**Universidad Peruana de Ciencias Aplicadas**



**HITO #1**

CURSO DE COMPLEJIDAD ALGORÍTMICA - CC184

Carrera de Ciencias de la Computación e Ing. Software

**SECCIÓN:** CC45

**DOCENTE:**

Patricia Daniela Reyes Silva

**INTEGRANTES:**

Alfaro Coveñas, Louis Piero U20191B299

Cisneros Tafur, Diego Rafael U20221A715

Ruiz Ramirez, Joaquin Sebastian U20201F678

**CICLO:**

2023-02

**GRUPO:**

4

**09/2023**

**ÍNDICE:**

[**1.** **Descripción del problema** 3](#_Toc150111289)

[1.1. Problema: 3](#_Toc150111290)

[1.2. Objetivos: 3](#_Toc150111291)

[*1.2.1.* Objetivo general: 3](#_Toc150111292)

[*1.2.2.* Objetivos específicos: 3](#_Toc150111293)

[**2.** **Descripción del conjunto de datos (Dataset)** 5](#_Toc150111294)

[2.1. Descripción: 5](#_Toc150111295)

[2.2. Visualización de muestra de datos: 8](#_Toc150111296)

[**3.** **Propuesta:** 9](#_Toc150111297)

[*3.1.* *Tecnica y Metodologia* 10](#_Toc150111298)

[*3.1.1.* *Grafo no dirigido:* 10](#_Toc150111299)

[*3.1.2.* *BFS:* 10](#_Toc150111300)

[*3.1.3.* *DFS:* 10](#_Toc150111301)

[*3.1.4.* *Kruskal:* 11](#_Toc150111302)

[*3.1.5.* *Árbol de expansión mínima:* 11](#_Toc150111303)

[**4.** **Diseño del Aplicativo:** 12](#_Toc150111304)

[*4.1.* *Análisis del sistema:* 12](#_Toc150111305)

[*4.1.1.* *Análisis de Requisitos:* 12](#_Toc150111306)

[*4.1.2.* *Análisis del algoritmo (Big O):* 12](#_Toc150111307)

[*4.2.* *Diseño del sistema:* 13](#_Toc150111308)

[*4.2.1.* *Diseño de Interfaz:* 13](#_Toc150111309)

[*4.2.2.* *Importación de Librerías:* 14](#_Toc150111310)

[*4.2.3.* *Recorrido del algoritmo:* 15](#_Toc150111311)

[**5.** **Bibliografía:** 17](#_Toc150111312)

# **Descripción del problema**

## Problema:

La demanda del anime ha ido en crecimiento con la popularización de series dentro del público occidental. Esto es un fenómeno que se ha hecho bastante notable durante los últimos años con la normalización del fanatismo hacía este “género” de la animación y muchos medios de noticias han hecho notar esto, tales como *enter.co* (2021) y *somoskudasai* (2022).

En Japón, el anime es parte de su propia cultura, teniendo constantes olas de popularidad a través de distintas generaciones a lo largo de la historia de la animación. Prueba de esto es la abrumante cantidad de animes que se producen cada año. A diferencia de la animación occidental, el anime ha tenido un público objetivo más diverso, por lo que la producción de animes ha sido más amplia. Además de esto, la corta duración de muchos animes facilita la producción casi masiva de estos. Es por ello que cada año se ven tantas nuevas producciones como algo normal.

Desafortunadamente, esta exorbitante cantidad de producciones puede presentarse como un factor negativo para aquellos que buscan empezar a introducirse en este interés. Empezar a ver un nuevo anime puede ser complicado cuando la lista en sí es increíblemente extensa y utiliza términos que para el espectador principiante pueden ser confusos (shonen, shojo, etc.)

Ante este contexto, es necesario la implementación de un sistema que catalogue de manera adecuada y en base a distintos factores como el género, la productora, las opiniones de usuarios y demás, a una amplia lista de animes para poder así encontrar relaciones sustanciales entre estos y poder recomendar directamente una producción específica.

Para abordar este problema, se pueden utilizar datos reales de las producciones anuales de animes, al cual se les pueda aplicar un algoritmo para identificar un recorrido mínimo que represente qué tanta relación guardan entre sí. Cada anime sería considerado como nodo y la relación que guardan entre ellos sería considerada como arista.

