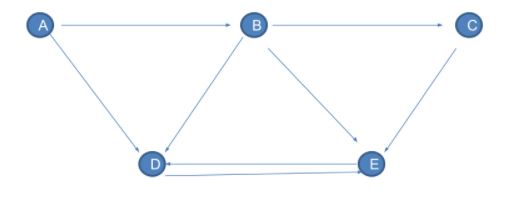
Estructura de Datos - División D  
Parcial 2 - Punto 1

Alumno: Teperino Leandro línea horizontal

# Grafos

Un **grafo** ​es un conjunto de objetos llamados vértices o nodos unidos por enlaces llamados aristas o arcos, que permiten representar relaciones binarias entre elementos de un conjunto.

Típicamente, un grafo se representa gráficamente como un conjunto de puntos (vértices o nodos) unidos por líneas (aristas).



Vértices = {A, B, C, D, E}

Aristas = {(A, B), (A, D), (B, C), (B, D), (B, E), (C, E), (D, E), (E, D)}

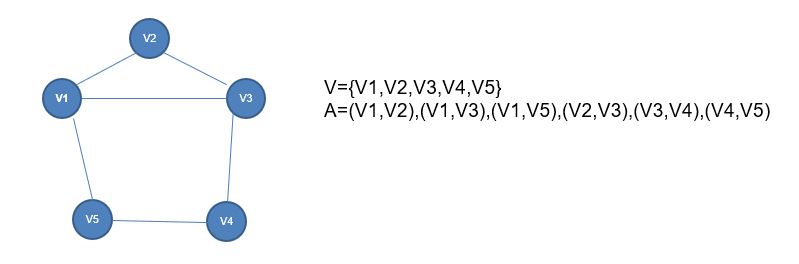
Las aristas se definen como el par de vértices que conectan y pueden tener asociado un valor el cual representa el peso o dificultad para desplazarse de un vértice a otro.

Desde un punto de vista práctico, los grafos permiten estudiar las interrelaciones entre unidades que interactúan unas con otras. Por ejemplo, una red [de](https://es.wikipedia.org/wiki/Red_de_computadoras) computadoras puede representarse y estudiarse mediante un grafo, en el cual los vértices representan terminales y las aristas representan conexiones (las cuales, a su vez, pueden ser cables o conexionesinalámbricas).

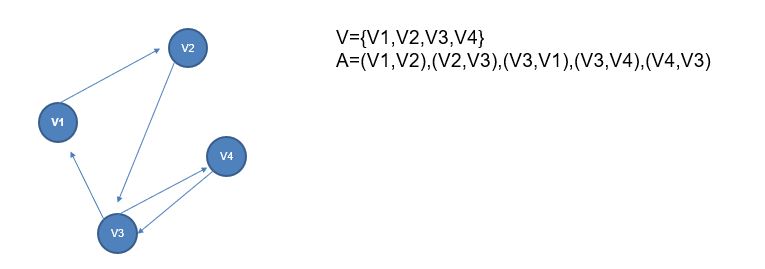
# 

# Tipos de Grafos

* **Grafos No Dirigidos.** Son aquellos en los cuales las aristas no están orientadas (no se representan con flechas). Cada arista se representa entre paréntesis, separando sus vértices por comas, y teniendo en cuenta que ambos vértices son origen y destino a la vez: (Vi, Vj) = (Vj, Vi).



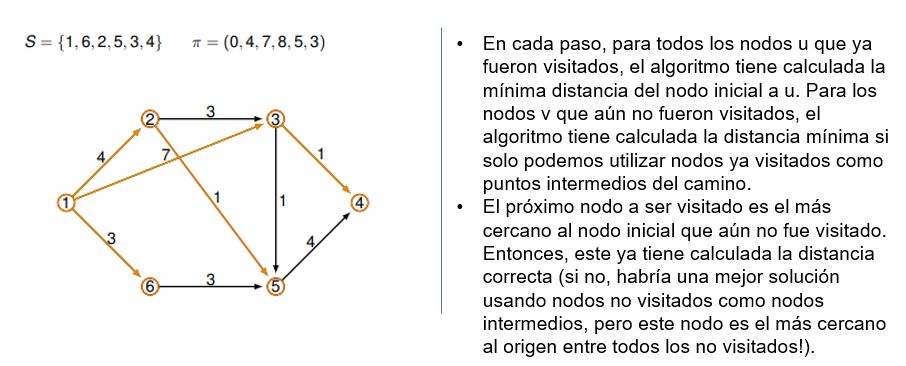
* **Grafos Dirigidos.** Son aquellos en los cuales las aristas están orientadas (se representan con flechas). Cada arista se representa entre paréntesis, separando sus vértices por comas.



