UNIDAD 4

INTRODUCCIÓN A GRAFOS

Mgtr. Vilma Duchi F.

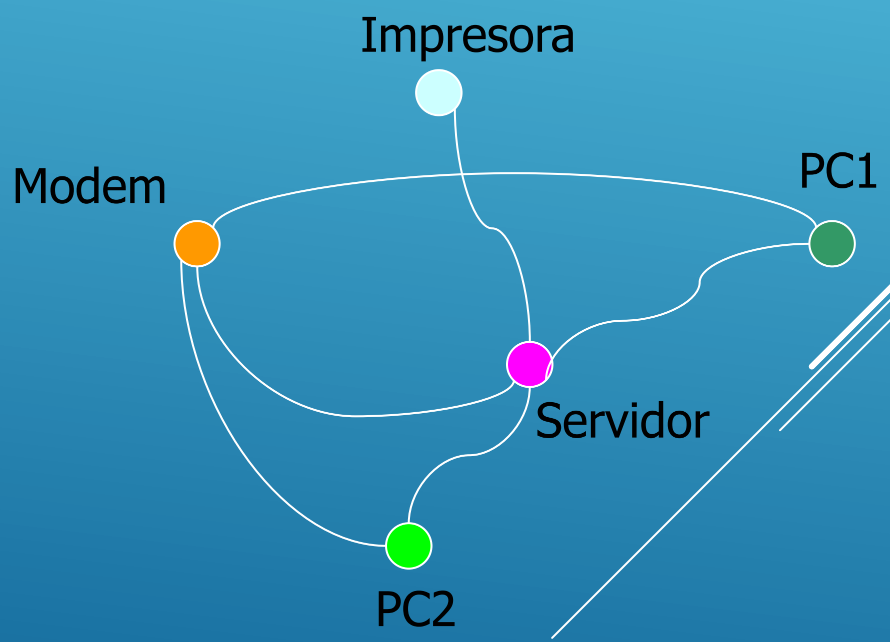
DEFINICIÓN

- El grafo no es más que un gráfico de datos representados por un conjunto de puntos llamados también nodos o vértices unidos por líneas, llamadas también arcos o aristas.
- Los grafos permiten estudiar interrelaciones entre elementos que interactúan unos con otros.

INTRODUCCIÓN

- Los grafos son estructuras de datos
- Representan relaciones entre objetos
  - Relaciones arbitrarias, es decir
  - No jerárquicas
- **Son aplicables en**
  - Química
  - Geografía
  - Ing. Eléctrica e Industrial, etc.
  - Modelado de Redes
    - De alcantarillado
    - Eléctricas
    - Etc.

Dado un escenario donde ciertos objetos se relacionan, se puede “modela el grafo” y luego aplicar algoritmos para resolver diversos problemas

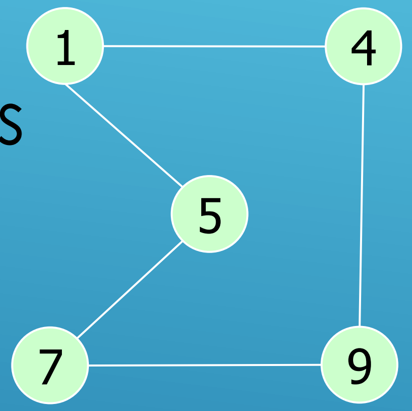


DEFINICIÓN

- Un grafo G = (V,A)
- V, el conjunto de vértices o nodos

  Representan los objetos
- A, el conjunto de arcos

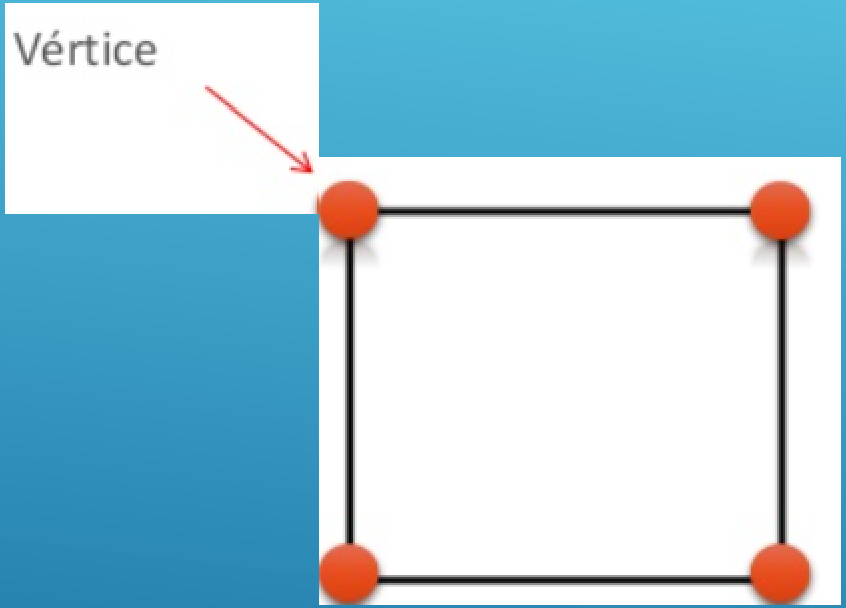
  Representan las relaciones

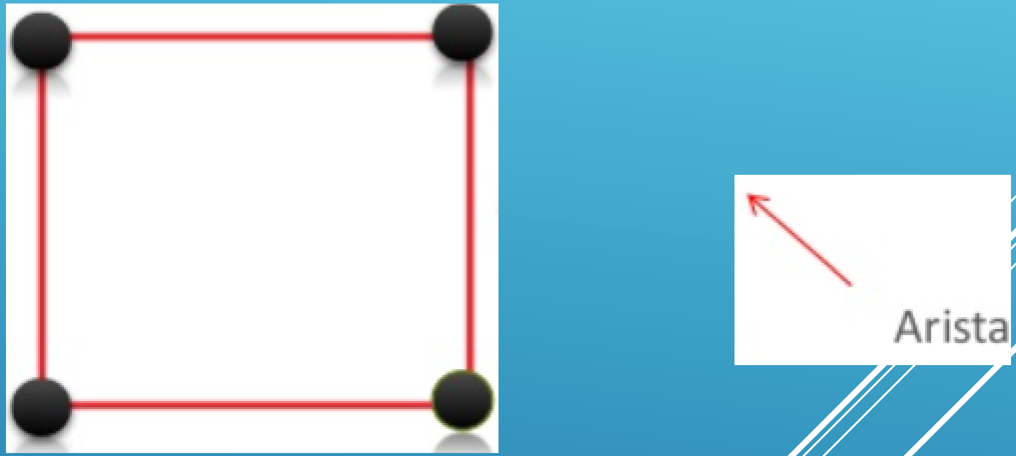


V = {1, 4, 5, 7, 9}

A= {(1,4), (5,1), (7,9), (7,5), (4,9), (4,1), (1,5), (9,7), (5, 7), (9,4)}

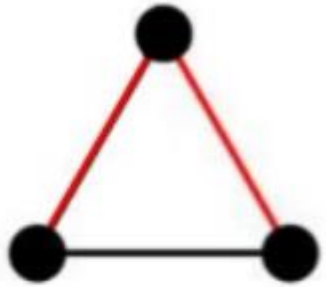
COMPONENTES DE UN GRAFO



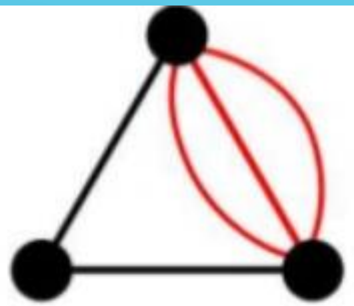


Típicamente, un grafo se representa gráficamente como un conjunto de puntos (vértices o nodos) unidos por líneas (aristas).

COMPONENTES DE UN GRAFO



Aristas adyacentes: dos aristas son adyacentes si tienen un vértice en común.



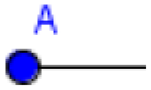
Aristas paralelas: Dos o más aristas que son incidentes (que se conectan) al menos dos vértices iguales.

Lazos: es un arista cuyos extr
vértice.

Vértices adyacentes: los vértices son adyacentes cuando comparten el mismo arista.

Arista Incidente: Una arista es incidente a un vértice si esta le une a otro vértice.

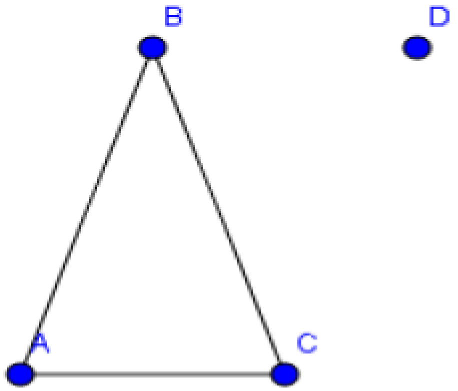
Ejemplo 4.



La arista f es in

Nodo aislado. - Es el nodo o vértice que no está conectado con ningún otro nodo por medio de una arista. Por ejemplo: D es el nodo aislado ya que no se conecta con ningún otro nodo.

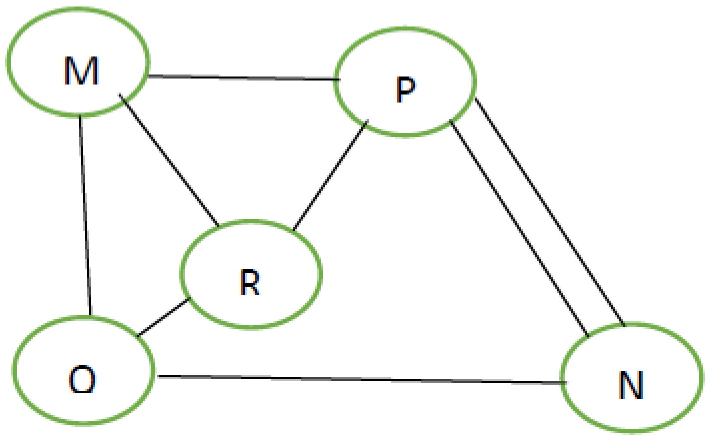
Ejemplo 5.



GRADO DE UN NODO

Es el número de arcos o aristas que inciden en un vértice.

EJERCICIO RESUELTO: Determine el grado de los vértices del siguiente grafo



Grado (M) = 3

Grado (P) = 4

Grado (R) = 3

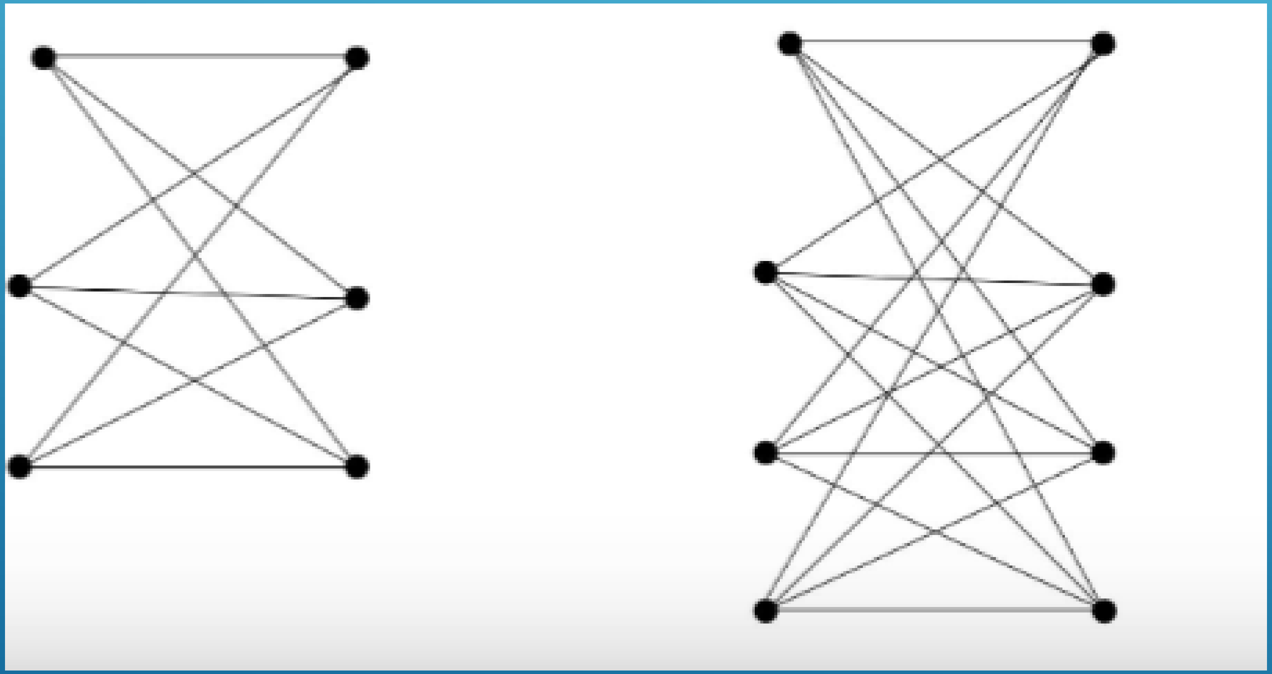
Grado (Q) = 3

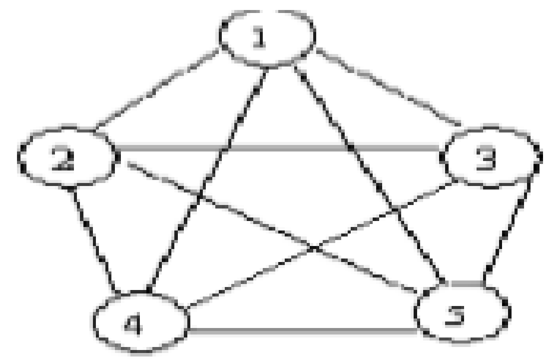
Grado (N) = 3

Grado del grafo = 16

ACTIVIDAD

Determine el grado de los siguientes grafos





G3

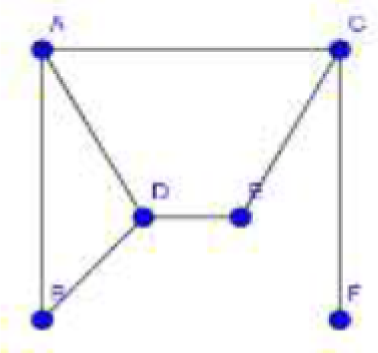
TEOREMA DE HANDSHAKING

Si $G = (V, E)$ es una gráfica no dirigida con aristas e, entonces $\sum_i grad(v_i) = 2e$

Es decir, la suma de los grados de todos los vértices de un grafo no dirigido es el doble del número de aristas y consecuentemente es par

EJERCICIO RESUELTO

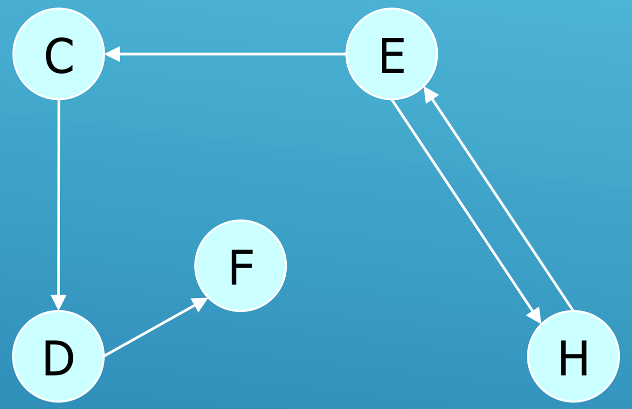
Determina el grado de los vértices o nodos y demuestre el teorema de handshaking.



$$\sum_i grad(v_i) = 2e$$

$$\text{Grad}(G) = 2 \cdot 7 = 14$$

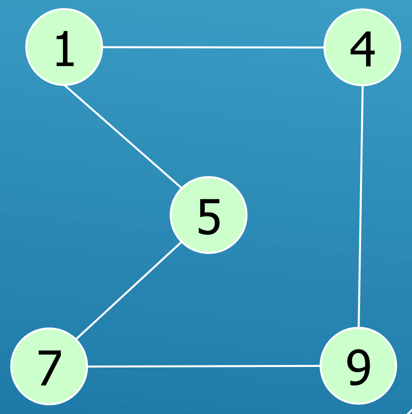
TIPOS DE GRAFOS



V = {C, D, E, F, H}

A= {(C,D), (D,F), (E,H), (H,E), (E,C)}

- Grafos dirigidos
  - Si los pares de nodos que forman arcos
  - Son ordenados. Ej.: (u->v)

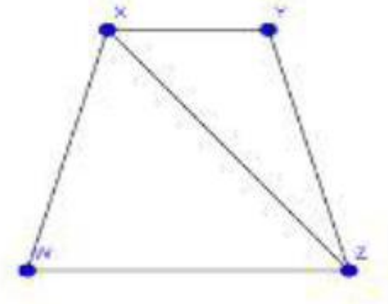


Grafo del ejemplo anterior

- Grafos no dirigidos
  - Si los pares de nodos de los arcos
  - No son ordenados Ej.: u-v

TIPOS SIMPLES

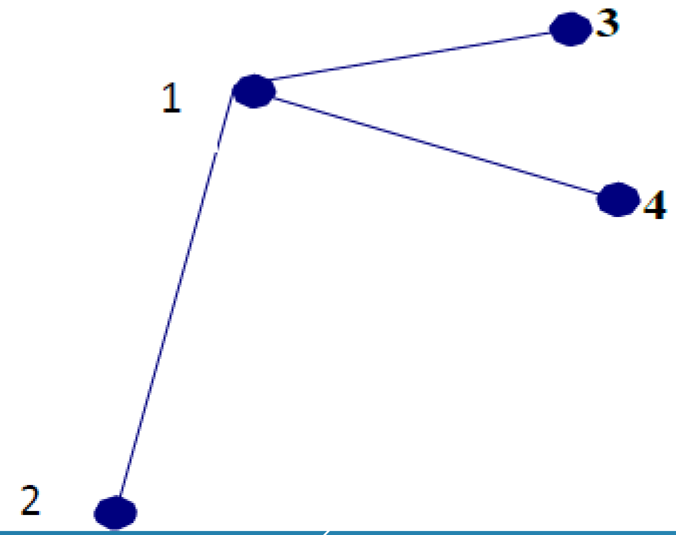
Es un grafo que a lo más solo una arista une dos vértices cualesquiera. Es decir, no tiene aristas múltiples o lazos.