## Objetivos:

### Objetivo general:

Generar un sistema de recomendaciones de animes en base al perfil del usuario, usando teoría de grafos y el algoritmo de Kruskal.

### Objetivos específicos:

OE1: Construir un grafo utilizando un dataset real sobre una lista extensa de animes.

OE2: Aplicar el algoritmo de Kruskal para encontrar el árbol de expansión mínima que permita conectar a todos los nodos utilizando el menor recorrido posible.

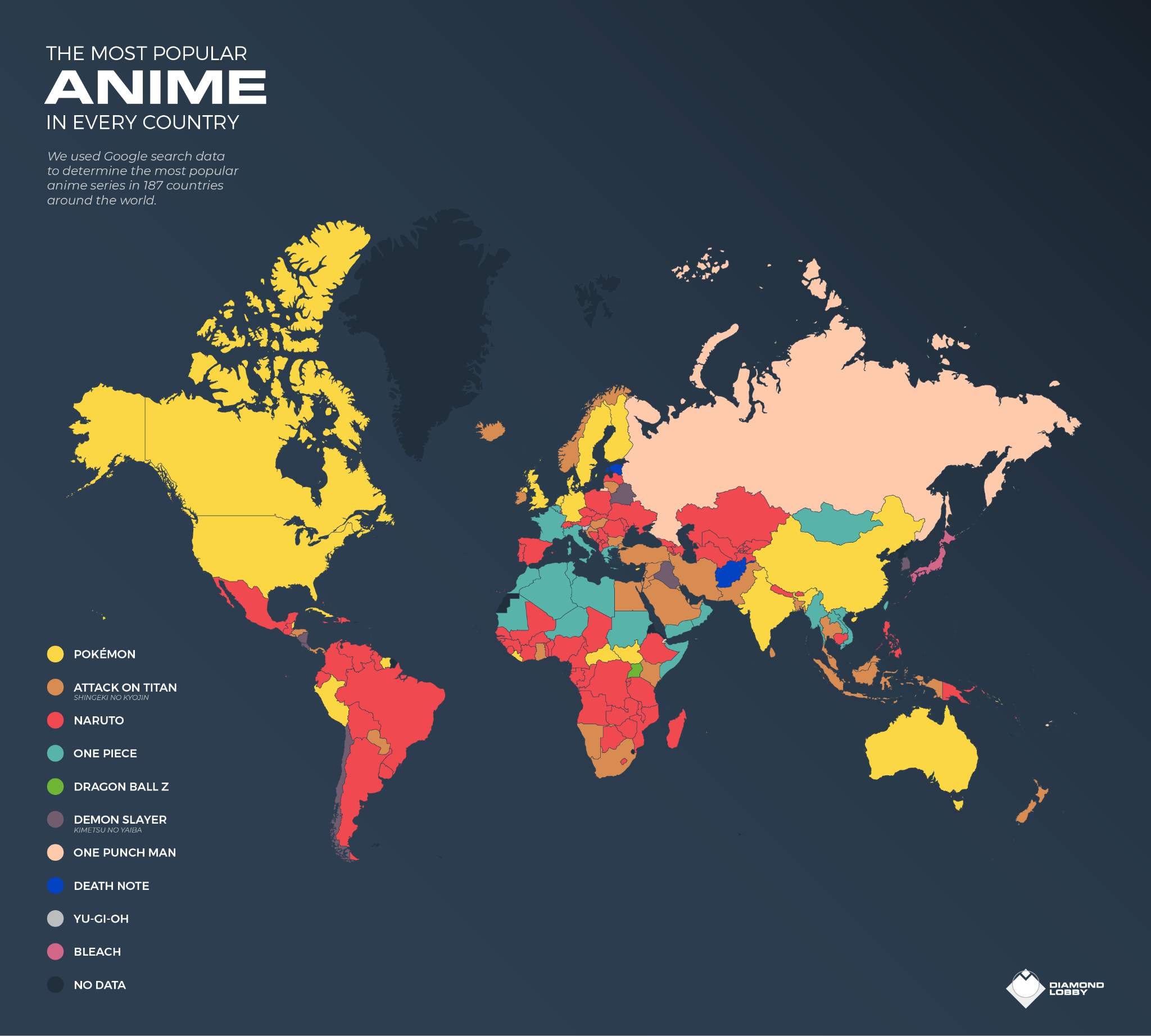
# **Descripción del conjunto de datos (Dataset)**