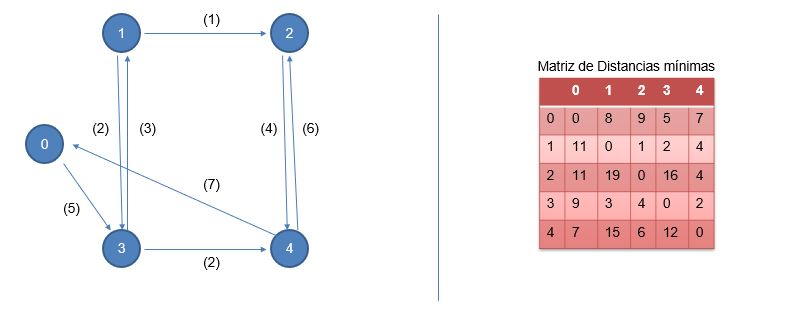
# Algoritmos aplicables a los Grafos

El **algoritmo de Dijkstra**, también llamado **algoritmo de caminos mínimos**, es un algoritmo para la determinación del camino más corto, dado un vértice origen, hacia el resto de los vértices en un grafo que tiene pesos en cada arista.

Sirve para cualquier grafo con pesos (dirigido o no) siempre y cuando sus pesos no sean negativos.



El **algoritmo de Floyd-Warshal,** el problema que intenta resolver este **algoritmo** es el de encontrar el camino más corto entre todos los pares de nodos o vértices de un grafo. Esto es semejante a construir una tabla con todas las distancias mínimas entre pares de ciudades de un mapa, indicando además la ruta a seguir para ir de la primera ciudad a la segunda. Este es uno de los problemas más interesantes que se pueden resolver con algoritmos de **grafos**.



# Algoritmos Lingüísticos

El propósito de un **algoritmo** **lingüístico** es agrupar un conjunto de palabras que son similares en su pronunciación.

En el trabajo **lingüístico**, se exponen las diferentes reglas que se han modelado para la obtención de palabras fonéticamente similares, y los resultados obtenidos a partir de ellas.

por ejemplo:

* Lopez o Lopes
* Viviana o Biviana o Bibiana
* Nuñez o Nunes
* Jessica o Yésica

En estos ejemplos si solo tenemos en cuenta la búsqueda por caracteres ASCII nos daría que son palabras totalmente distintas, sin embargo suenan igual y que hayan sido escritas distinto se puede deber a un error de tipeo. Esto es más común de lo que se cree. Para ayudarnos con este tipo de problemas existen los algoritmos fonéticos.

**Tipos de Algoritmos Linguisticos:**

1. **Soundex** es tal vez uno de los más conocidos, se basa en qué tan cerca están dos palabras dependiendo de su pronunciación. Uno de los inconvenientes es que Soundex está optimizado para el inglés y no funciona muy bien en español.
2. **Levenshtein** funciona distinto, mide la diferencia entre dos palabras escritas.
3. **Needleman-Wunsch:** El algoritmo Needleman-Wunsch es un algoritmo utilizado en bioinformática para alinear secuencias de proteínas o nucleótidos. Fue una de las primeras aplicaciones de programación dinámica para comparar secuencias biológicas. El algoritmo fue desarrollado por Saul B. Needleman y Christian D. Wunsch y publicado en 1970.
4. **Metaphone**: Es un algoritmo fonético, publicado por Lawrence Philips en 1990, para indexar palabras por su pronunciación en inglés. [1] Mejora fundamentalmente el algoritmo Soundex mediante el uso de información sobre variaciones e inconsistencias en la ortografía y pronunciación en inglés para producir una codificación más precisa, que hace un mejor trabajo al combinar palabras y nombres que suenan similares. Al igual que con Soundex, las palabras de sonido similar deberían compartir las mismas claves. Metaphone está disponible como operador integrado en varios sistemas.
5. **Double Metaphone:** Contrariamente al algoritmo original cuya aplicación se limita al inglés solamente, esta versión tiene en cuenta las peculiaridades ortográficas de varios otros idiomas.
6. **Metaphone 3:** Logra una precisión de aproximadamente el 99% para las palabras en inglés, palabras que no están en inglés familiares para los estadounidenses, y los nombres y apellidos que se encuentran comúnmente en los Estados Unidos, que se han desarrollado de acuerdo con según los estándares de ingeniería modernos contra un arnés de prueba de codificaciones correctas preparadas

# Expresiones Regulares

Una **expresión regular**,​ también conocidas como **regex** o **regexp**,​ por su contracción de las palabras inglesas ***reg****ular* ***ex****pression*, es una secuencia de caracteres que conforma un patrón de búsqueda. Se utilizan principalmente para la búsqueda de patrones de cadenas de caracteres u operaciones de sustituciones.

Las expresiones regulares son patrones utilizados para encontrar una determinada combinación de caracteres dentro de una cadena de texto. Las expresiones regulares proporcionan una manera muy flexible de buscar o reconocer cadenas de texto.

Ejemplo:

"<?(?<TagName>[a-zA-Z][\w**\r\n**]\*?) ?(?:(?<Attribute>[\w-**\r\n**]\*?)='?"?(?<Value>[\w-:;,\./= \r\n]\*?)'?"? ?)>"

Esta expresión permite encontrar el nombre de la etiqueta HTML , el nombre del atributo y su valor.