**UNIVERSIDAD NACIONAL**

**DE**

**SAN AGUSTÍN**

**FACULTAD DE PRODUCCIÒN Y SERVICIOS**

**ESCUELA:**

**INGENIERÌA DE SISTEMAS**

**CURSO:**

**ESTRUCTURA DE DATOS Y ALGORITMOS**

**“INFORME laboratorio 9 grafos”**

**ALUMNO:**

**Aroldo Muñoz Romani.**

**DOCENTE:**

**Mgt. Edith Pamela Rivero Tupac**

**AREQUIPA-PERÚ**

**2021**

**EJERCICIOS PROPUESTOS**

1. Crear un repositorio en GitHub, donde incluyan la resolucion de los ejercicios propuestos y el informe.

<https://github.com/AroldoMunoz/9_eda_grafos>

2. Implementar el cogido de Grafo cuya representacion sea realizada mediante LISTA DE ADYACENCIA.

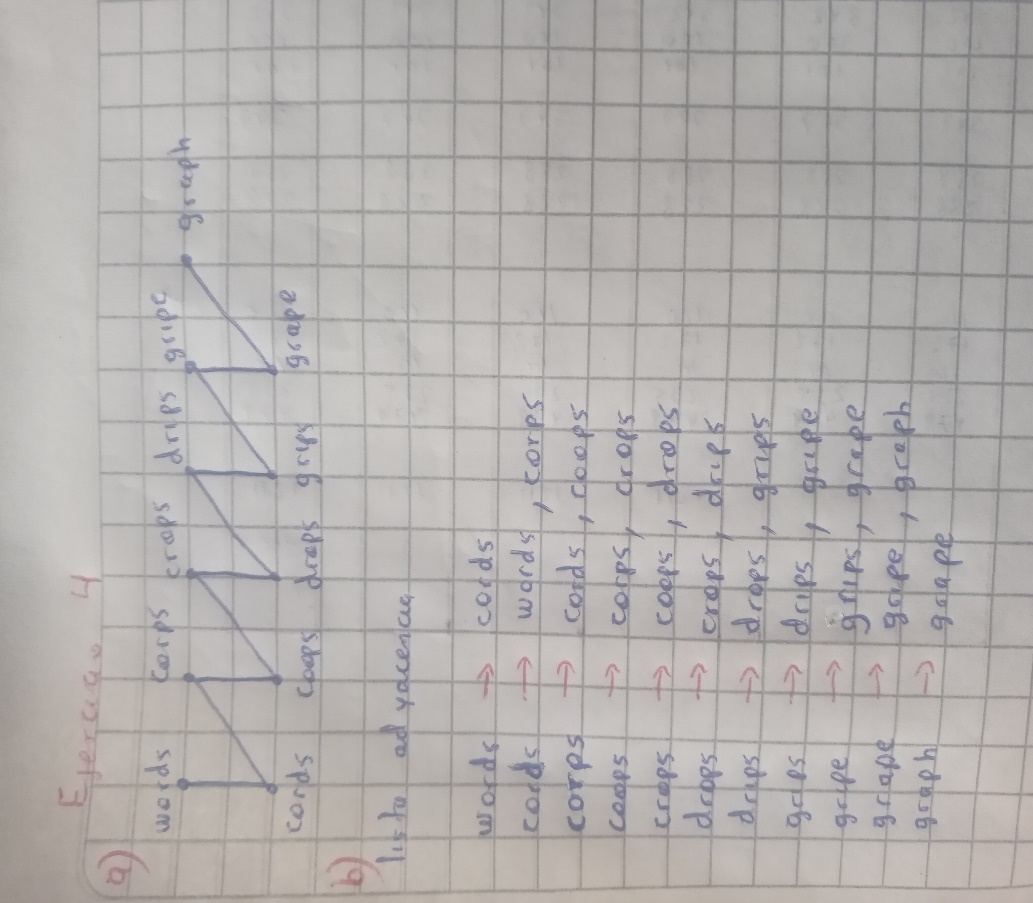
<https://github.com/AroldoMunoz/9_eda_grafos/blob/main/ejercicio2/test.java>

3. Implementar BSF, DFS y Dijkstra con sus respectivos casos de prueba.

<https://github.com/AroldoMunoz/9_eda_grafos/tree/main/ejercicio3>

4. Solucionar el siguiente ejercicio: (5 puntos)

El grafo de palabras se define de la siguiente manera: cada vértice es una palabra en el idioma Inglés y dos palabras son adyacentes si difieren exactamente en una posición. Por ejemplo, las **cords** y los **corps** son adyacentes, mientras que los **corps** y **crops** no lo son. a) Dibuje el grafo definido por las siguientes palabras: words cords corps coops crops drops drips grips gripe grape graph b) Mostrar la lista de adyacencia del grafo.



5. Realizar un metodo en la clase Grafo. Este metodo permitira saber si un grafo esta incluido en otro. Los parametros de entrada son 2 grafos y la salida del metodo es true si hay inclusion y false el caso contrario. (4 puntos)

Se hace búsqueda de todos los vértices de la lista

**CUESTIONARIO**

1. ¿Cuantas variantes del algoritmo de Dijkstra hay y cuál es la diferencia entre ellas?

Dijkstra sin cola de prioridad de acuerdo a los vértices se da la prioridad al ultimo nodo visitado asi que el árbol sera muy largo

Dijkstra con cola de prioridad, da prioridad al nodo mas viejo en registrarse, asi genera un árbol mas ramificado

2. Invetigue sobre los ALGORITMOS DE CAMINOS MINIMOS e indique, ¿Qué similitudes encuentra, qué diferencias, en qué casos utilizar y porque?

**Los algoritmos de caminos mínimos**

El problema del camino más corto es el aquel que consiste en encontrar un camino entre dos nodos de manera que la suma de los costes de los nodos que lo constituyen es mínima. Por supuesto, este tipo de algoritmos permiten estudiar costes de trayecto diferentes, como la distancia, el tiempo de viaje, el coste generalizado, etc. Y no sólo eso, sino que es posible trabajar con matrices de adyacencias donde no sólo se representen costes medidos sino también ponderados. Un ejemplo clásico en el mundo del transporte es la penalización de cada modo de que se emplea en la 3ª etapa de los modelos de demanda de transporte (recordemos que son: 1) Generación de viajes,  2) Distribución de viajes; 3) Selección modal y 4) Selección de ruta. Por eso, se suele hablar de «modelos de cuatro etapas»).

Los algoritmos más importantes para resolver este problema son:

Dijkstra: resuelve el problema de los caminos más cortos desde un único nodo origen hasta todos los otros nodos del grafo (aunque aplicando una regla de repetición del algoritmo, se puede automatizar la resolución del problema desde todos los nodos de origen hasta todos los nodos del grafo).

Bellman-Ford: resuelve el problema de los caminos más cortos desde un origen permitiendo que la ponderación de los nodos sea negativa.

Algoritmo de Floyd-Warshall: resuelve el problema de los caminos más cortos entre todos los nodos.

Algoritmo de Johnson: resuelve el problema de los caminos más cortos entre todos los nodos y puede ser más rápido que el de Floyd-Warshall en grafos de baja densidad.