## Descripción:

Este conjunto de datos contiene información sobre las preferencias de los usuarios de 73.516 usuarios en 24.906 animes. El dataset cuenta con 24 variables

* anime\_id : Identificador único para cada registro en el conjunto de datos
* Name: Nombre del anime en su idioma original
* English name: Nombre del anime en inglés
* Other name: Nombre del anime en Kanji
* Score: Puntuación brindada por espectadores o críticos
* Genres: Los géneros existentes de todos los animes
* Synopsis: Esquema o exposición gráfica de los puntos generales del anime.
* Type: Tipo de transmisión del anime.
* Episodes: Los episodios disponibles del anime.
* Aired: Fecha de la primera emisión del capítulo del anime.
* Premiered: Fecha de estreno del anime.
* Status: Estado del anime (emisión o finalizada)
* Producers: Productora de animación que fue responsable de la creación del anime
* Licensors: Licenciantes que fueron responsables del anime
* Studios: Estudio responsable
* Source: Fuente del anime
* Duration: Duración de cada capitulo del anime
* Rating: Clasificación del anime
* Rank: Rango del anime
* Popularity: Popularidad del anime
* Favorites: Favoritos de los consumidores
* Scored By: Puntuaciones del anime
* Members: Número de miembros
* anime\_url : Variable que almacena una URL o enlace web relacionado con el anime.

Todas estas interconexiones de géneros del anime obtenidos de la data set, donde hay 10 puntos de distintos colores, que son representadas en un mapa acerca de los animes más vistos en todo el mundo.



**Tabla 1**

*Muestra de dataset a utilizar en el proyecto*

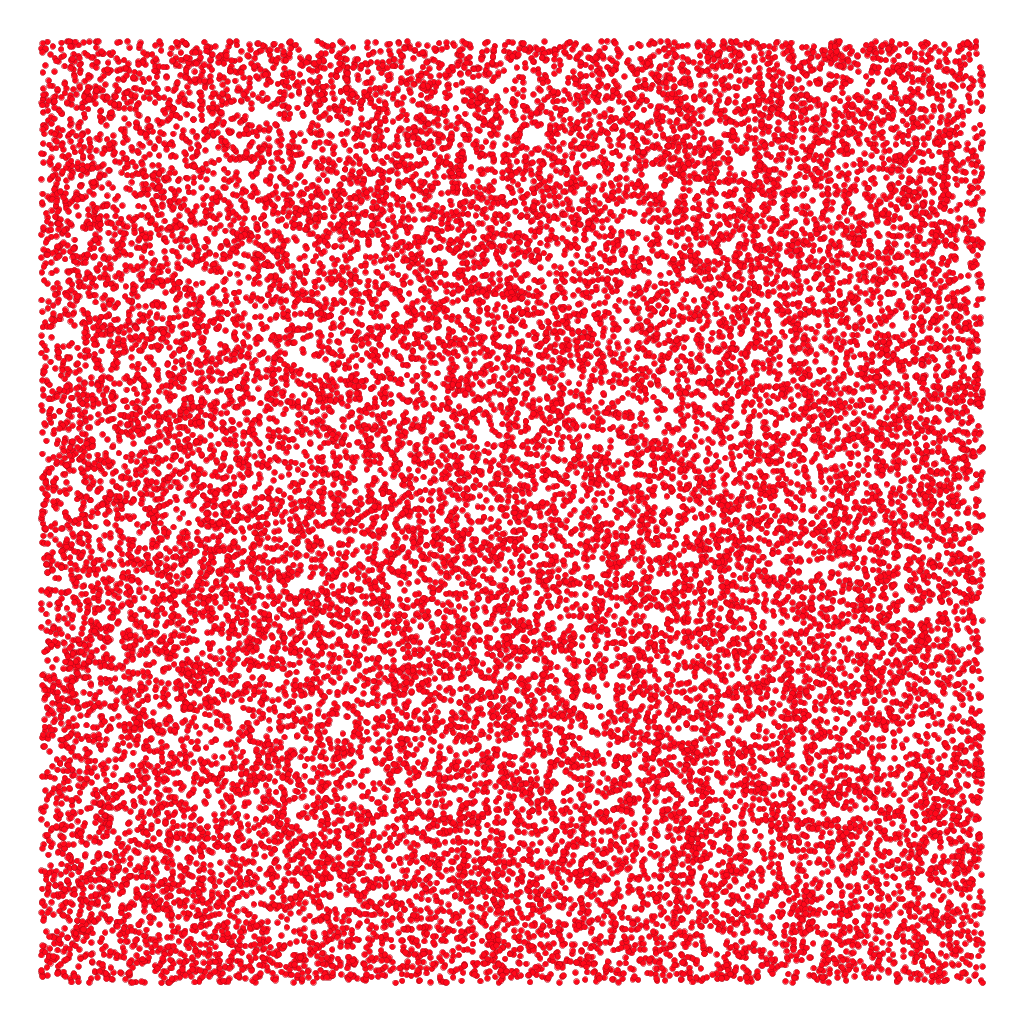
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| anime\_id | name | genre | type | episodes | rating |
| 5114 | Fullmetal Alchemist: Brotherhood | Action, Adventure, Drama, Fantasy, Magic, Military, Shounen | TV | 64 | 9.26 |
| 32935 | Haikyuu!!: Karasuno Koukou VS Shiratorizawa Gakuen Koukou | Comedy, Drama, School, Shounen, Sports | TV | 10 | 9.15 |
| 11061 | Hunter x Hunter (2011) | Action, Adventure, Shounen, Super Power | TV | 148 | 9.13 |
| 28171 | Shokugeki no Souma | Ecchi, School, Shounen | TV | 24 | 8.61 |
| 30276 | One Punch Man | Action, Comedy, Parody, Sci-Fi, Seinen, Super Power, Supernatural | TV | 12 | 8.82 |