Ejemplo: grafo simple es $V = 1,2,3,4$

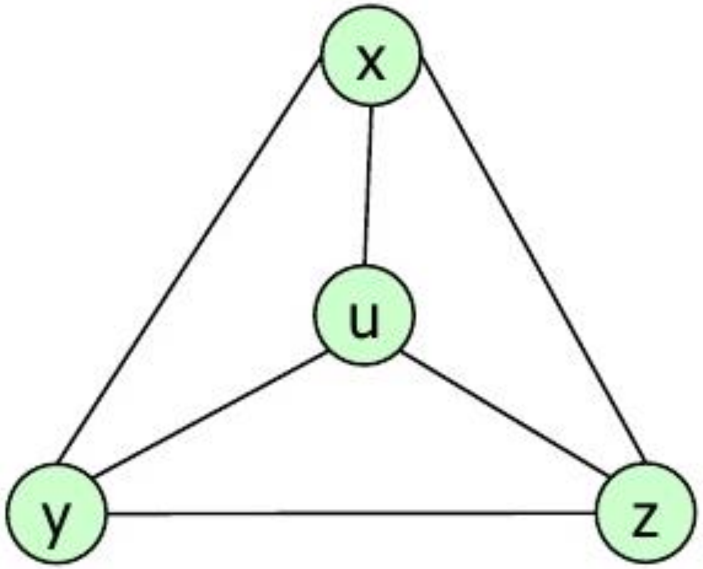
E={(1,2),(1,3), (1,4)}

Un diagrama de este grafo es como sigue:



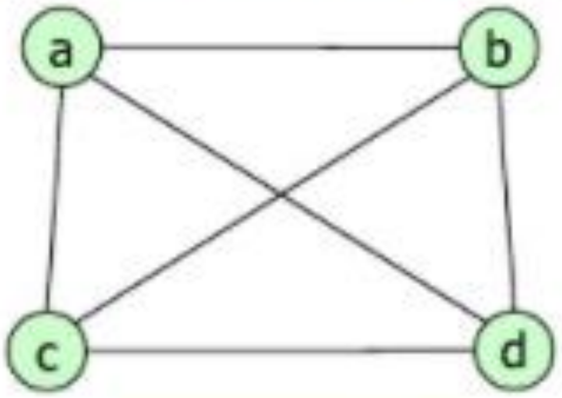
Tipos de Grafos

- Grafo **Regular**
  - Todos los vértices tienen el mismo grado
  - Si el grado es k, el grafo es k-regular

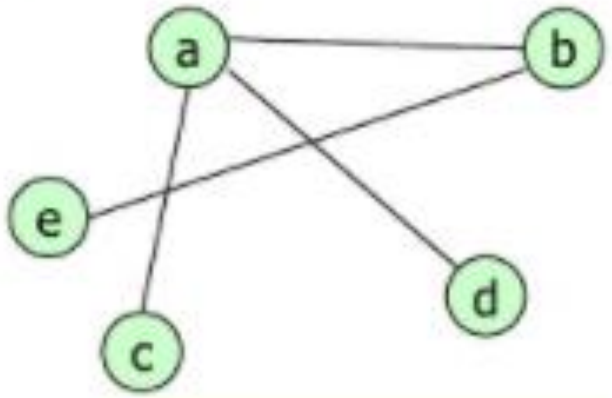


Grafo 3 - regular

- Grafo **Completo**
  - Tiene una arista entre cualquier par de vértices

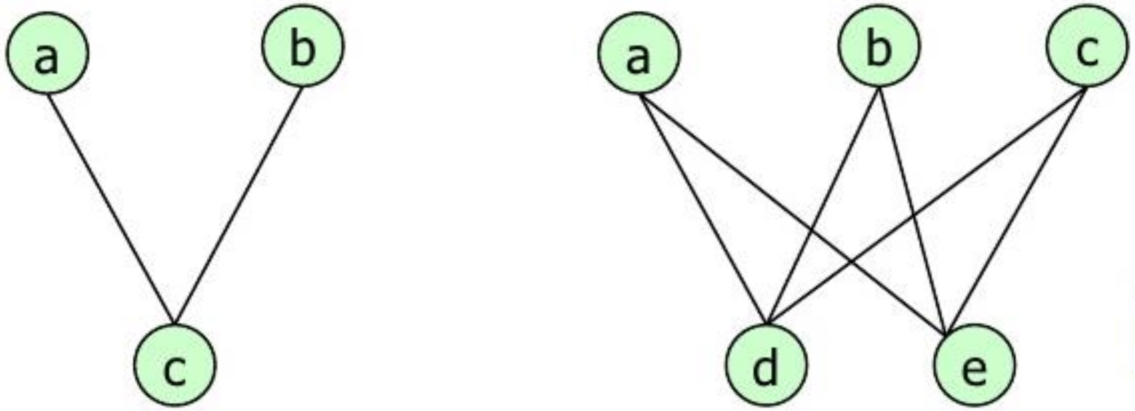


Grafo completo



Grafo **No** completo

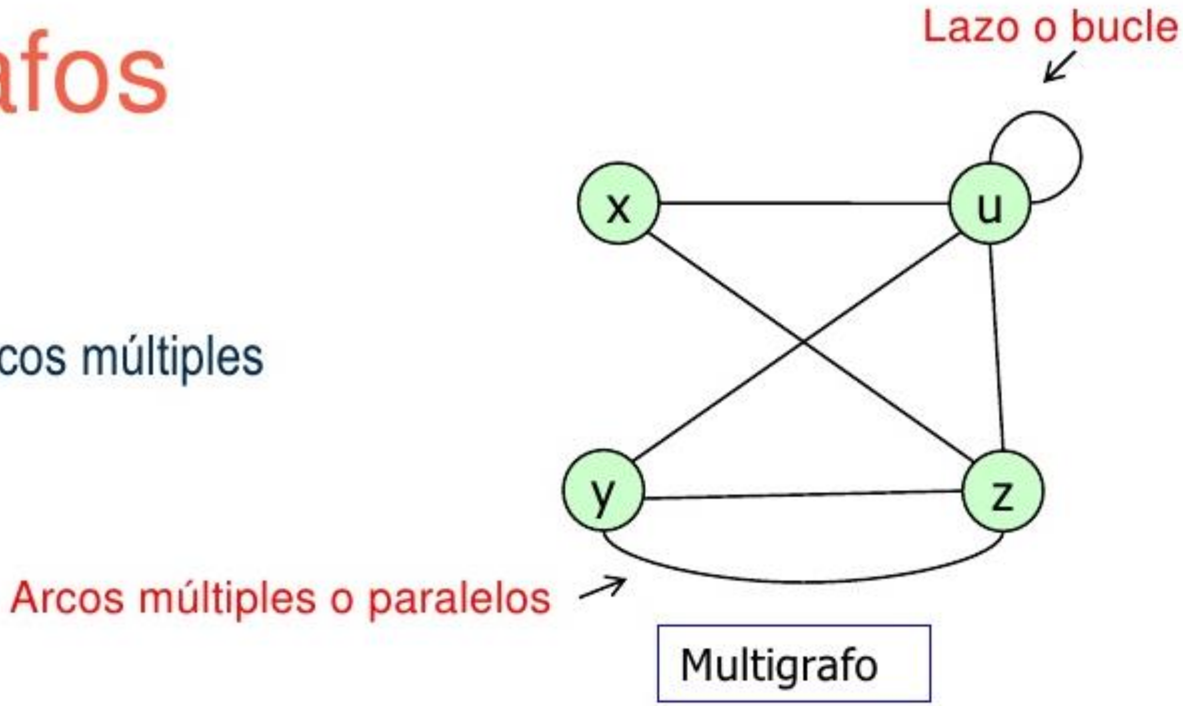
- Grafo **Bipartito**
  - “Bipartito” significa que tiene 2 partes
  - G= { V1 u V2, E}
  - Sus vértices son la unión de dos grupos de vértices bajos las siguientes condiciones:
    - V1 y V2 son conjuntos disjuntos
    - Cada arista del Grafo une un vértice de V1 con uno de V2
    - No existen aristas uniendo vértices del mismos conjunto V1 o V2



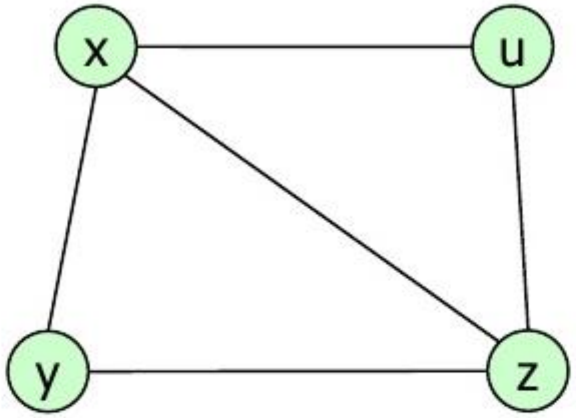
Grafos Bipartitos

Tipos de Grafos

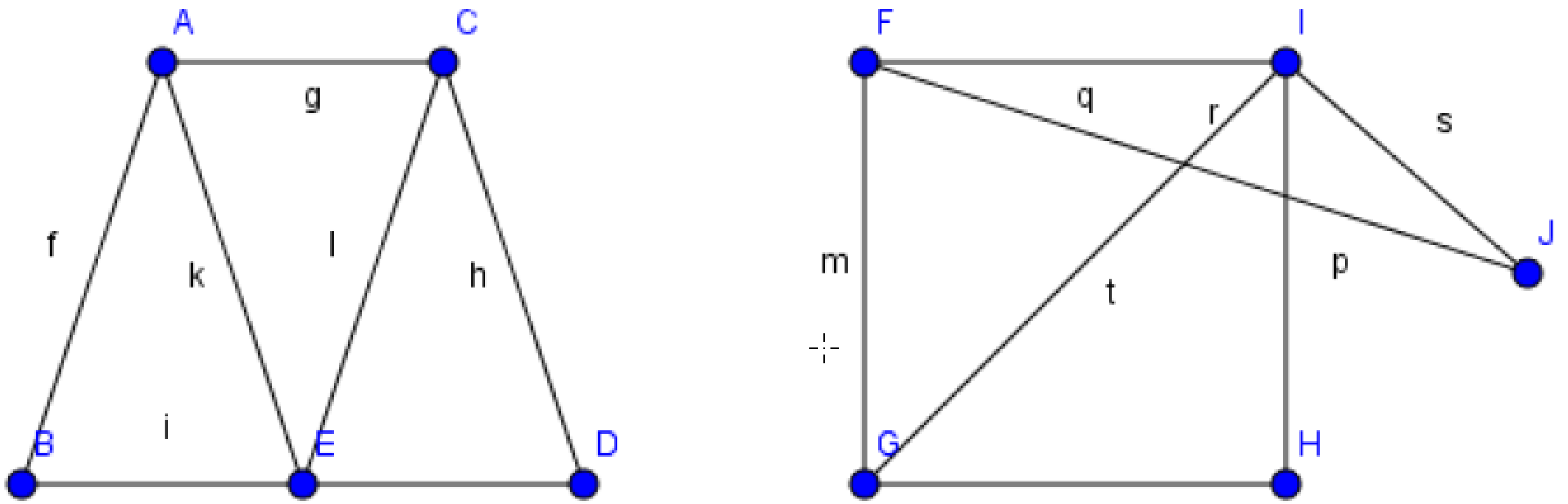
- Mulitgrafo
  - Es un grafo que tiene arcos múltiples (paralelos) o lazos



- Grafo Simple
  - Es un grafo o digrafo que <u>no</u> tiene bucles y que <u>no</u> es un multigrafo



Grafos Isomorfos. - Observemos los siguientes grafos.



Nos damos cuenta de que son diferentes, pero si los estudiamos detenidamente, podemos ver que encontramos muchas semejanzas entre ellos.

1. Ambos tienen el mismo número de vértices o nodos e igual número de lados o aristas.

2. Existe un nodo en cada uno de ellos que está unido al resto de vértices nodo E en el primero y nodo I en el segundo.

3. Si renombramos los vértices del segundo grafo: $F \rightarrow A_1, G \rightarrow B_1,\ I \rightarrow C_1,\ H \rightarrow D_1, J \rightarrow E_1$

REPRESENTACIÓN MATRICIAL DE GRAFOS

MATRIZ DE ADYACENCIA

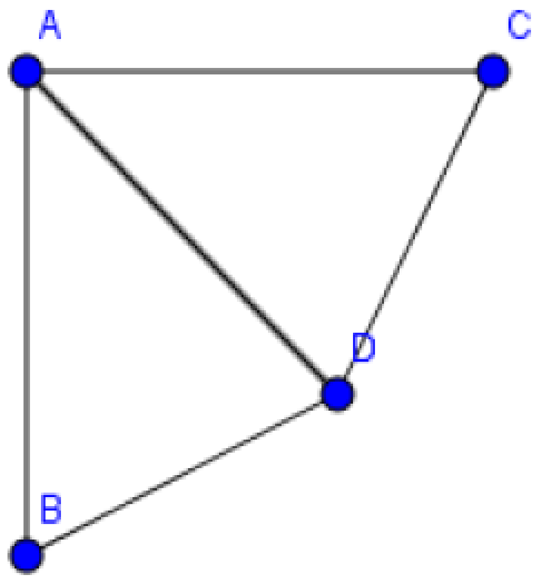
Representación matricial

Sea $G=(V,E)$, con $|V|=n$

Llamamos **matriz de adyacencia** de G a la matriz nxn, $A=(a_{ij})$, donde

$$a_{ij} = \begin{cases} 1 & \text{si } (v_i, v_j) \in E \\ 0 & \text{si } (v_i, v_j) \in E \end{cases}$$

Determinar la matriz de adyacencia del siguiente grafo.



