

**Programación III**

TPO

**Alumnos**:

Shevchuk Calo, Miguel Omar LU 1116936

Christian Casal LU 126510

Carlos Huerta LU 1133112

# Índice

[Índice 2](#_Toc108115579)

[Consignas 2](#_Toc108115580)

[Desarrollo 3](#_Toc108115581)

[**Algoritmos elegidos** 3](#_Toc108115582)

[**Datos de Prueba** 3](#_Toc108115583)

# Consignas

**Fecha de Entrega**: 07/07/2022 por correo electrónico a guirodriguez@uade.edu.ar. El entregable consiste en la entrega del link del repositorio GitHub con el proyecto Java y un informe describiendo cada una de las etapas del proyecto. El trabajo se puede realizar en grupo de hasta 2 personas o individual.

**Enunciado**

Usted va a recibir la Clase Grafo en Java y deberá implementar 2 (dos) de los siguientes algoritmos:

• Algoritmo Breadth-First Search (BFS)

• Algoritmo Depth-First Search (DFS)

• Algoritmo de Prim

• Algoritmo de Kruskal

• Algoritmo de Dijsktra

• Algoritmo de Floyd

Tenga en cuenta la representación del grafo (dinámica o estática) para la implementación de los algoritmos elegidos.

**Nota**: Todos los archivos de configuración, ejemplos de entrada y cualquier otro recurso deben ser anexados al proyecto dentro de una carpeta /resources

# Desarrollo

Repositorio: <https://github.com/miguelshevchuk/grafos-Progra3>

## **Algoritmos elegidos**

Para el desarrollo del TPO seleccionamos e implementamos los siguientes algoritmos:

1. BFS (Breath First Search)
2. Dijkstra

## **Datos de Prueba**

Los algoritmos desarrollados se probaron con el siguiente grafo no dirigido:

Imagen en blanco y negro

Descripción generada automáticamente con confianza media

Para probarlo en el proyecto, se pueden ejecutar los Test Cases programados en la carpeta Test:

Captura de pantalla de un celular

Descripción generada automáticamente

Los algoritmos imprimen por pantalla los resultados obtenidos de la ejecución según:

Resultados obtenidos BFS:

Captura de pantalla de computadora

Descripción generada automáticamente con confianza media

[Arista(padre=null, hijo=1), Arista(padre=1, hijo=2), Arista(padre=1, hijo=5), Arista(padre=2, hijo=3), Arista(padre=3, hijo=4)]

Resultados obtenidos Dijkstra:

Texto

Descripción generada automáticamente

{1=Dijkstra.NodoDistancia(verticePadre=null, distanciaTotal=0, estado=PERMANENTE), 2=Dijkstra.NodoDistancia(verticePadre=1, distanciaTotal=5, estado=PERMANENTE), 3=Dijkstra.NodoDistancia(verticePadre=2, distanciaTotal=7, estado=PERMANENTE), 4=Dijkstra.NodoDistancia(verticePadre=3, distanciaTotal=10, estado=PERMANENTE), 5=Dijkstra.NodoDistancia(verticePadre=2, distanciaTotal=7, estado=PROVISORIO)}