Fuente: kaggle.com

La representación gráfica para nuestro análisis es un grafo no dirigido en el que los nodos representan las interconexiones de género de los animes y las conexiones entre ellas son las aristas, el peso de cada arista representa la distancia entre los nodos.

## 

## Visualización de muestra de datos:



# 

# **Propuesta:**

Para realizar nuestro proyecto, se utilizarán algoritmos de Kruskal y los de BFS y DFS. Los cuales nos permitirá conocer los vecinos de nuestro nodo seleccionado y podremos filtrar para obtener las recomendaciones, Kruskal se usará para encontrar la mejor recomendación entre dos nodos o en este caso dos animes.

El primer paso con el que nos podremos a trabajar será el de determinar el peso de las conexiones de nuestros nodos, los cuales se delimitaran a través de una función la cual, analizará cada dato del dataset y utilizando los géneros, fechas de emisión, rating, etc.

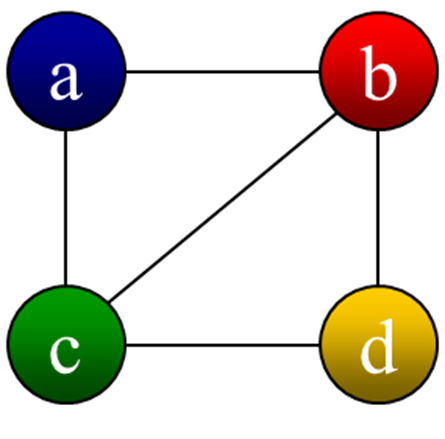
Luego del proceso anterior, guardaremos la información en un archivo para visualizar la información y una vista gráfica de los grafos gracias a algunas herramientas como Gephi, asimismo nos servirá para los siguientes pasos en nuestro proyecto.

Por consiguiente, luego de obtener toda la información necesaria, se creará una clase grafo la cual contendrá las conexiones y pesos. En esta clase contendrá los algoritmos necesarios para completar nuestra solución.