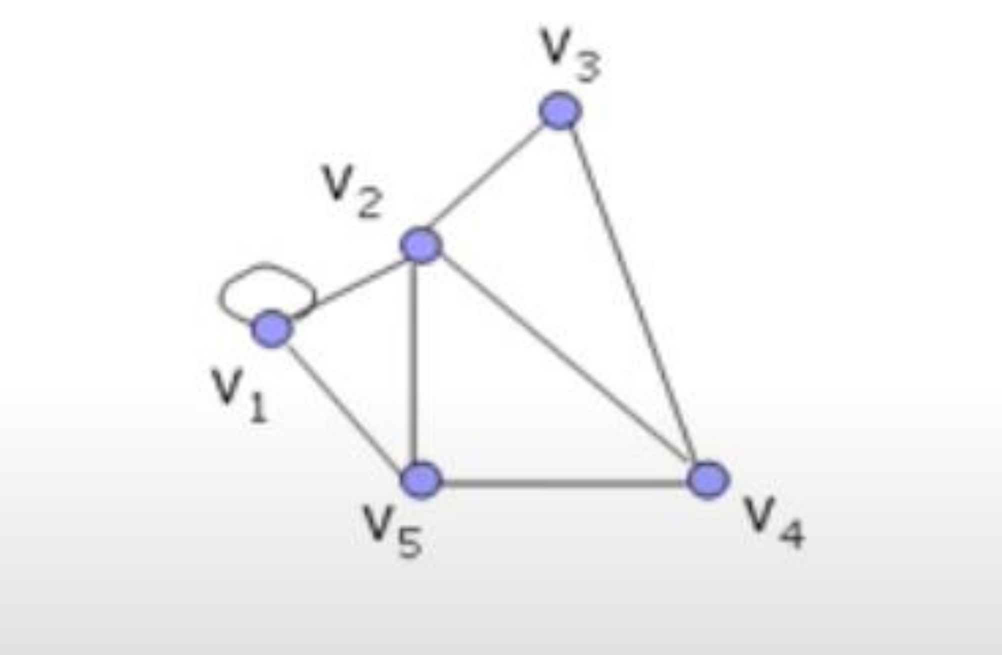
	A	B	C	D
A	0	1	1	1
B	1	0	0	1
C	1	0	0	1
D	1	1	1	0

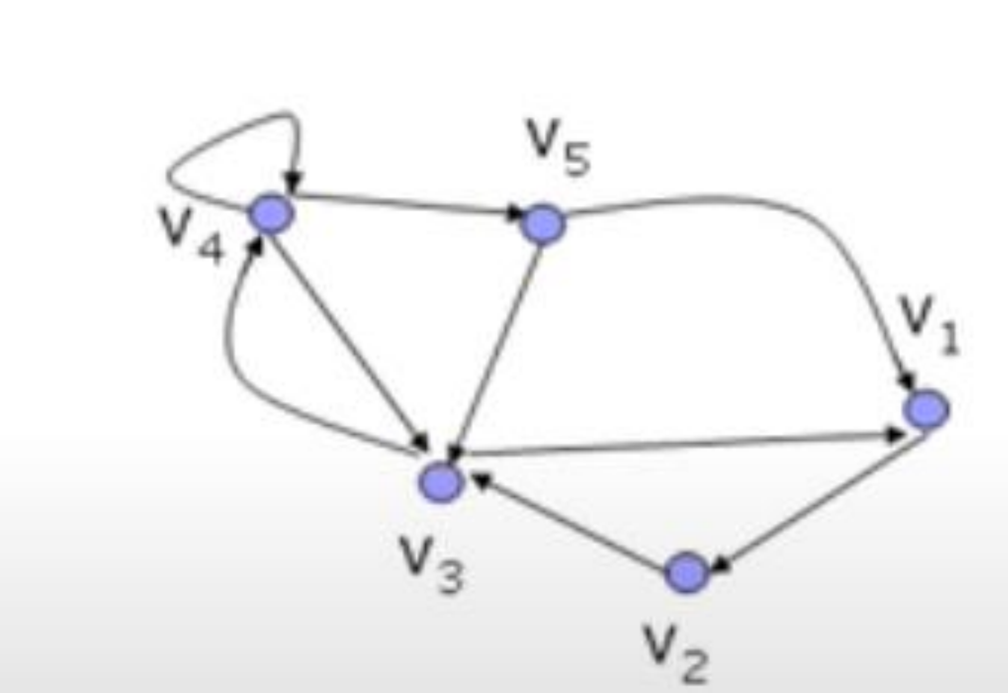
1. Los nodos representan las columnas y filas del grafo.

2. Por cada arista que unos dos nodos se escribe el valor de 1, si no hay unión se escribe el 0.

3. Si la arista es un bucle o lazo es escribe el valor de 2.

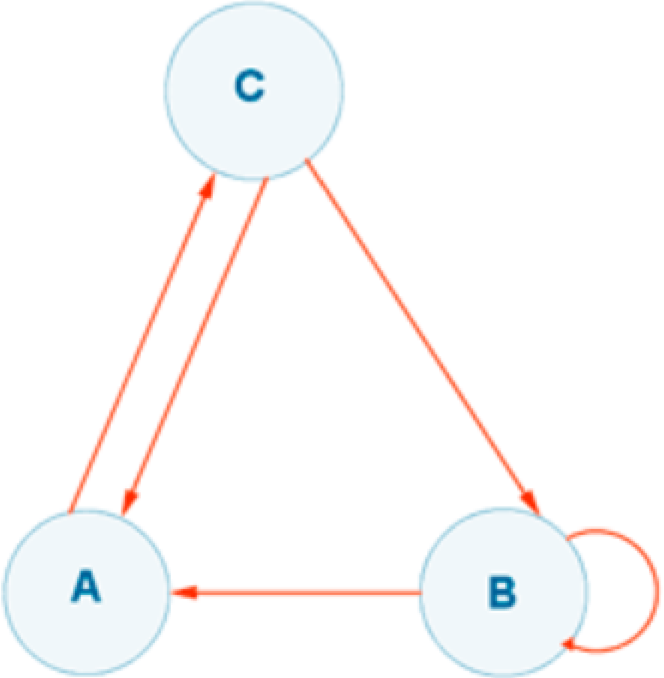
DETERMINE LA MATRIZ ADYACENTE DE:



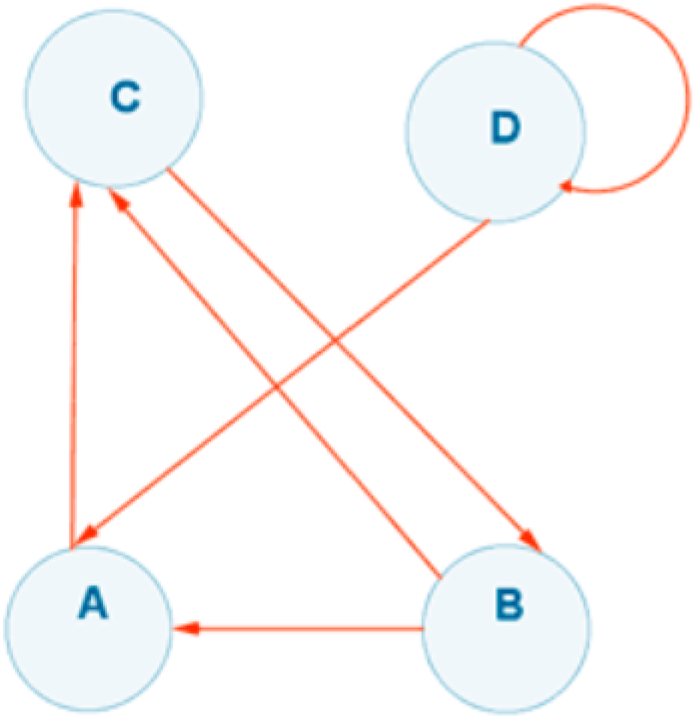


1.

$$M_1 = \begin{array}{c c} & \begin{array}{cc} A & B \end{array} \\ \begin{array}{c} A \\ B \end{array} & \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix} \end{array}$$

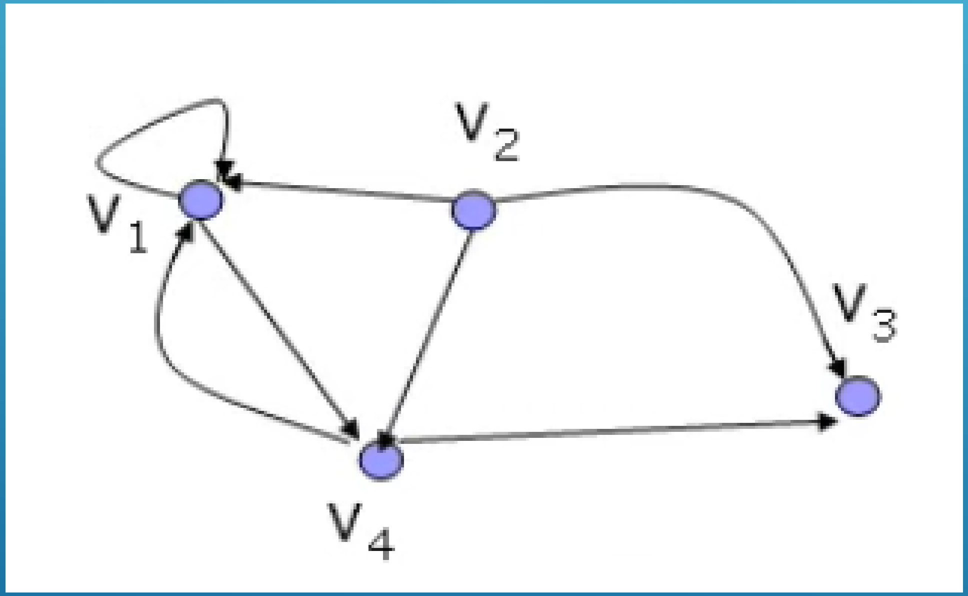


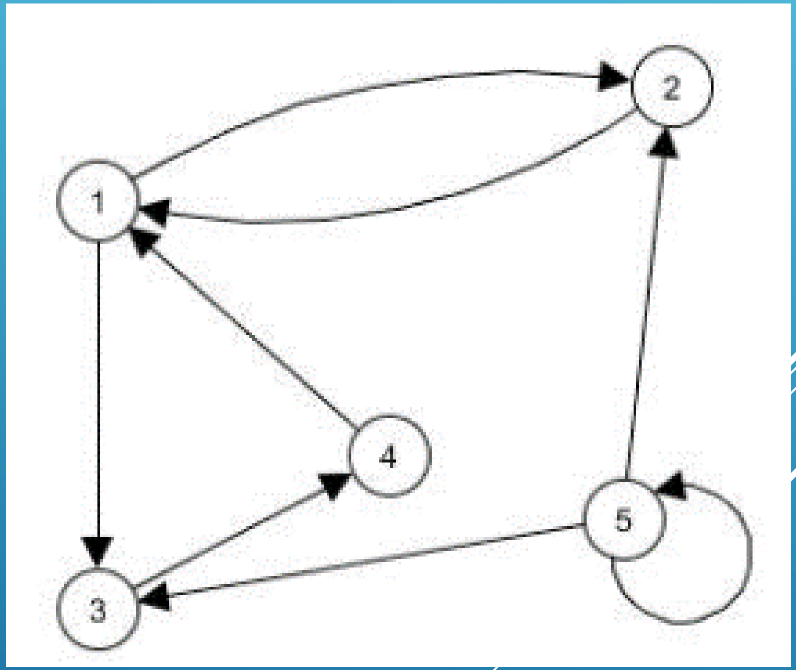
$$M_2 = \begin{array}{c c} & \begin{array}{ccc} A & B & C \end{array} \\ \begin{array}{c} A \\ B \\ C \end{array} & \begin{pmatrix} 0 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 1 & 0 \end{pmatrix} \end{array}$$

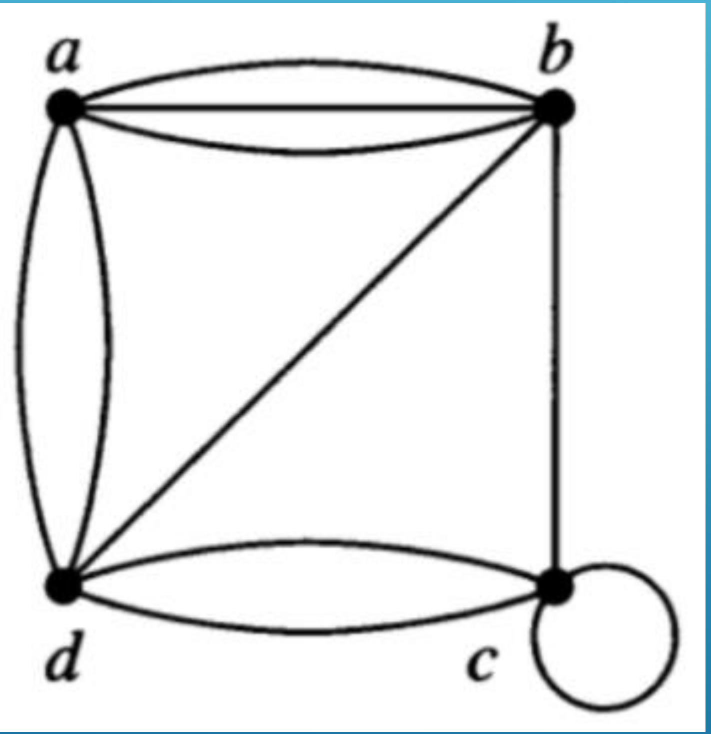


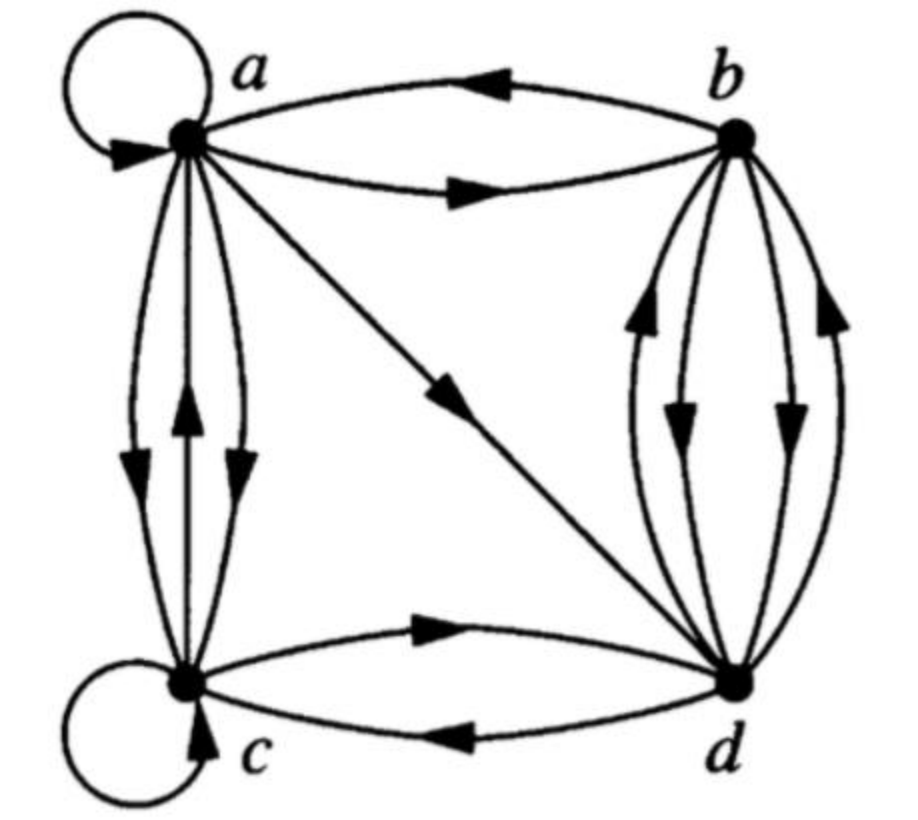
$$M_3 = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{pmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 \end{pmatrix} \end{array}$$

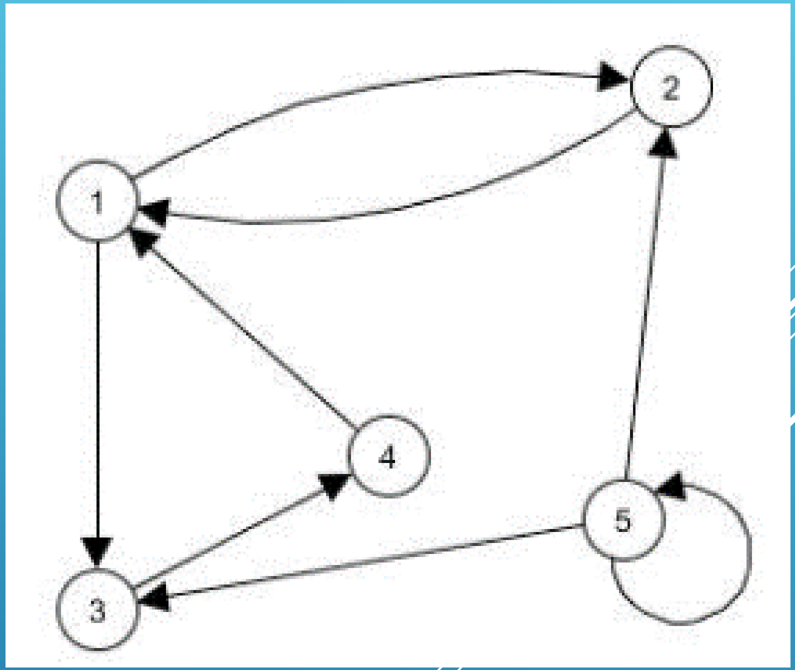
ACTIVIDAD






a
b
d
c


a
b
c
d


1
2
3
4
5

MATRIZ DE INCIDENCIA

Una matriz de incidencia representa las relaciones binarias entre dos elementos, en nuestro caso entre un vértice o nodo y una arista.

