones.

2) Explique cómo funciona **Quicksort**. Aplique el algoritmo al vector **V = [6,1,3,6,8,4,2]** de manera que quede ordenado de menor a mayor. Explique cada paso del algoritmo y muestre el estado del vector luego de cada paso del algoritmo.Justifique la complejidad del método.

3) Explique qué es un árbol **AVL** y cuales son sus diferencias respecto de un **ABB**. Comenzando por un árbol **AVL** vacío, inserte los siguientes elementos en secuencia: **[6,1,3,7,8,9,1,2]**. Muestre el estado del árbol luego de cada paso.

4) Escriba un programa en C (definiendo las variables, estructuras y tipos que crea conveniente) de forma tal que el uso de memoria del mismo sea como el que se muestra a continuación (puede agregar variables o punteros auxiliares si es necesario). Muestre el código que hace que dicho programa libere correctamente toda la memoria reservada.

Recuerde que para este ejercicio es fundamental que se haga la correcta verificación de las operaciones de memoria dinámica y de acceso a los punteros. Si el uso de memoria es incorrecto, el ejercicio queda invalidado. En la imagen los elementos que se muestran pegados son elementos contiguos en memoria (vectores) y los recuadros de colores estructuras de datos.



5) Escriba un algoritmo recursivo (sin utilizar **for**, **while**, etc) que dado un string devuelva el primer caracter del string que se encuentra repetido (o 0 en caso de que no se repita ninguno). No está permitido utilizar **ninguna función de la biblioteca estándar de C**. Como máximo se puede declarar **una** función auxiliar. Justifique la complejidad de la solución.

char primer_caracter_repetido(char* string);