## 

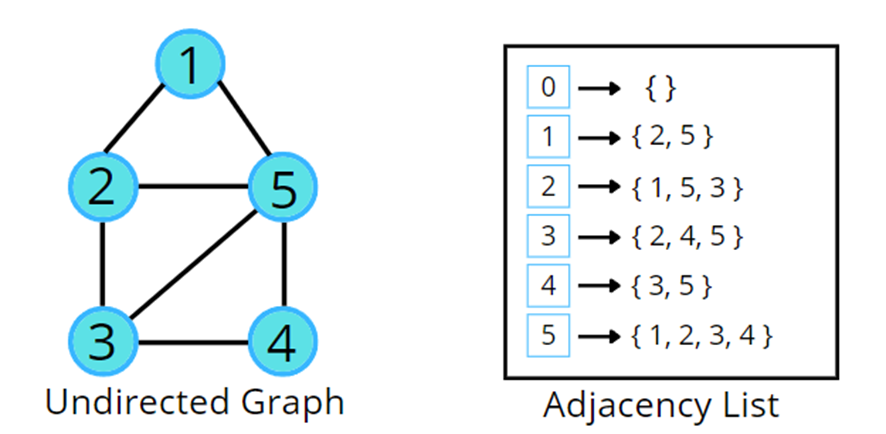
## *Tecnica y Metodologia*

### *Grafo no dirigido:*

Un grafo no dirigido es un conjunto de nodos conectados por aristas. Las aristas pueden tener un peso asociado, que representa la distancia o el costo de viajar entre dos nodos. (Cormen et al., 2022).

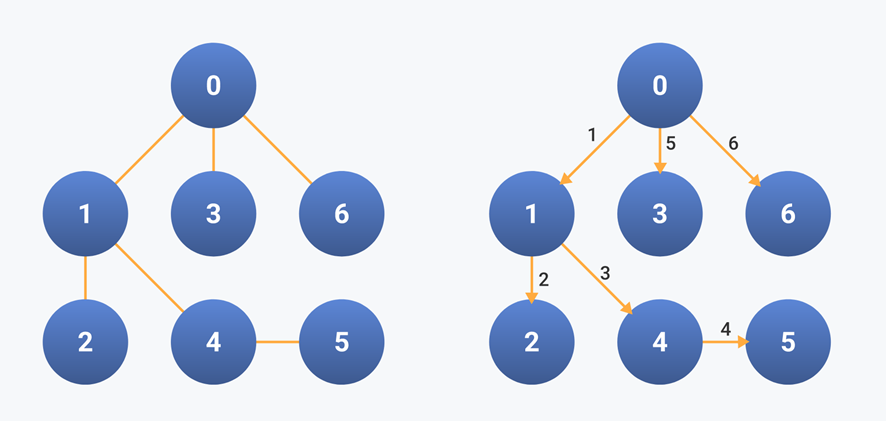
### *BFS:*

El algoritmo de búsqueda en amplitud (BFS) es un algoritmo de búsqueda que explora un grafo comenzando en un nodo inicial y explorando todos los nodos adyacentes antes de explorar los nodos adyacentes a los nodos adyacentes, y así sucesivamente. (Cormen et al., 2022).

****

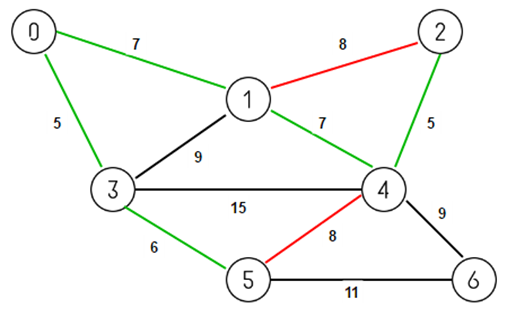
### *DFS:*

El algoritmo de búsqueda en profundidad (DFS) es un algoritmo de búsqueda que explora un grafo comenzando en un nodo inicial y explorando todos los nodos adyacentes antes de explorar los nodos adyacentes a los nodos no explorados, y así sucesivamente. (Cormen et al., 2022).



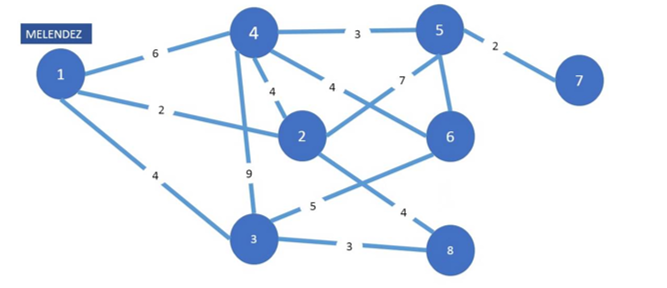
### *Kruskal:*

El algoritmo de Kruskal es un algoritmo para encontrar un árbol de expansión mínima (MST) en un grafo no dirigido. Un MST es un árbol que conecta todos los nodos del grafo con el menor costo posible. (Kruskal, 1956).