Para construir la matriz de incidencia a partir de un grafo debemos realizar una matriz de n x a donde n es el número de vértices y a es el número de arista.

1. Las columnas de la matriz representan las aristas del grafo.

2. Las filas representan a los distintos nodos.

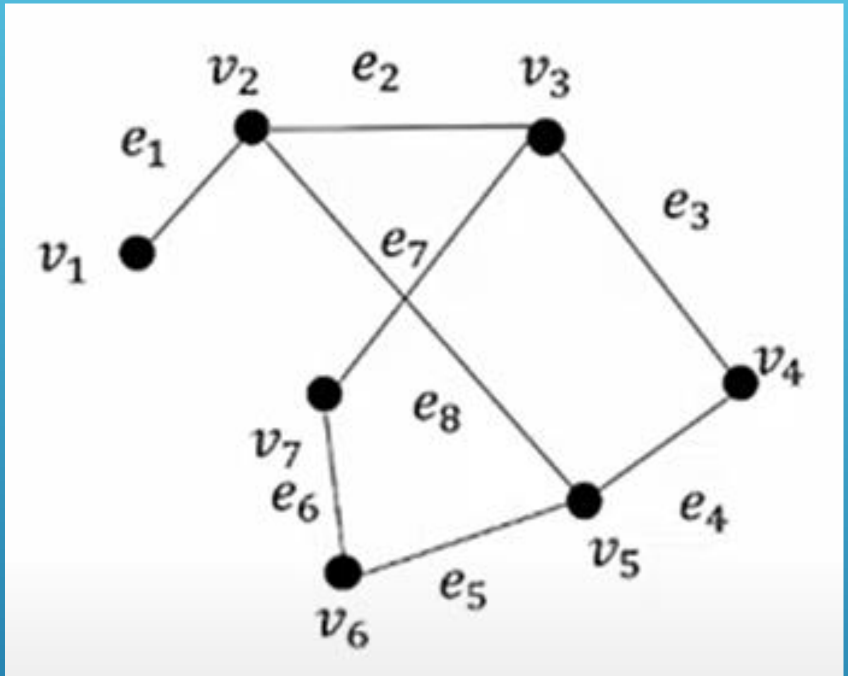
3. Por cada nodo unido por una arista ponemos el 1, en el lugar correspondiente, si no está unido el nodo a ninguna arista ponemos 0.

MATRIZ DE INCIDENCIA

- Sea G un grafo de n vértices, llamaremos **matriz de incidencia** A del grafo G, a una matriz $A = [a_{ij}] \in \mathcal{M}_n(\mathbb{R})$, que cumple:
  - Las columnas de la matriz representan las aristas del grafo
  - Las filas representan a los distintos nodos.
  - Por cada nodo unido por una arista:

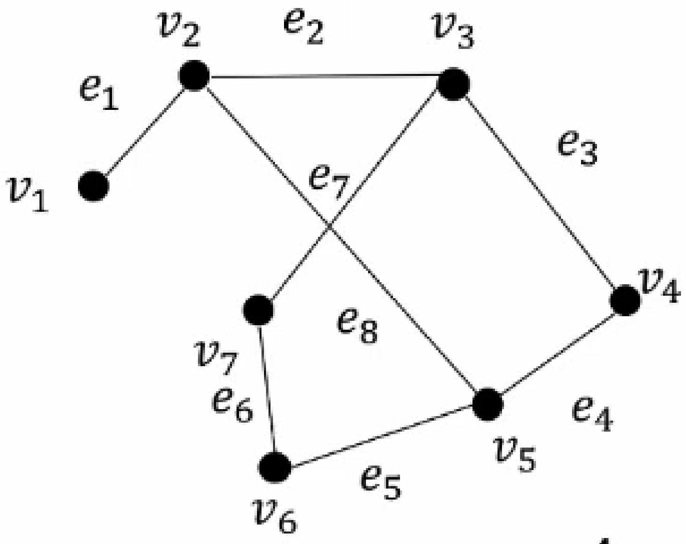
$$a_{ij} = \begin{cases} 1, si\ v_i\ es\ adyacente\ a\ e_j \\ 0, si\ v_i\ no\ es\ adyacente\ a\ e_j \end{cases}$$

EJEMPLO



Para ello elaboraremos la matriz tomando en cuenta las recomendaciones las filas serán los nodos o vértices y las columnas serán las aristas.

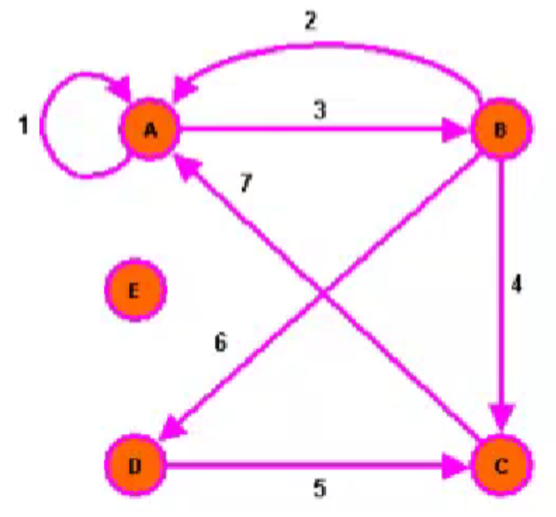
Ejemplo



$$
A = \begin{array}{cc}
 & \begin{array}{cccccccc} e_1 & e_2 & e_3 & e_4 & e_5 & e_6 & e_7 & e_8 \end{array} \\
\begin{array}{c} v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \\ v_7 \end{array} &
\begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 0
\end{bmatrix}
\end{array}
$$

Matriz de Incidencia

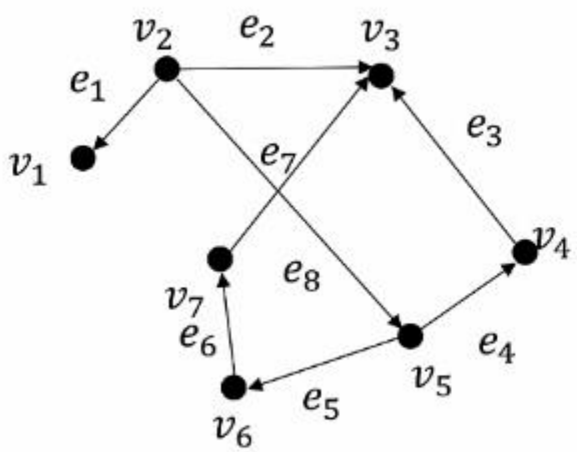
$$a_{i,j} = \begin{cases} 1 & \text{Si la línea } e_j \text{ sale del vértice } v_i \\ -1 & \text{Si la línea } e_j \text{ llega al vértice } v_i \\ 0 & \text{Para cualquier otro valor} \end{cases}$$



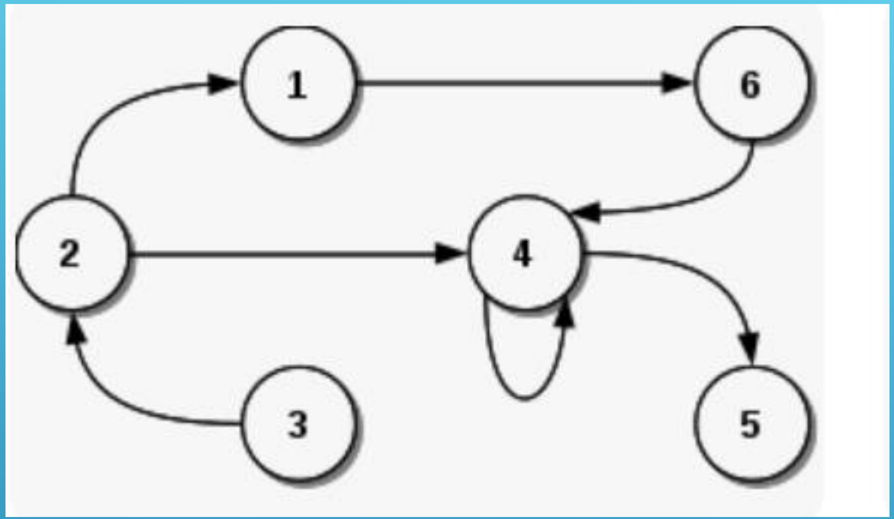
A(G)

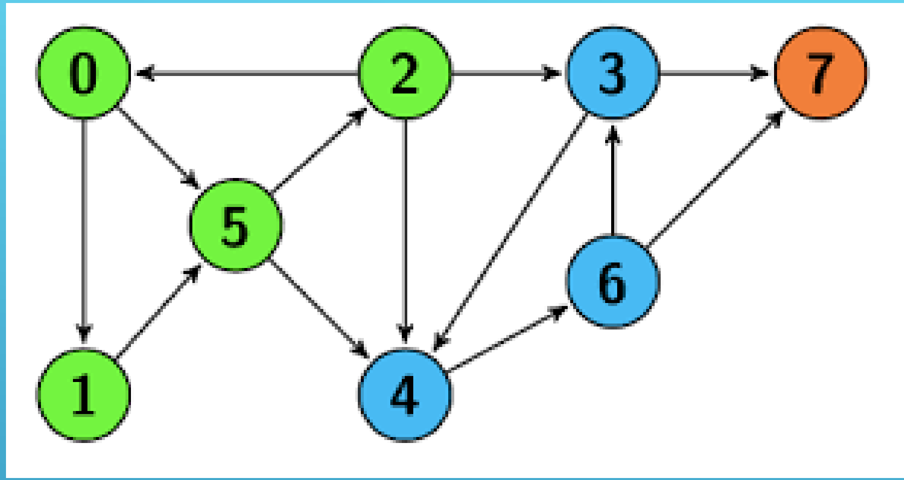
	1	2	3	4	5	6	7
A	±1	-1	1	0	0	0	-1
B	0	1	-1	1	0	1	0
C	0	0	0	-1	-1	0	1
D	0	0	0	0	1	-1	0
E	0	0	0	0	0	0	0

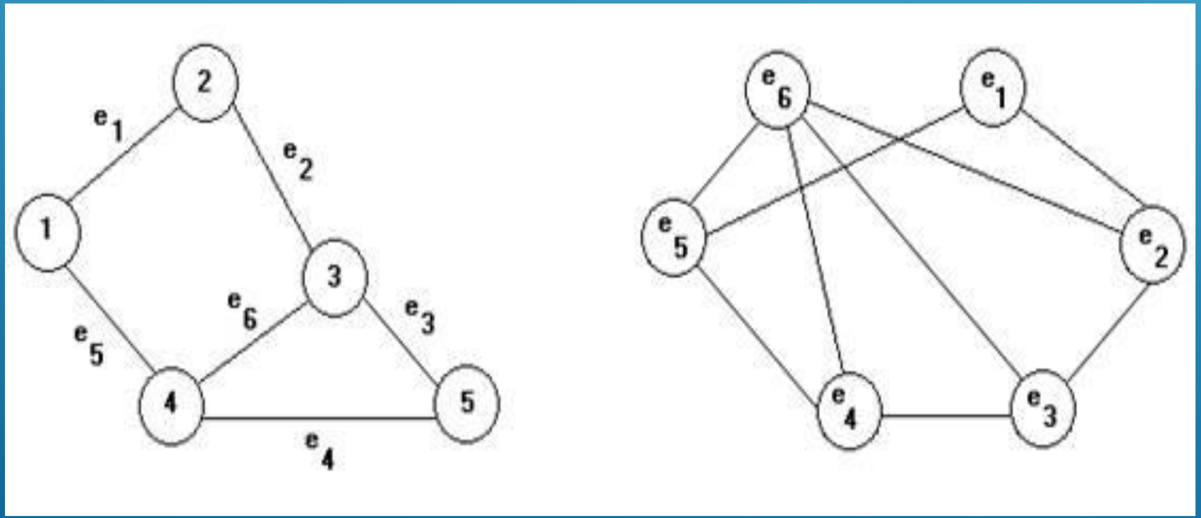
Ejemplo

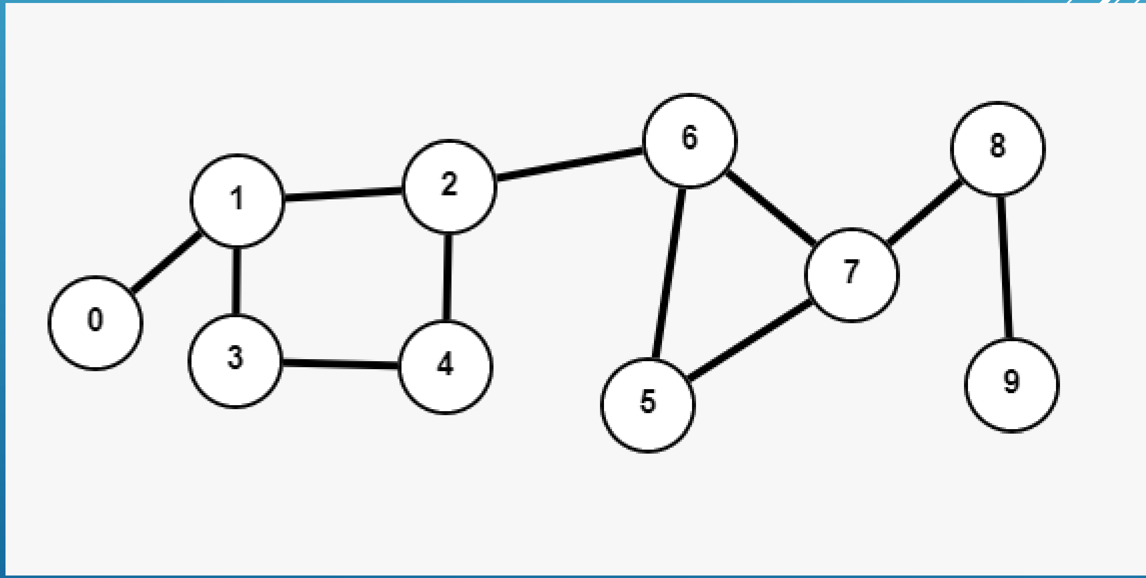


$$A = \begin{array}{cc} & \begin{array}{cccccccc} e_1 & e_2 & e_3 & e_4 & e_5 & e_6 & e_7 & e_8 \end{array} \\ \begin{array}{c} v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \\ v_7 \end{array} & \begin{bmatrix} -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & -1 & -1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & -1 & 0 \end{bmatrix} \end{array}$$


1
6
2
4
3
5


0
2
3
7
5
6
1
4


2
e_1
e_2
1
3
e_6
e_3
e_5
4
5
e_4
e_6
e_1
e_5
e_2
e_4
e_3


6
8
1
2
7
0
3
4
5
9

CAMINOS Y CICLOS EN GRAFOS:

CAMINO → sucesión de aristas adyacentes

CICLO (o circuito) → camino cerrado (vértice inicial = vértice final)

