1. **Propuesta de recorrido**

Para el recorrido de grafos se plantearon 2 posibles soluciones: Algoritmos DFS y BFS

**BFS:**

En el algoritmo BFS los nodos que son visitados más de una vez son eliminados de inmediato de la queue, lo cual permite ahorrar tiempo al acortar posibilidades.

Se sigue el estándar FIFO, el primero en entrar, el primero en salir, es decir, una cola

El BFS ocupa más espacio en memoria que el DFS

**DFS:**

En el algoritmo DFS se recorren todos los nodos, y se eliminan los ya visitados al final del proceso.

Se sigue el principio LIFO, primero en entrar, último en salir, es decir, una fila

El DFS ocupa considerablemente menos espacio en memoria frente al BFS

**ELECCIÓN:**

El BFS resalta para nuestra solución, pues para esto lo necesitamos, no para crear un mapeo de todas las aristas/conexiones que yacen de los nodos, sino simplemente buscar el camino más corto (en términos de nodo) entre dos vértices.

A pesar de que este ocupa algo más de espacio, hace un excelente trabajo en conjunto de los algoritmos de recubrimiento en árbol;

ya que no necesitamos una imagen del grafo (el cual ya nos proporciona este algoritmo en cierta forma), sino que necesitamos un segundo criterio para tomar decisiones sobre la dirección del flujo del agua, gracias a conocer todos los menores caminos posibles.

En cuanto a términos de implementación, se decidió usar una queue en base a una linked-list, ya que estas listas nos ahorran un pequeño espacio en memoria, al guardarse cada nodo conexo como un atributo (ocupando solo un espacio en memoria, que es el otro objeto) y no como un slot de un arraylist + el tamaño del objeto mismo.