### *Árbol de expansión mínima:*

Un árbol de expansión mínima es aquel grafo que resulta de hallar el camino mínimo que une todos los nodos de un árbol de expansión (Wu y Chao, 2004).



# **Diseño del Aplicativo:**

En esta sección se describe el diseño del aplicativo desarrollado para resolver el problema planteado.

## *Análisis del sistema:*

### *Análisis de Requisitos:*

Se definieron las siguientes historias de usuario para reconocer los requisitos del proyecto:

* US01: Como usuario del sistema, quiero recibir una lista de recomendados en base a mi búsqueda.
* US02: Como usuario del sistema, quiero filtrar búsquedas que no sean de mi agrado
* US03: Como usuario del sistema, quiero utilizar un interfaz intuitivo y de características reconocibles.

### *Análisis del algoritmo (Big O):*

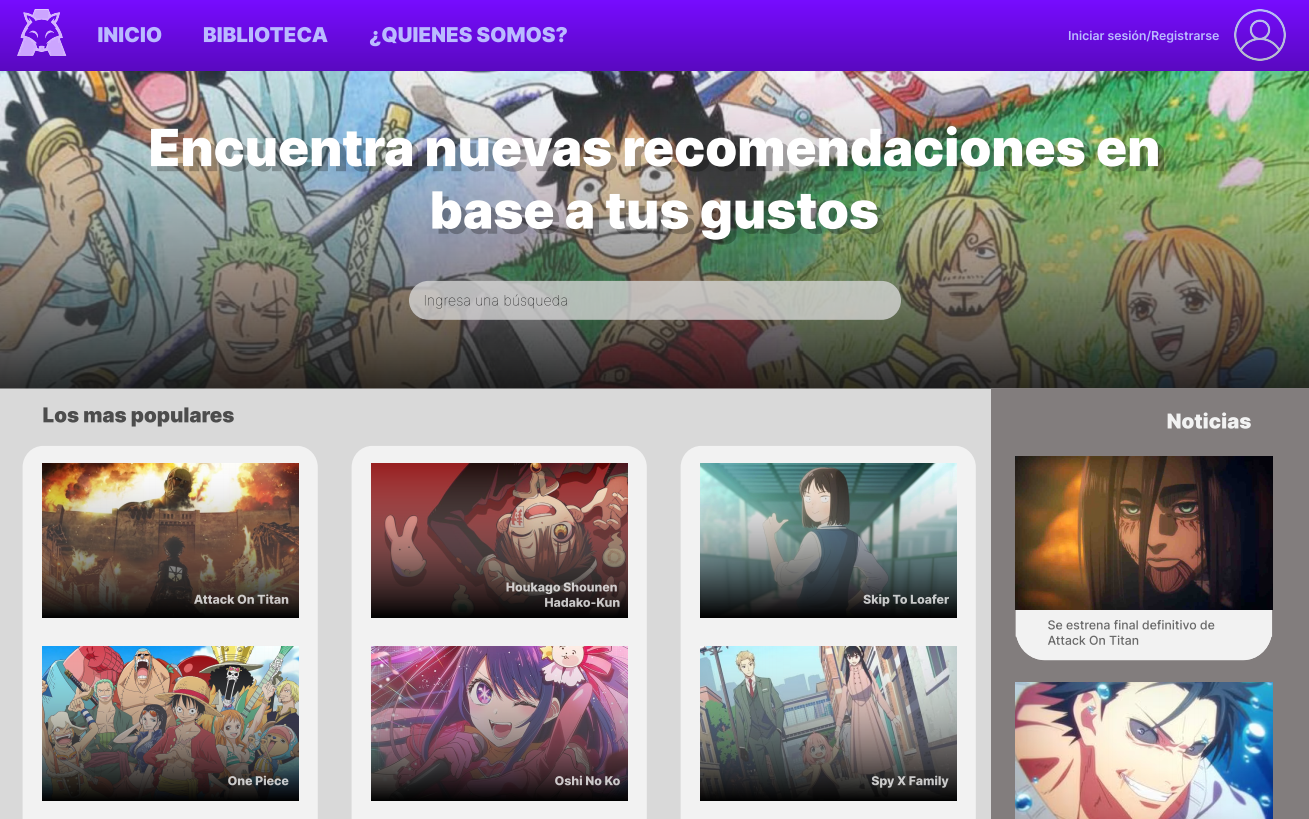
Para encontrar el árbol de expansión mínima en el conjunto de datos que se utilizaran para este proyecto, se implementara el algoritmo de Kruskal. En base a esto podemos calcular su complejidad de la siguiente manera