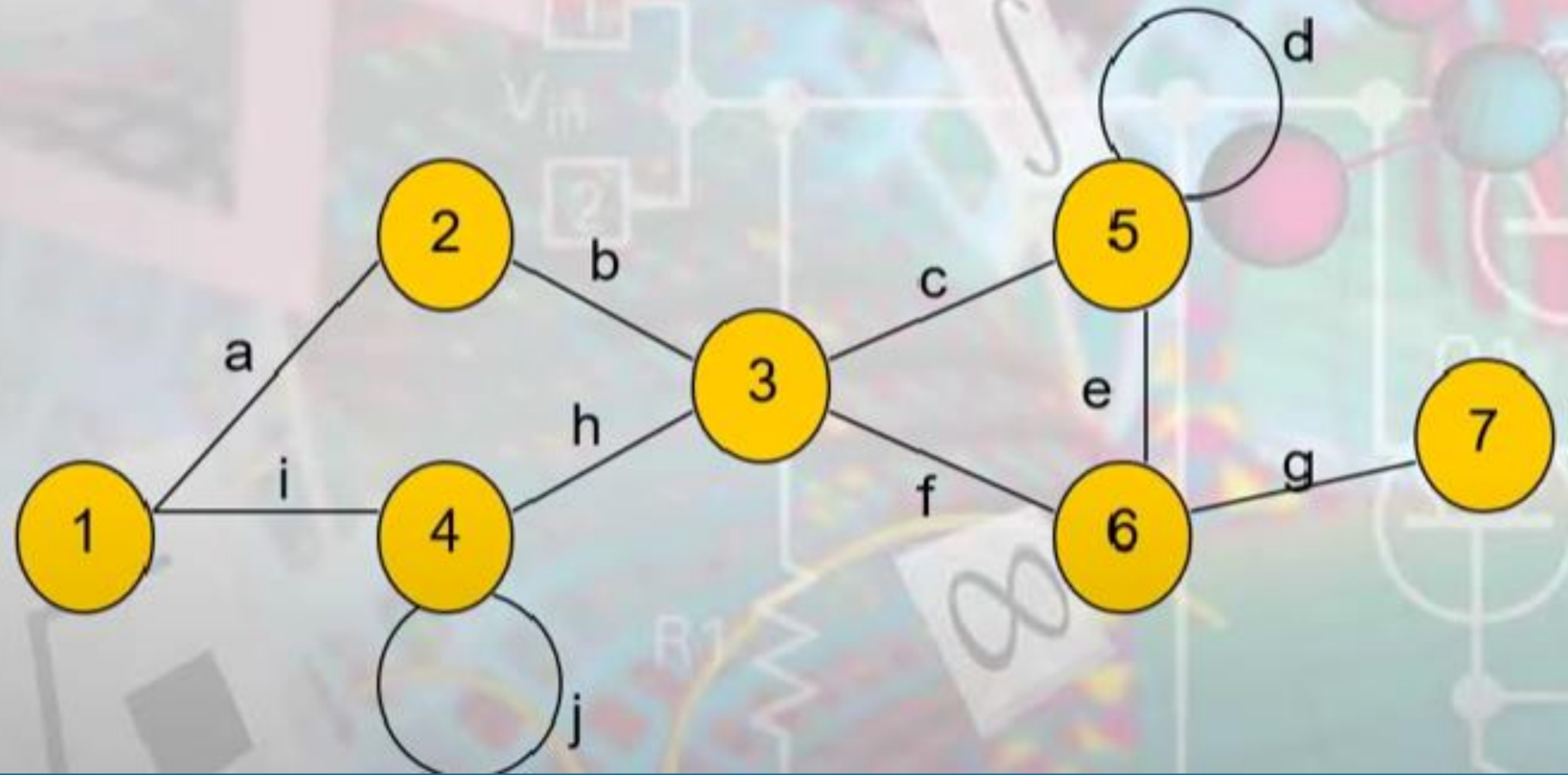
LONGITUD de un camino → cantidad de aristas que lo componen

CAMINO SIMPLE → si todos los vértices son distintos

EJEMPLO:

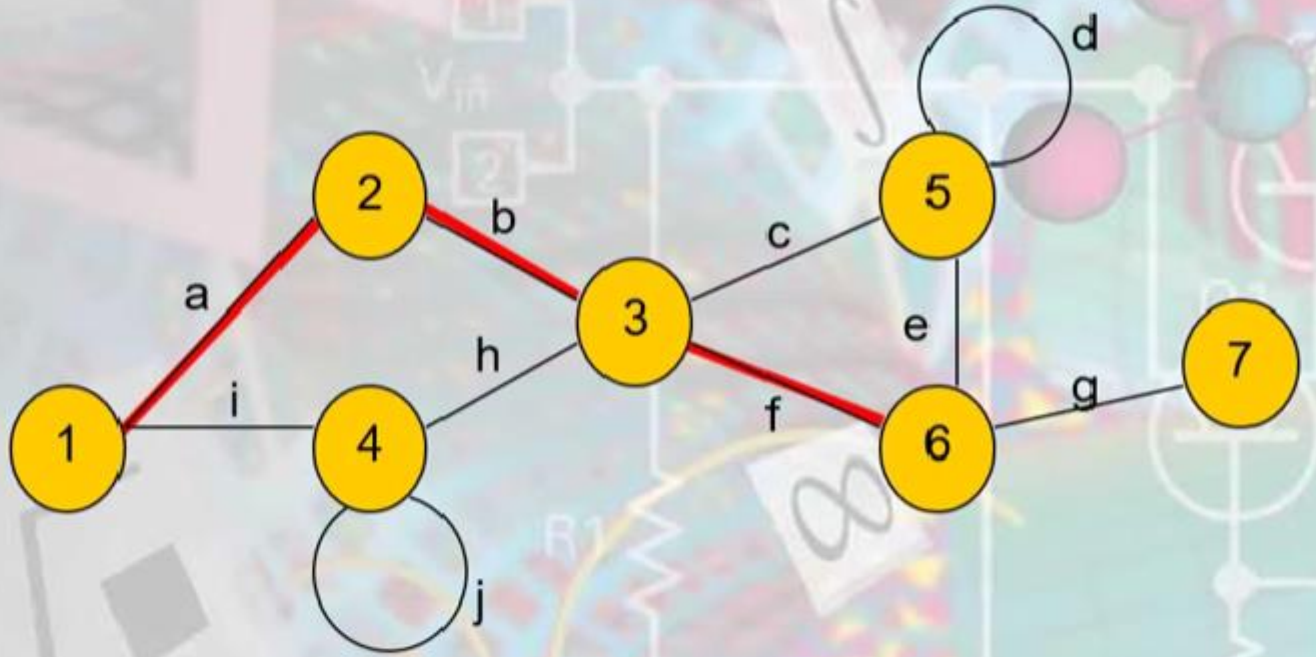
En el siguiente grafo: G =(V;A; φ) con V = {1, 2, 3, 4, 5, 6, 7},

A={a, b, c, d, e, f, g, h, i, j}, busquemos caminos entre los vértices "1" y "6", e indiquemos la longitud de cada uno de ellos:



Un posible camino de 1 a 6 es: $C_1 = (1; a; 2; b; 3; f; 6)$

La longitud es: $long[C_1] = 3$

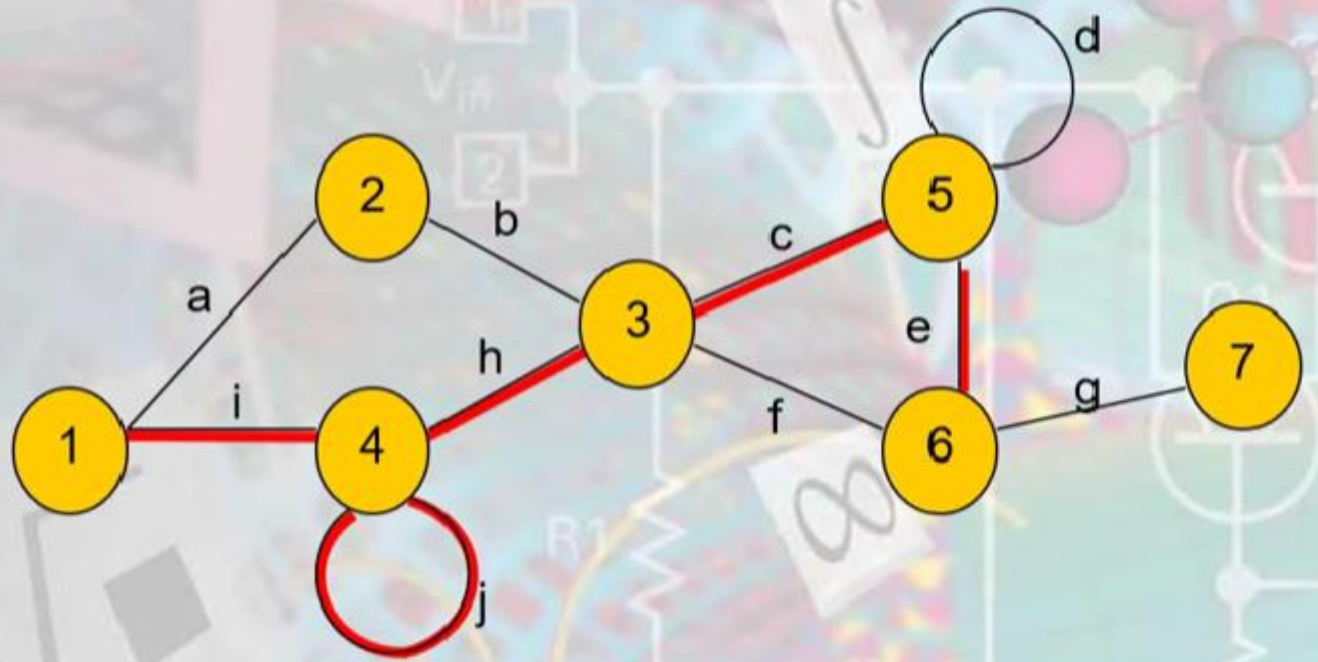


¿Este camino es SIMPLE?

Sí, este camino es SIMPLE pues no repite vértices.

Otro posible camino de 1 a 6 es: $C_2 = (\ 1;\ i;\ 4;\ j;\ 4;\ h;\ 3;\ c;\ 5;\ e;\ 6\ )$

La longitud es: $long[C_2] = 5$

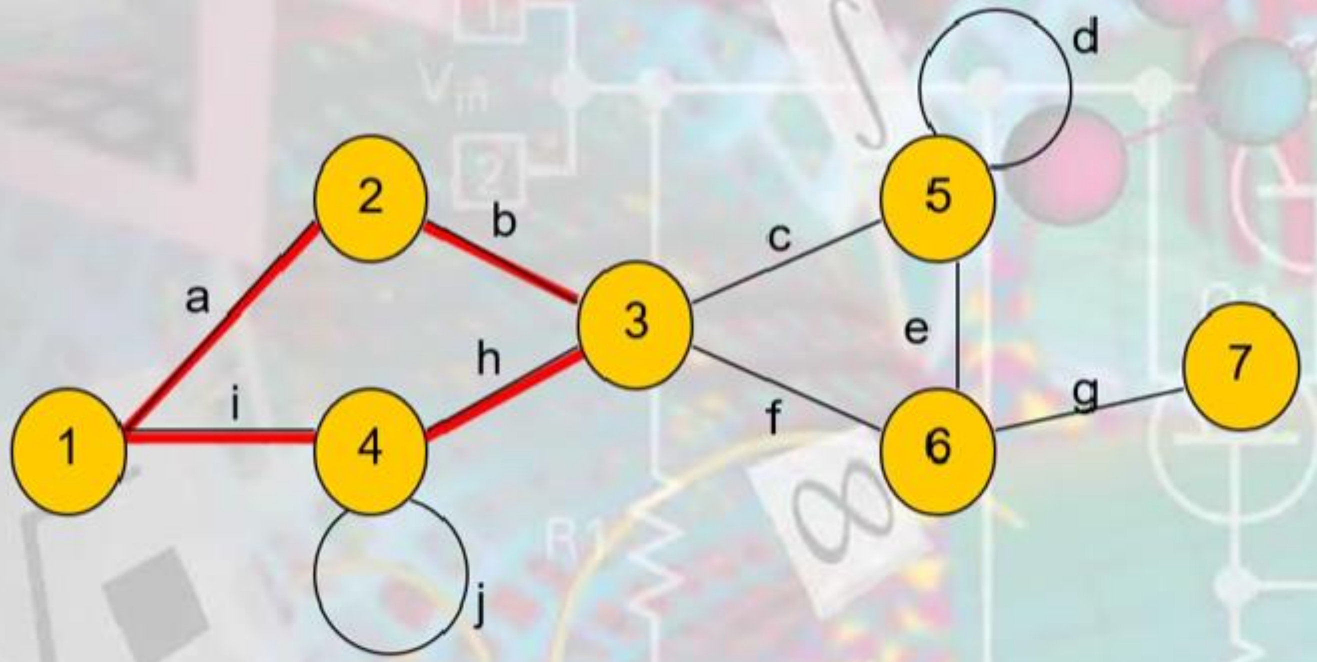


¿Este camino es SIMPLE?

NO, este camino NO es SIMPLE pues repite el vértice 4.

Un posible ciclo es: $C_1 = (1; a; 2; b; 3; h; 4; i; 1)$

La longitud es: $long[C_1] = 4$

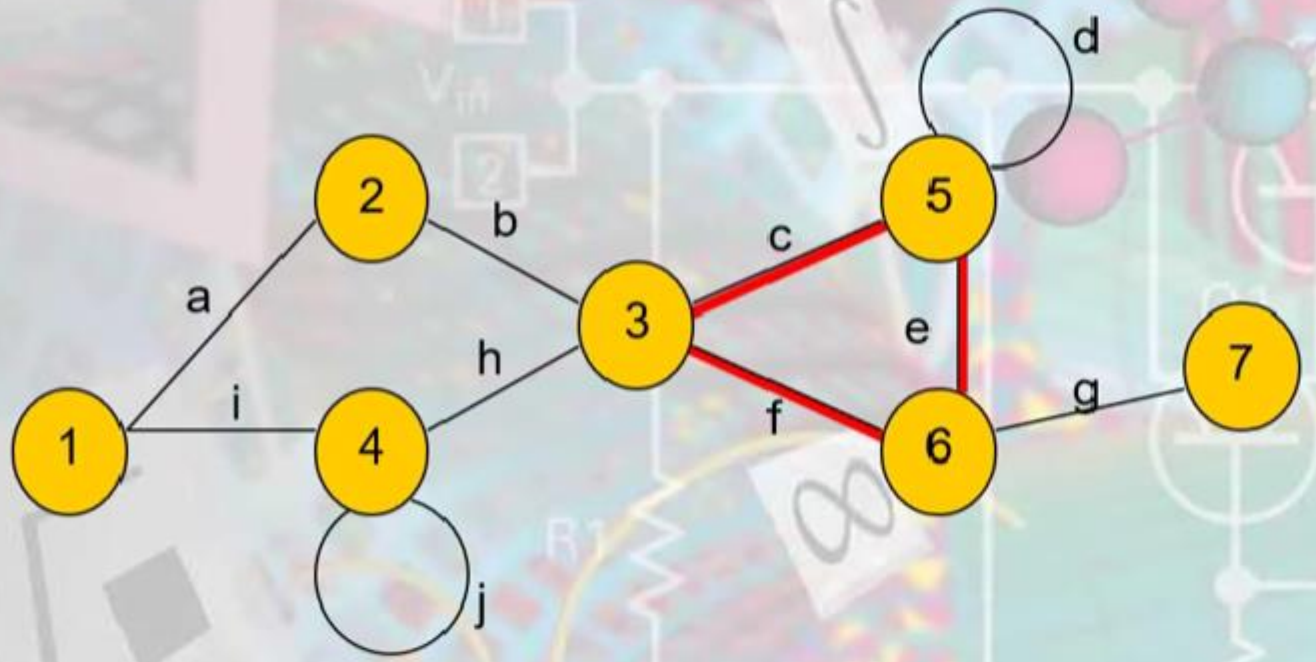


¿Este ciclo es SIMPLE?

Sí, este ciclo es SIMPLE pues no repite vértices.

