1. Amplía la función **`construirArbolAnalisis`** para que pueda manejar expresiones matemáticas que no tienen espacios entre cada carácter.
2. Modifica las funciones **`construirArbolAnalisis`** y **`evaluar`** para que puedan manejar las sentencias booleanas (**and**, **or**, y **not**). Recuerda que "**not**" es un operador unario, por lo que esto complicará un poco la realización del código.
3. Utilizando el método **`encontrarSucesor`**, escribe un recorrido **`inorden` no recursivo** para un árbol binario de búsqueda.
4. Modifica la implementación de árbol binario de búsqueda para que maneje correctamente claves duplicadas. Es decir, si una clave ya está en el árbol, entonces la nueva carga útil debería sustituir a la antigua en lugar de **`agregar`** otro nodo con la misma clave.
5. Modifica la función **`imprimirExpresion`** para que no incluya un par de paréntesis 'extra' alrededor de cada número.
6. Utilizando el método **`construirMonticulo`**, escribe una función de ordenamiento que pueda ordenar una lista en tiempo $O(nlogn)$. (Buscar heapsort)
7. Implementa un montículo binario como un montículo **máx**.
8. Utilizando la clase **`MonticuloBinario`**, implementa una nueva clase llamada **`ColaPrioridad`**. Esta nueva clase **`ColaPrioridad`** debe implementar el constructor, además de los métodos **`agregar`** y **`avanzar`**.
9. Implementa la eliminación de un nodo y su posterior actualización y reequilibrio en un árbol **AVL**.
10. Dada la secuencia de claves enteras 20, 10, 30, 5, 25, 12, 3, 35, 22, 11, 6, 2, y la secuencia 100, 29, 71, 82, 48, 39, 101, 22, 46, 17, 3, 20, 25, 10, representa gráficamente los árboles **AVL** correspondientes e indica en qué momento se efectuó una rotación.

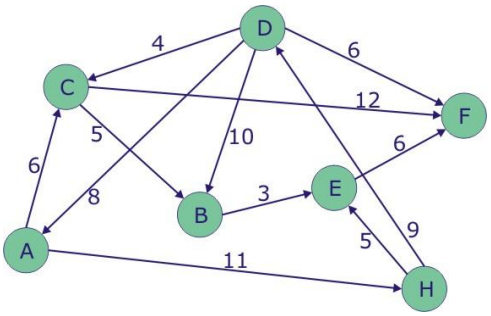
---

11. Crea el grafo correspondiente a la siguiente lista de aristas.

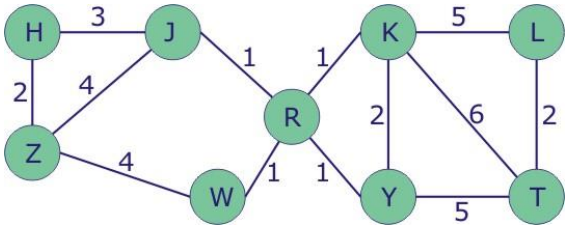
de	a	costo
1	2	10
1	3	15
1	6	5
2	3	7
3	4	7
3	6	10
4	5	7
6	4	5
5	6	13

12. Haciendo caso omiso de las ponderaciones, realiza una **búsqueda en anchura** en el grafo de la pregunta anterior.
13. ¿Cuál es el tiempo de ejecución **O-grande** de la función **`construirGrafo`**?
14. Utilizando el ejercicio 8, implementar el método **`decrementarClave`** en la clase **`MonticuloBinario`** para que funcione el algoritmo de **Dijkstra**. Muestra cada paso al aplicar el algoritmo de **Dijkstra** al grafo resultante del problema 11.

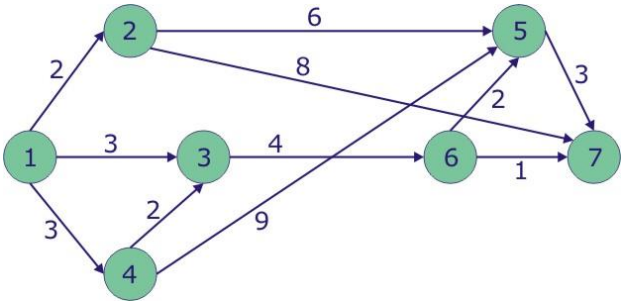
15. Dado el grafo dirigido y valorado de la figura, encuentra el camino más corto desde el vértice **A** a todos los demás vértices usando el algoritmo de **Dijkstra**. Describe el proceso paso a paso.



16. Usando el algoritmo de **Prim**, encuentra el árbol de expansión de ponderación mínima
17. ¿Cuál es el tiempo de ejecución **O-grande** para el algoritmo de **Prim** del árbol de expansión mínimo?
18. Dado el grafo de la figura, encuentra un árbol de expansión de coste mínimo describiendo el proceso paso a paso mediante el algoritmo de **Prim**



19. Modifica la función de **búsqueda en profundidad** para producir un **ordenamiento topológico**.
20. Dada la siguiente red, encontrar una ordenación topológica, explicando el proceso paso a paso.



21. Escribe el método **`transponer`** para la clase **`Grafo`**.
22. Utilizando la **búsqueda en anchura**, escribe un algoritmo que puede determinar la ruta más corta de cada vértice a cada uno de los otros vértices. Esto se llama el ***problema de la ruta más corta de todas las parejas.***