El paso inicial, y lo más esencial para el funcionamiento del mismo algoritmo, es ordenar las aristas en base a su distancia entre sí. Esto se realiza mediante el método “sort” en Python. La característica más importante de este método es que se basa en el algoritmo de ordenamiento “Timsort” al ser implementado de manera estándar en Python. Este, al ser un algoritmo bastante poderoso, es bastante relevante para el uso en este proyecto; sin embargo, su complejidad es de “O (n log n)”. Al ser comparado con los demás segmentos que conforman el código final, podemos asegurar que este ordenamiento vendría a dominar la complejidad del algoritmo de Kruskal; por lo tanto, la complejidad final vendría a ser de “O (n log n)”.

## *Diseño del sistema:*

### *Diseño de Interfaz:*

El diseño del interfaz a continuación es una proyección del modelo deseado al que se desea alcanzar, realizado con la herramienta Figma.



Figma: <https://www.figma.com/file/TY62jnXdGJgOR9JLl6zuzY/Proyecto-Web?type=design&node-id=0%3A1&mode=design&t=rMe4RyqBKIu05HRB-1>

La característica principal que busca tener este interfaz es el de presentar sus funciones de tal manera que el usuario las visualice de una manera intuitiva. Reconocer el propósito del proyecto y sus cualidades es esencial para la utilidad que le pueda dar al usuario.

Asimismo, en esta versión modelo, se busca también mantener un interfaz que genere confianza a través de su visualización

### *Importación de Librerías:*

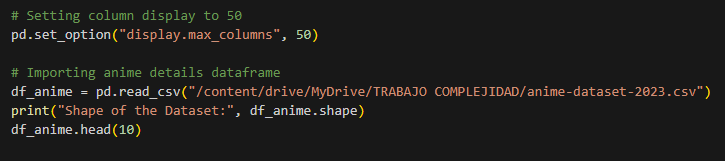
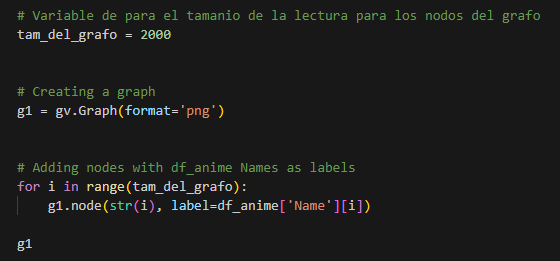
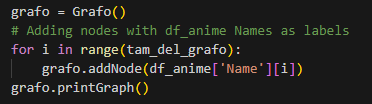
A continuación, se mostrarán las librerías utilizadas para la realización del algoritmo, junto a su propósito.

