Algoritmos 2, Curso Mendez ~ 1er Final, 1er Cuatrimestre 2025 ~ 2025-07-03

Apellido y nombre: _____

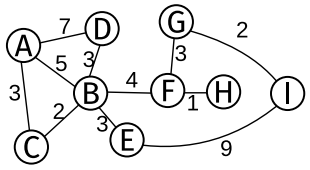
Padrón: _____ Modalidad: Completo / Reducido

Nota final:				

1) Ordene el siguiente vector de menor a mayor utilizando **QuickSort**. Muestre cada paso del algoritmo. Justifique la complejidad. Explique qué cuidado hay que tener para al aplicar el **Teorema Maestro** y por qué estas consideraciones no son necesarias en el caso de **Mergesort**. **V = [6,4,2,9,8,1,7,3]** .

2) Explique cómo funciona un árbol **B** y qué características lo definen. En un árbol B con 3 claves por nodo, inserve los siguientes elementos en el orden dado: '**M**', '**A**', '**L**', '**T**', '**G**', '**R**', '**I**', '**O**', '**S**'. Luego elimine '**M**' y '**R**'. Muestre el estado del árbol en cada paso.

3) Explique qué es el algoritmo de **Dijkstra** y para qué sirve. Aplíquelo al siguiente grafo mostrando el resultado de cada paso y el resultado final comenzando desde **A**.



4) Explique qué son los puntos de articulación de un grafo. Escriba (en **C99** o **Python**) un algoritmo para obtener los puntos de articulación. Explique cómo funciona y aplique el algoritmo al grafo del punto 3.

5) Explique qué es un **diccionario** y para qué se utiliza. Explique cómo implementar un diccionario utilizando una tabla de hash de direccionamiento cerrado. Dicho diccionario debe poseer una primitiva que permita iterar las claves insertadas (**en orden de inserción**). Muestre cómo debería ser la estructura en memoria y explique cómo funcionan las operaciones de inserción, eliminación iteración y búsqueda. Justifique la complejidad de cada operación.

Algoritmos 2, Curso Mendez ~ 1er Final, 1er Cuatrimestre 2025 ~ 2025-07-03

Apellido y nombre: _____

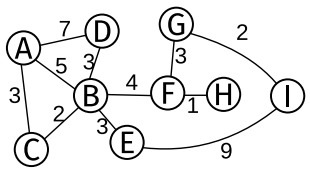
Padrón: _____ Modalidad: Completo / Reducido

Nota final:				

1) Ordene el siguiente vector de menor a mayor utilizando **QuickSort**. Muestre cada paso del algoritmo. Justifique la complejidad. Explique qué cuidado hay que tener para al aplicar el **Teorema Maestro** y por qué estas consideraciones no son necesarias en el caso de **Mergesort**. **V = [6,4,2,9,8,1,7,3]** .

2) Explique cómo funciona un árbol **B** y qué características lo definen. En un árbol B con 3 claves por nodo, inserve los siguientes elementos en el orden dado: '**M**', '**A**', '**L**', '**T**', '**G**', '**R**', '**I**', '**O**', '**S**'. Luego elimine '**M**' y '**R**'. Muestre el estado del árbol en cada paso.

3) Explique qué es el algoritmo de **Dijkstra** y para qué sirve. Aplíquelo al siguiente grafo mostrando el resultado de cada paso y el resultado final comenzando desde **A**.



4) Explique qué son los puntos de articulación de un grafo. Escriba (en **C99** o **Python**) un algoritmo para obtener los puntos de articulación. Explique cómo funciona y aplique el algoritmo al grafo del punto 3.

5) Explique qué es un **diccionario** y para qué se utiliza. Explique cómo implementar un diccionario utilizando una tabla de hash de direccionamiento cerrado. Dicho diccionario debe poseer una primitiva que permita iterar las claves insertadas (**en orden de inserción**). Muestre cómo debería ser la estructura en memoria y explique cómo funcionan las operaciones de inserción, eliminación iteración y búsqueda. Justifique la complejidad de cada operación.