Otro posible ciclo es: $C_2 = (\ 3;\ c;\ 5;\ e;\ 6;\ f;\ 3\ )$

La longitud es: $\mathrm{long}[C_2] = 3$

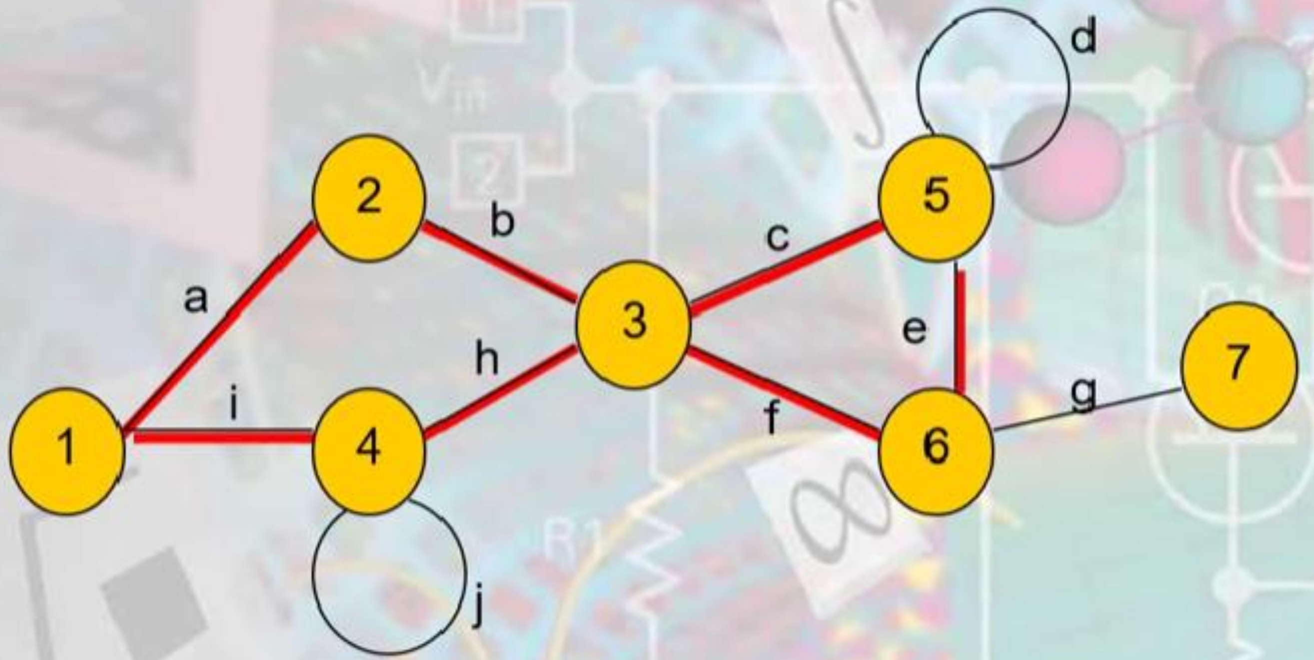


¿Este ciclo es SIMPLE?

Sí, este ciclo es SIMPLE pues no repite vértices.

Otro posible ciclo es: $C_3 = (\ 1;\ a;\ 2;\ b;\ 3;\ c;\ 5;\ e;\ 6;\ f;\ 3;\ h;\ 4;\ i;\ 1\ )$

$long[C_3] = 7$



¿Este ciclo es SIMPLE?

No, este ciclo NO es SIMPLE pues repite el vértice 3.

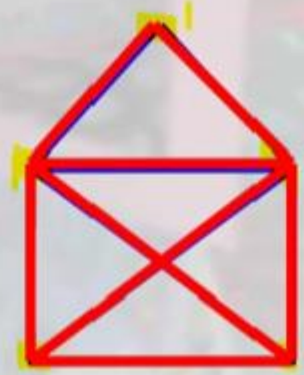
CAMINOS Y CICLOS EULERIANOS

CAMINO DE EULER → camino que pasa por todas las aristas sólo una vez

CICLO DE EULER → ciclo que pasa por todas las aristas del grafo

Un grafo tiene camino euleriano ⟺ Es conexo, y todos los vértices tienen grado par, o a lo sumo dos grado impar.

Un grafo tiene ciclo euleriano ⟺ Es conexo, y todos los vértices tienen grado par.



Este grafo no tiene ciclo euleriano
pues hay dos vértices de grado 3.
Tiene sólo camino euleriano.

Este grafo tiene ciclo euleriano pues
todos sus vértices tienen grado par.

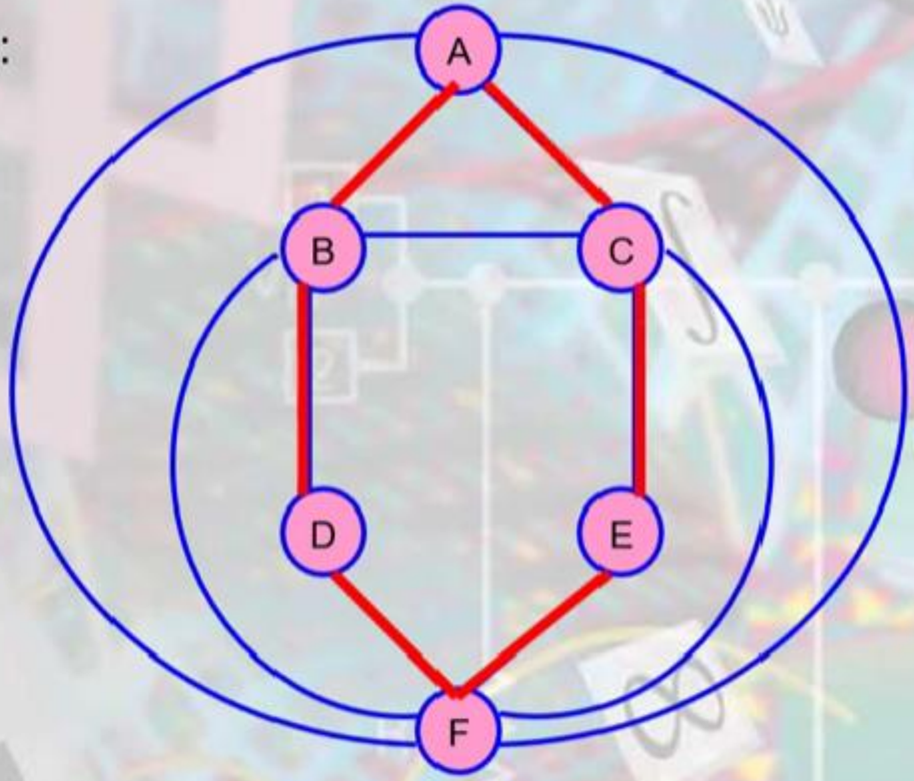
CAMINOS Y CICLOS HAMILTONIANOS

CAMINO DE HAMILTON → camino simple que pasa por todos los vértices

CICLO DE HAMILTON → ciclo simple que pasa por todos los vértices

Observación: no necesariamente va a pasar por todas las aristas, pues en muchos casos repetiría vértices y no sería hamiltoniano.

EJEMPLO:



Un posible ciclo hamiltoniano es: (A;B;D;F;E;C;A)

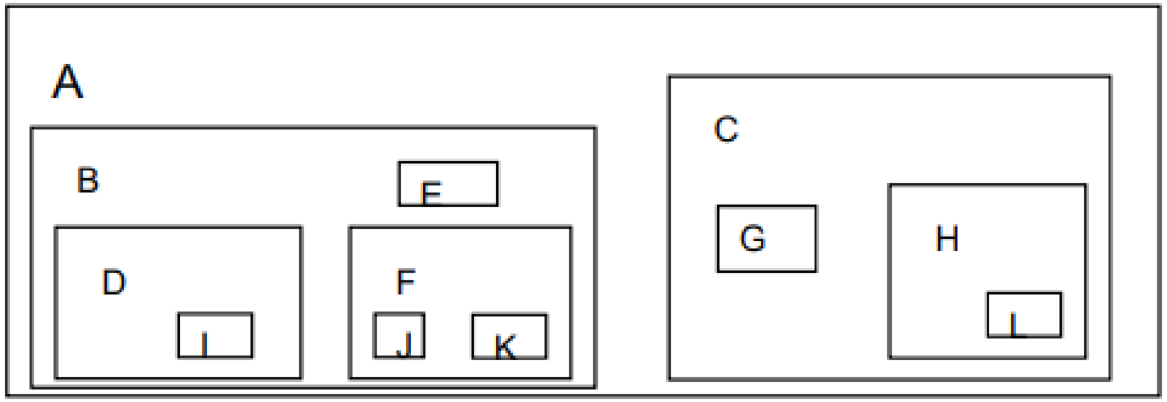
TEORÍA DE ÁRBOLES
¿QUÉ ES UN ÁRBOL?

- Un árbol es una estructura de datos *no lineal* formada por un conjunto de nodos.
- Son estructuras jerárquicas, cada elemento puede tener diferentes siguientes elementos.
- El concepto de árbol implica una estructura en la que los datos se organizan de modo que los elementos de información están relacionados entre sí a través de ramas

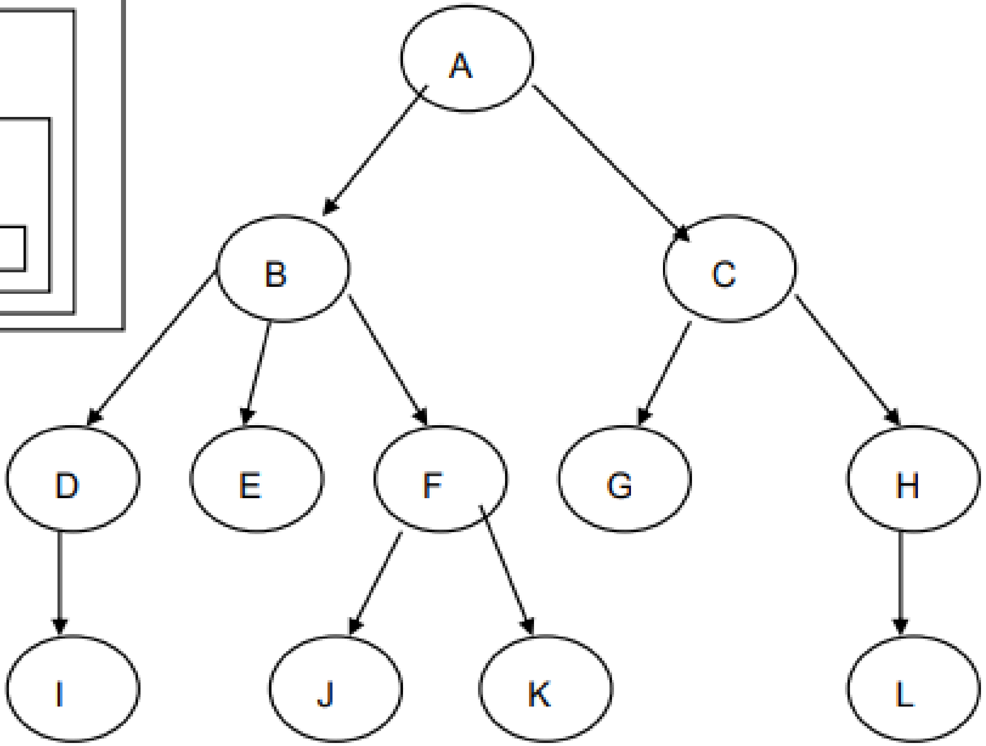
PROPIEDADES

- En la ciencia de la computación definimos un **árbol como un conjunto de nodos y líneas**. Un **nodo es un elemento de información** que reside en el árbol. Una **línea es un par de nodos ordenados <u,v>**, y a la **secuencia de líneas se le denomina ruta (path).**
- Además, los árboles tienen las siguientes propiedades:
- Tienen un nodo al que se le llama raíz del árbol.
- Todos los nodos, excepto la raíz, tienen una sola línea de entrada (el nodo raíz no tiene ninguna).
- Existe una ruta única del nodo raíz a todos los demás nodos del árbol.
- Si hay una ruta <a,b>, entonces a ‘b’ se le denomina ‘hijo’ de ‘a’ y es el nodo raíz de un subárbol.

Gráficamente puede representarse una estructura árbol de diferentes maneras y todas ellas equivalentes:



Árbol por medio de diagramas de Venn



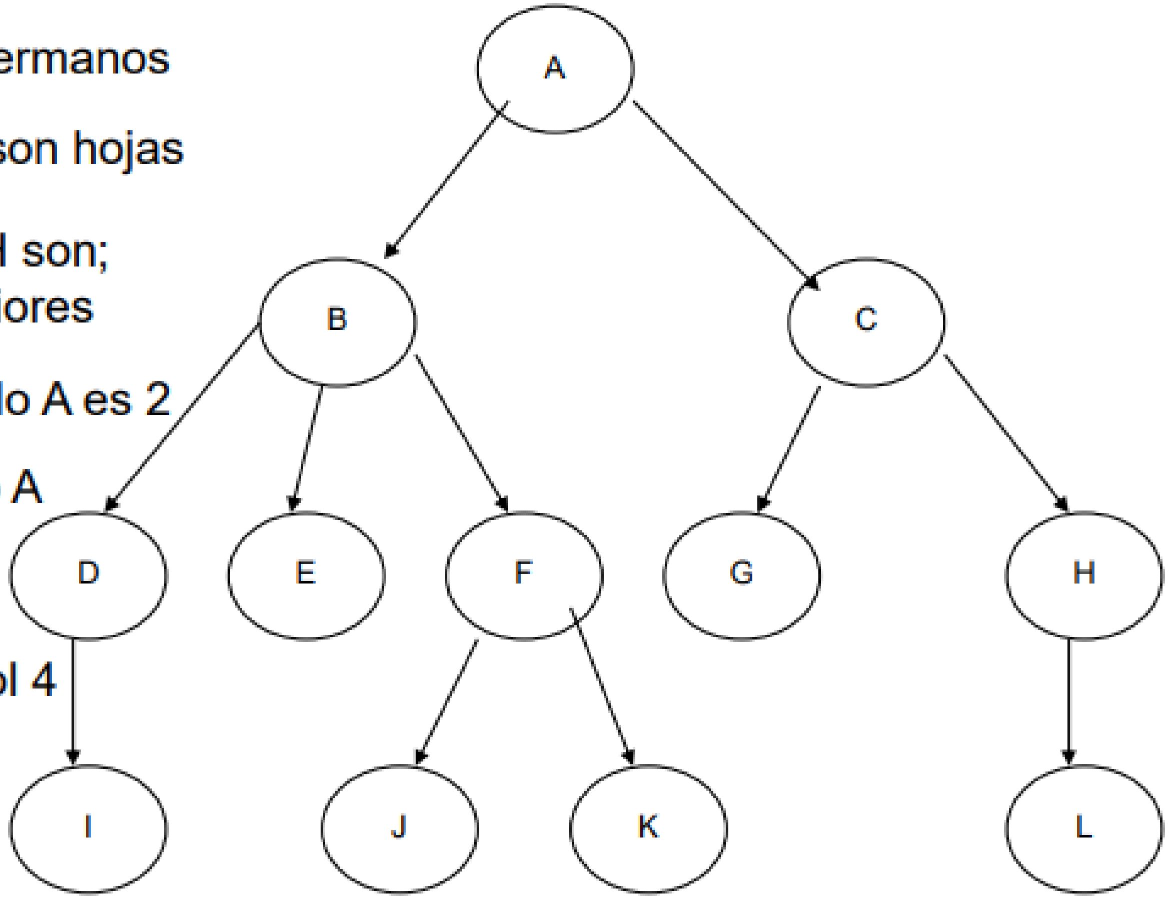
Árbol mediante grafo.

CARACTERÍSTICAS Y PROPIEDADES DE LOS ÁRBOLES.

1. * **NODO** indica un elemento, o ítem, de información.
2. * Todo árbol que no es vacío, tiene **un único nodo raíz.**
3. * Un nodo X es descendiente directo de un nodo Y, si el nodo X es apuntado por el nodo Y. **X es hijo de Y.**
4. * Un nodo X es antecesor directo de un nodo Y, si el nodo X apunta al nodo Y. **X es padre de Y.**
5. *Se dice que todos los nodos que son descendientes directos (hijos) de un mismo nodo (padre), son **hermanos.**
6. * Todo nodo que no tiene ramificaciones (hijos), se conoce con el nombre de **terminal u hoja.**
7. * Todo nodo que no es raíz, ni terminal u hoja se conoce con el nombre de **interior.**
8. * **Grado** es el número de descendientes directos de un determinado nodo. **Grado del árbol** es el máximo grado de todos los nodos del árbol.
9. * **Nivel** es el número de arcos que deben ser recorridos para llegar a un determinado nodo. Por definición, la raíz tiene nivel 1.
10. ***Altura** del árbol es el máximo número de niveles de todos los nodos del árbol.