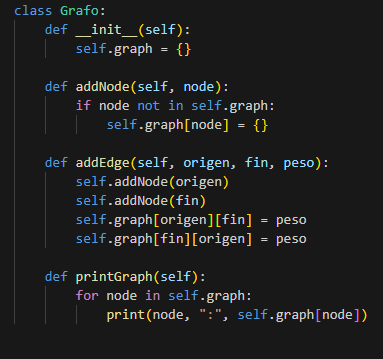
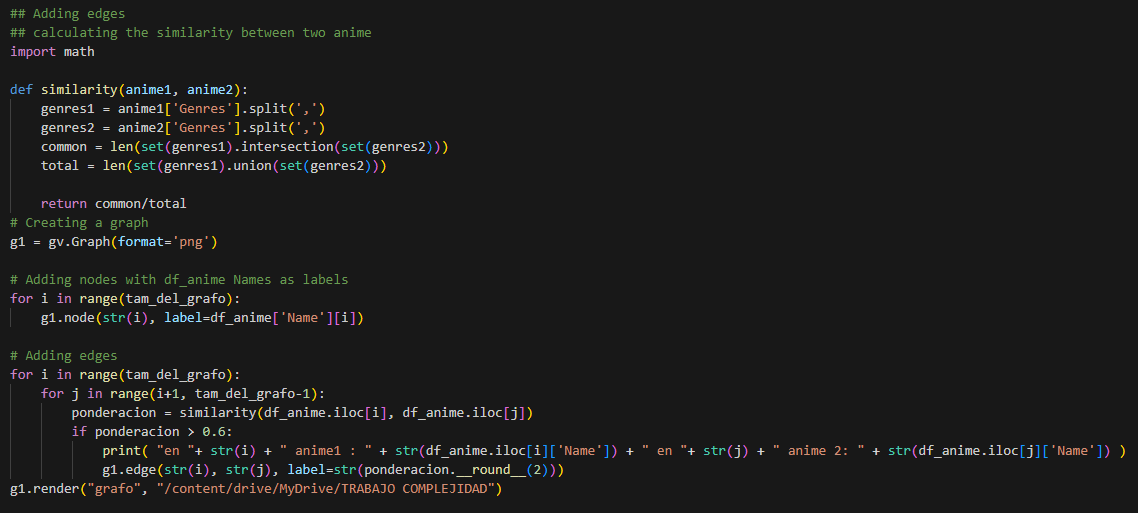
* **import numpy:** Proporciona funciones útiles para el análisis de grandes cantidades de datos. Permite el manipular, analizar, visualizar y procesar grandes matrices de datos.
* **import pandas:** Permite trabajar con datos en diferentes formatos (tabulaciones, categorias), así como su ordenamiento y manipulación.
* **import graphviz:** Permite el uso y creación de grafos.
* **import math:** Extiende la lista de funciones matemáticas que se pueden usar para el algoritmo.

****

### *Recorrido del algoritmo:*

Esta sección estará dedicada a detallar las partes del código, así como las funciones que cumplen.

* Para empezar, se importaron los datos que se van a utilizar para el proyecto en el programa, así como un entorno en el cual se puedan visualizar
* En segundo lugar, se genera un entorno en el cual se pueda realizar un grafo donde se visualicen las relaciones que tienen las diferentes aristas entre si a través de los nodos
* De igual forma, se crea la clase grafo en la cual se implementarán los nodos que tendrán las aristas entre sí.
* Posteriormente, se calcula el peso de los nodos que existen entre las aristas para definir el peso que tienen estos, y de esta manera hallar la similitud que tienen los datos (la lista de animes) entre sí.



# **Bibliografía:**

* Enter.co (2021) *Se ha doblado el consumo de anime en los últimos 4 años.* Enter.co. Recuperado el 22 de septiembre de 2023. <https://www.enter.co/cultura-digital/entretenimiento/se-ha-doblado-el-consumo-de-anime-en-los-ultimos-4-anos/>
* Somoskudasai (2022) *La demanda mundial por el contenido de anime se disparó en los últimos dos años.* SomosKudasai. Recupèrado el 22 de septiembre de 2023. <https://somoskudasai.com/noticias/cultura-otaku/la-demanda-mundial-por-el-contenido-de-anime-se-disparo-en-los-ultimos-dos-anos/>
* Diamond Lobby (2023) *What 's the most popular anime around the world. Diamond Lobby.* Recuperado el 22 de septiembre de 2023. <https://diamondlobby.com/geeky-stuff/most-popular-anime/>
* Cormen, T. C., Leiserson, C. L., Riverst, R. R., & Stein, C. S. (2022). *Introduction to Algorithms* (4.a ed.). <https://dl.ebooksworld.ir/books/Introduction.to.Algorithms.4th.Leiserson.Stein.Rivest.Cormen.MIT.Press.9780262046305.EBooksWorld.ir.pdf>
* Kruskal, J. B. (1956). On the shortest spanning subtree of a graph and the traveling salesman problem. *Proceedings of the American Mathematical Society*, *7*(1), 48–50.
* Wu, B. y Chao, K. (2004). *Árboles de expansión y problemas de optimización*. Chapman y Hall/CRC.