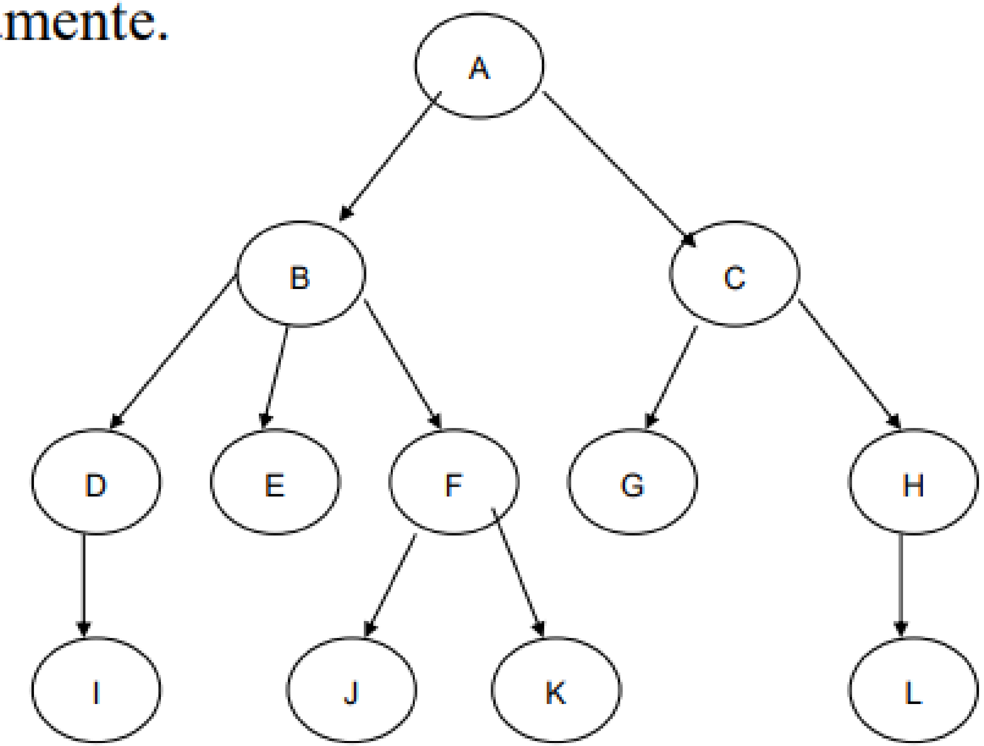
Ejemplo

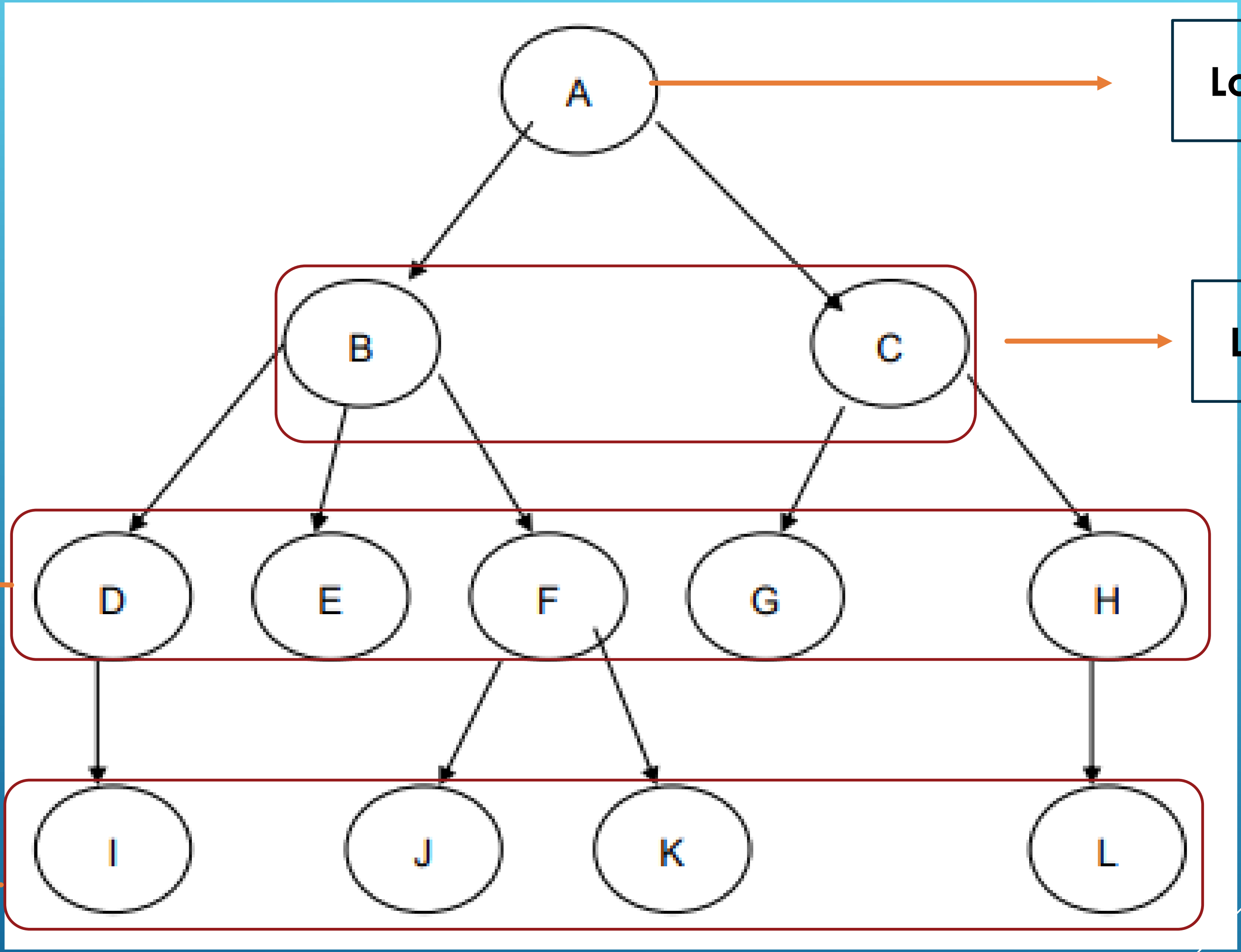
A es la raíz del árbol
B es hijo de A
A es padre de B
B y C son hermanos
I,E,J,K,G,L son hojas
B, D, F, C, H son; nodos interiores
El grado de nodo A es 2
Nivel del nodo A es 1 (def)
Nivel B es 2
Altura del árbol 4



LONGITUD DE CAMINO INTERNO Y EXTERNO

- Se define la longitud de camino X como el número de arcos que deben ser recorridos para llegar desde la raíz al nodo X. Por definición la raíz tiene longitud de camino 1, sus descendientes directos longitud de camino 2 y así sucesivamente.





Longitud de Camino 1

Longitud de Camino 2

Longitud de Camino 3

Longitud de Camino 4

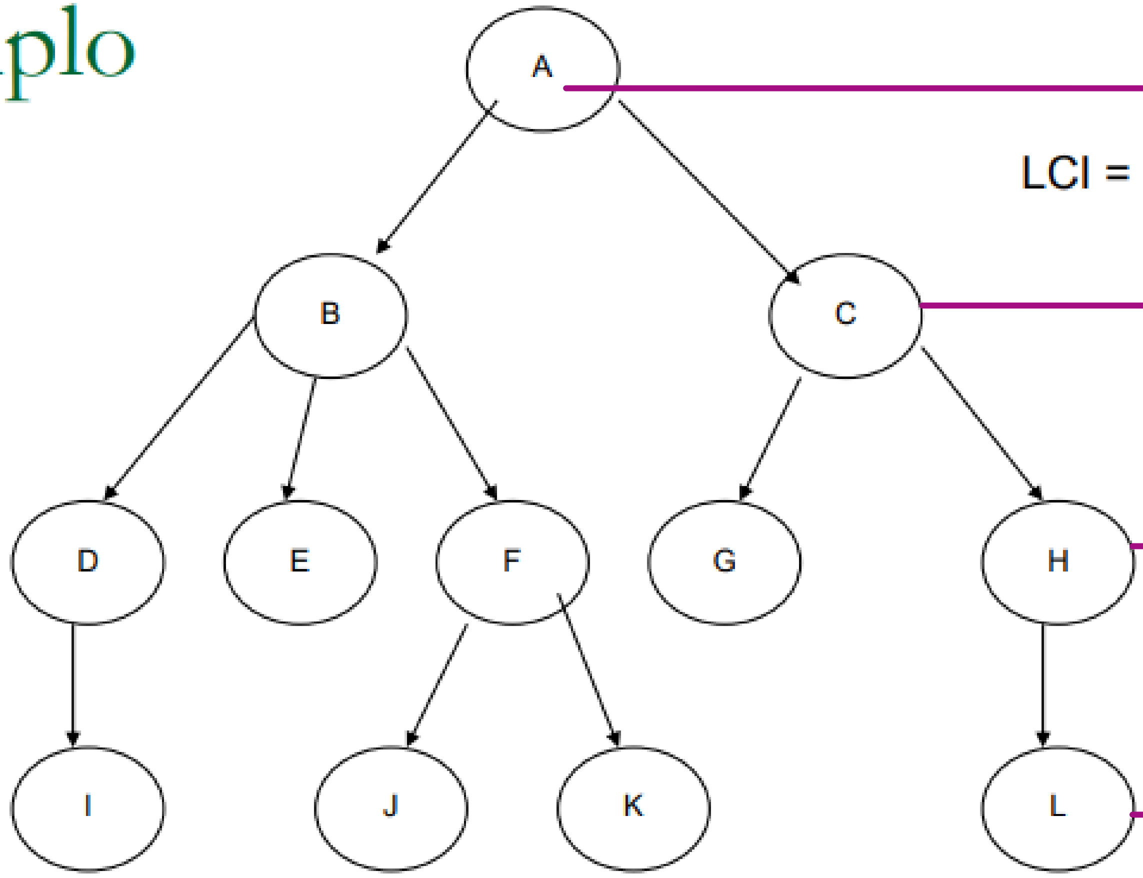
LONGITUD DE CAMINO INTERNO.

- La longitud de camino interno es la suma de las longitudes de camino de todos los nodos del árbol. Es importante por que permite conocer los caminos que tiene el árbol. Puede calcularse por medio de la siguiente fórmula:

$$LCI = \sum_{i=1}^{h} ni * i$$

- donde ‘i’ representa el nivel del árbol, ‘h’ su altura y ‘ni’ el número de nodos en el nivel ‘i’.

Ejemplo



$$LCI = \sum_{i=1}^{h} ni * i$$

1*1=1

2*2=4

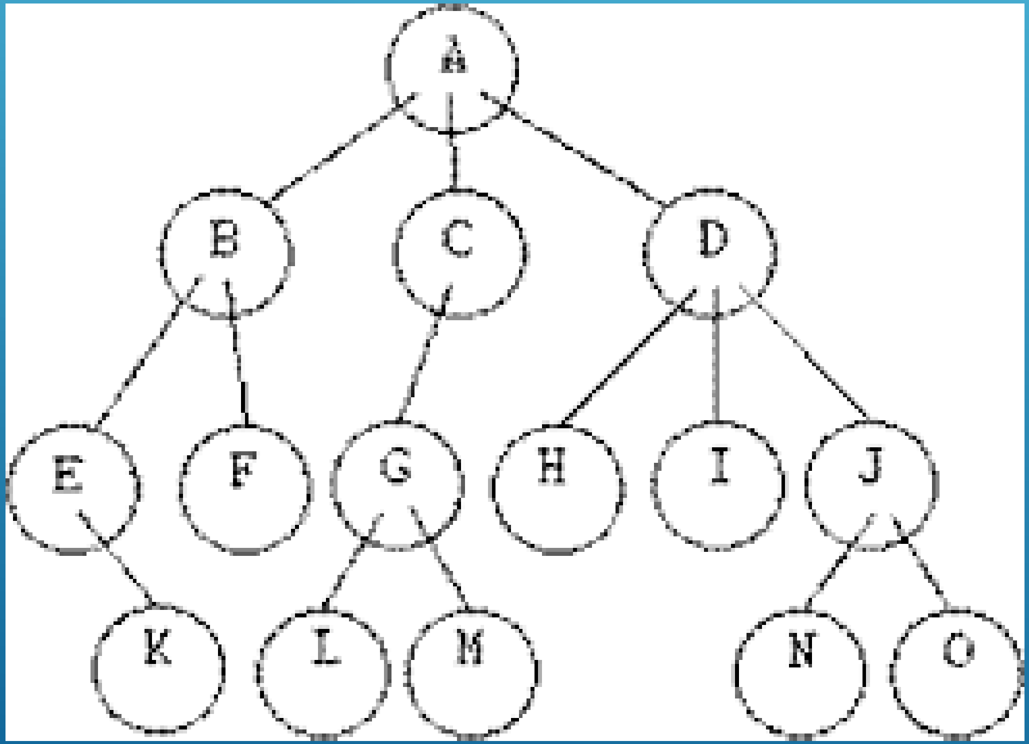
3*5=15

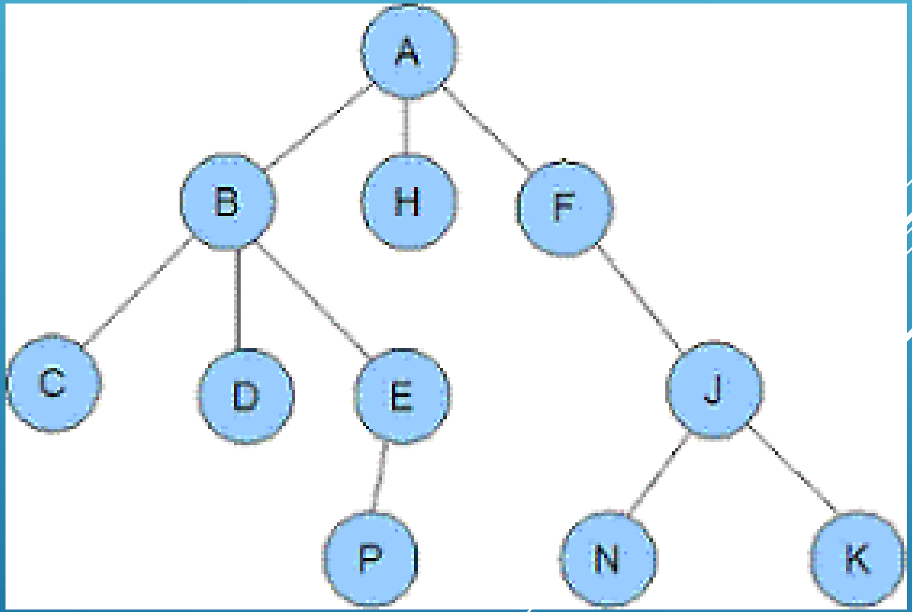
4*4=16

LCI=1*1+2*2+3*5+4*4=1+4+15+16=36

ACTIVIDAD

Determine la Longitud del Camino Interno, hojas, nodos interiores





MEDIA DE LA LONGITUD DE CAMINO INTERNO (LCIM)

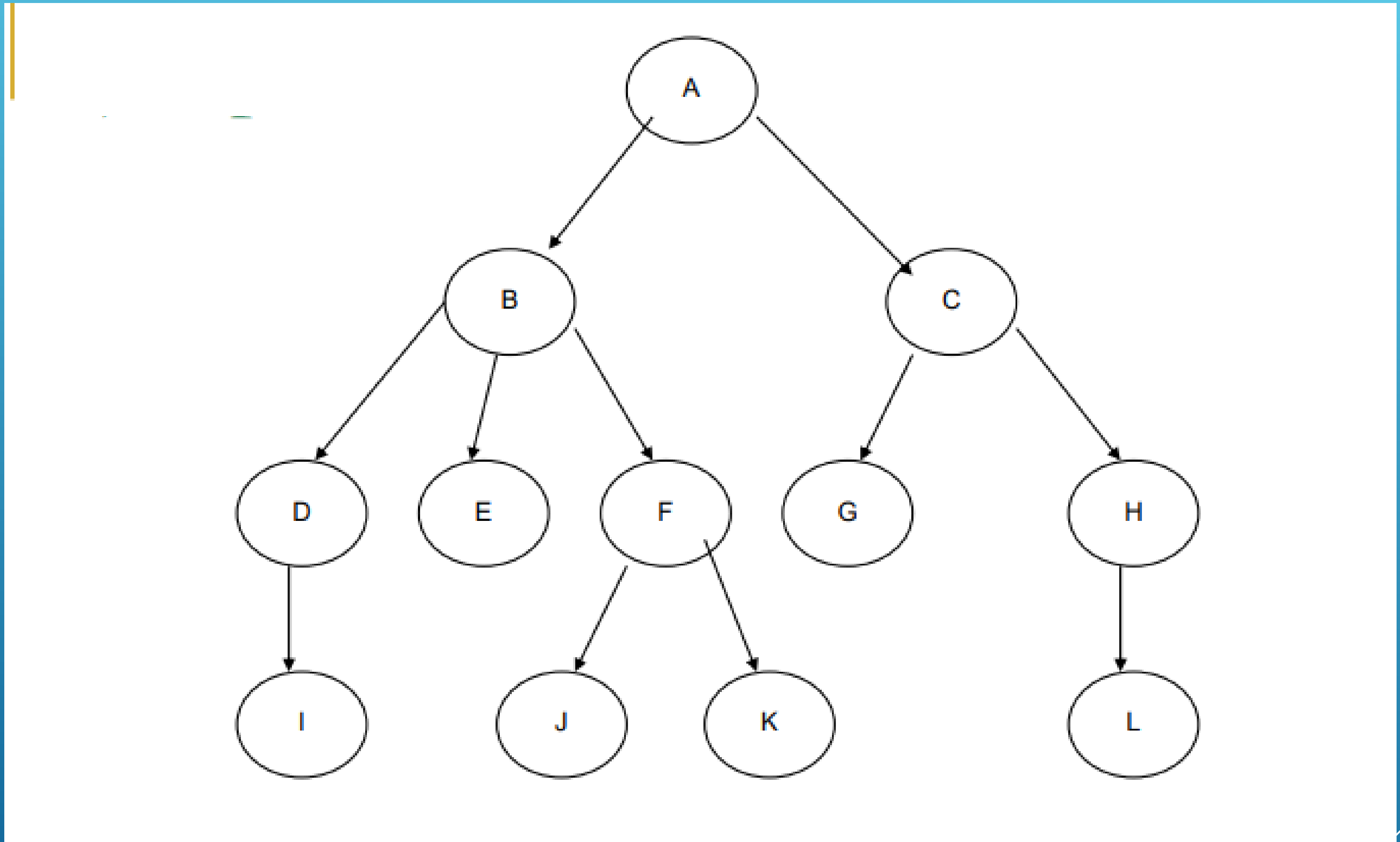
- Se calcula dividiendo la LCI entre el número de nodos del árbol (n).
- $$LCIM = LCI / n$$
- Y significa el número de arcos que deben ser recorridos en promedio para llegar, partiendo de la raíz, a un nodo cualquiera del árbol.
- La LCIM del árbol anterior es:
  - $LCIM = 36 / 12 = 3$

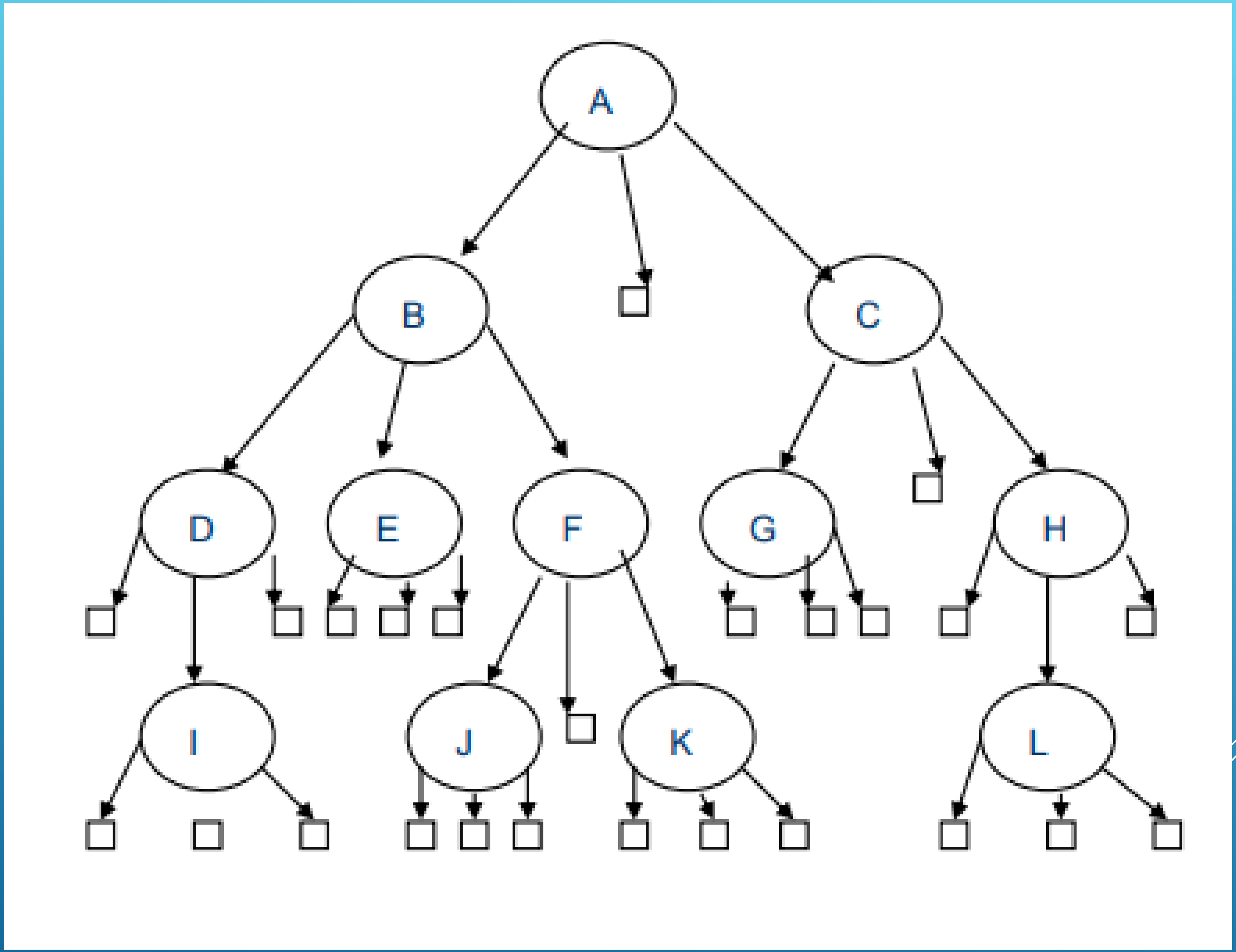
ARBOL EXTENDIDO

Árbol extendido es aquel en el que el número de hijos de cada nodo es igual grado del árbol.

Si alguno de los nodos del árbol no cumple con esta condición entonces debe incorporársele al mismo nodos especiales; tantos como sea necesario para satisfacer la condición.

Ejemplo: El número de nodos especiales de este árbol es 25, del árbol de grado 3




A
B
C
D
E
F
G
H
I
J
K
L

ARBOL EXTENDIDO MÍNIMO

Definición: Si *G* es una gráfica ponderada conectada, el árbol extendido de *G* con el peso total más pequeño (es decir, la suma de los pesos de sus aristas) recibe el nombre de árbol extendido mínimo de *G*.

Dos algoritmos populares para construir árboles extendidos mínimos se presentan a continuación.

Algoritmo de Prim

Paso 1

Cualquier arista de una gráfica dada *G* con el peso más pequeño se elige y se pone en el árbol extendido.

Paso 2

Se agregan sucesivamente las aristas de peso mínimo de la gráfica que son incidentes a un vértice ya en el árbol y que no forman un circuito con las aristas ya en el árbol.

Paso 3

El procedimiento se interrumpe cuando se han agregado (n – 1) aristas.

De manera equivalente, es posible seguir el procedimiento de trabajo dado de la manera siguiente:

Sean v_1, v_2, ..., v_n los vértices de la gráfica G dada. La matriz de peso W de G se forma con ∞ y se selecciona de esta lista la arista v_1, se listan las aristas incidentes elegibles sobre v_1 y se selecciona de esta lista la arista v_1v_j (por ejemplo) con el menor peso. Después de eliminar v_1v_j de la lista, se listan todas las nuevas elegibles incidentes sobre v_j y se selecciona de la lista de aristas elegibles (consistentes en el viejo conjunto excluyendo v_1v_j y el nuevo conjunto) la arista con el menor peso. Este proceso se repite hasta que los n vértices se conectan mediante (n – 1) aristas. El árbol extendido requerido es el consistente en las aristas seleccionadas.

Ejemplo: Utilice el algoritmo de Prim para encontrar un árbol extendido mínimo correspondiente a la gráfica ponderada de la figura 8.6.

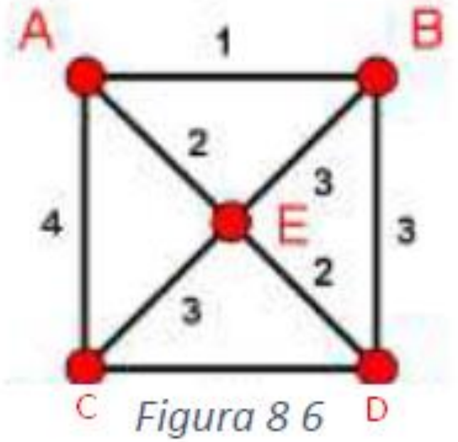


Figura 8 6

La matriz ponderada de la gráfica es

$$w = \begin{array}{c c} & \begin{array}{ccccc} A & B & C & D & E \end{array} \\ \begin{array}{c} A \\ B \\ C \\ D \\ E \end{array} & \begin{pmatrix} \infty & 1 & 4 & \infty & 2 \\ 1 & \infty & \infty & 3 & 3 \\ 4 & \infty & \infty & 1 & 3 \\ \infty & 3 & 1 & \infty & 2 \\ 2 & 3 & 3 & 2 & \infty \end{pmatrix} \end{array}$$

Número de iteración (i)	*Aristas elegibles después de la iteración i – ésima*	*Arista seleccionada con peso*
1	*AB (1), AC (4), AE (2)*	*AB (1)*
2	*BD (3), BE (3)*	*AE (2)*
3	*EC (3), ED (2)*	*ED (2)*
4	*DC (1)*	*DC (1)*

Puesto que los 5 vértices están conectados por 4 aristas que no forman un circuito, las aristas del árbol extendido mínimo son *BA, AE, ED, y DC*. El árbol extendido mínimo con peso 6 se ilustra en la figura 8.7.

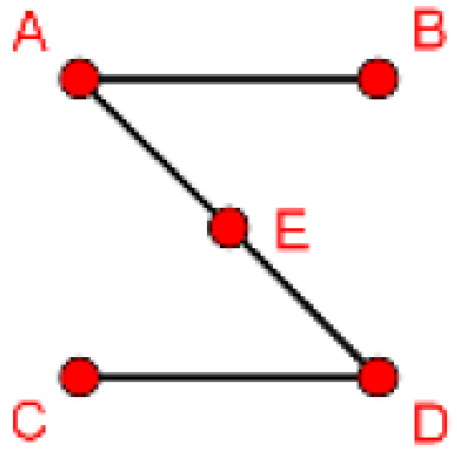


Figura 8 7

Mediante el algoritmo de Prim determina el árbol extendido mínimo correspondiente a la siguiente gráfica ponderada.

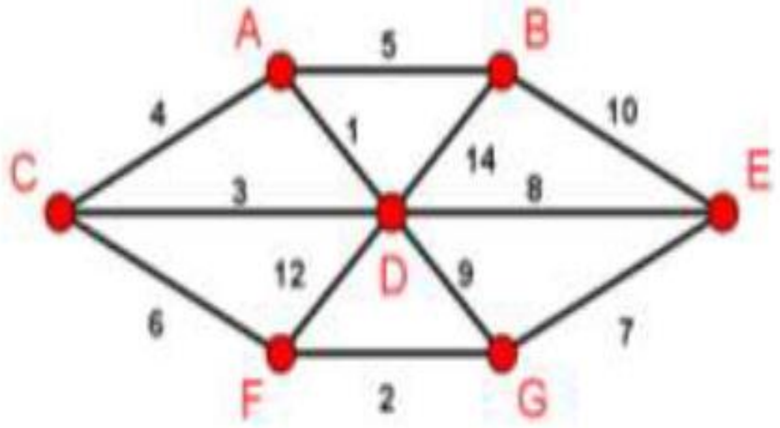


Figura 8 8

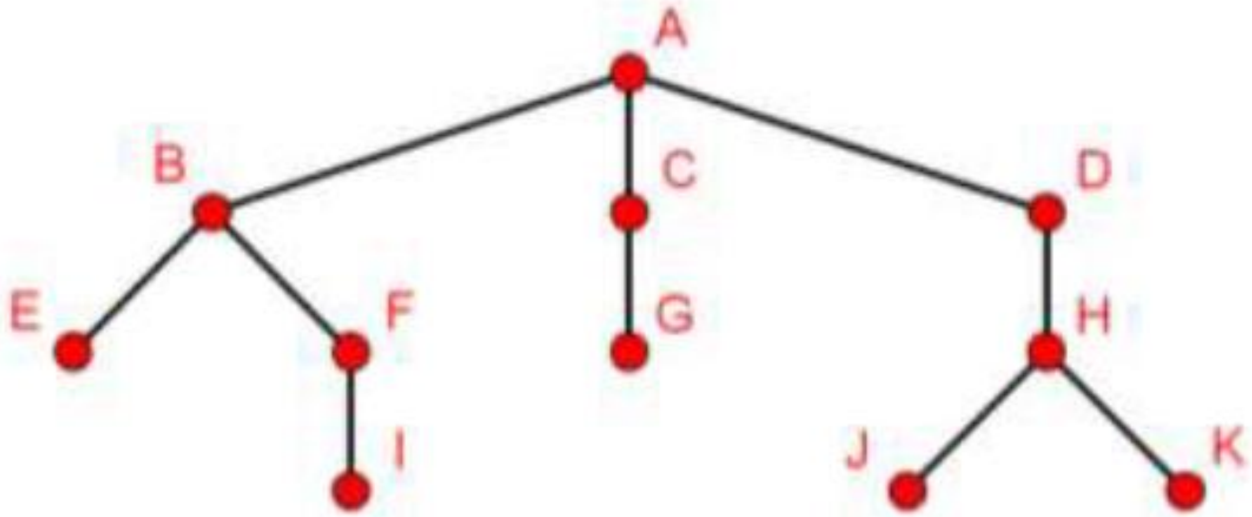
La matriz ponderada de la gráfica es

$$
\begin{array}{c} \\ A \\ B \\ C \\ D \\ E \\ F \\ G \end{array}
\begin{array}{c}
\begin{array}{ccccccc} A & B & C & D & E & F & G \end{array} \\
\begin{pmatrix}
\infty & 5 & 4 & 1 & \infty & \infty & \infty \\
5 & \infty & \infty & 14 & 10 & \infty & \infty \\
4 & \infty & \infty & 3 & \infty & 6 & \infty \\
1 & 14 & 3 & \infty & 8 & 12 & 9 \\
\infty & 10 & \infty & 8 & \infty & \infty & 7 \\
\infty & \infty & 6 & 12 & \infty & \infty & 2 \\
\infty & \infty & \infty & 9 & 7 & 2 & \infty
\end{pmatrix}
\end{array}
$$

ÁRBOLES N - ARIOS

Definición: un árbol en el cuál un vértice particular se designa como la raíz del árbol recibe el nombre de árbol de raíz.

El árbol raíz se dibuja con la raíz en la parte superior. En la siguiente figura, A es la raíz del árbol y está a nivel cero. Los vértices B, C, D están en el nivel 1, E, F, G, H se encuentran en el nivel 2 e I, J, K están en el nivel 3.



La atura del árbol es 3.

Los vértices E, F e I son descendientes de B.

Similarmente, H, J y K son descendientes de D, E y F son hijos de B, J y K son hijos de H.

Los vértices E, i, G, J y K son hojas del árbol.

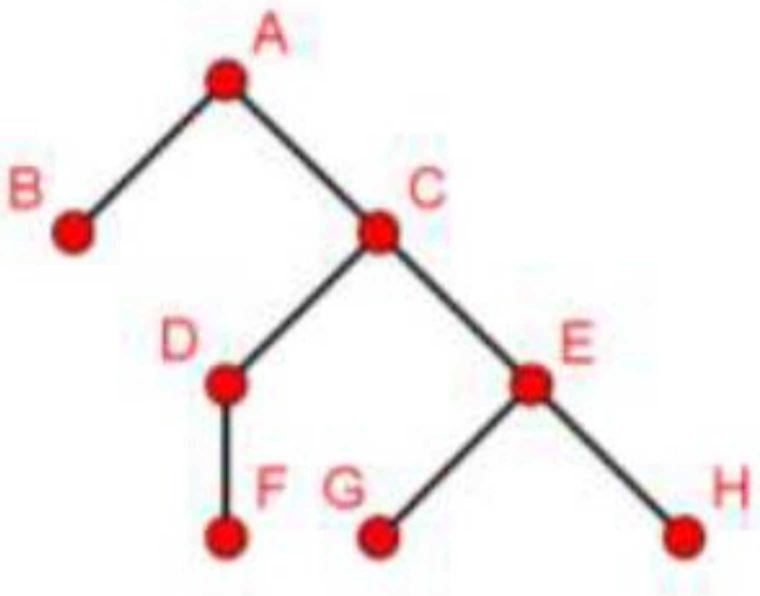
Los vértices A, B, F, C, D y H son vértices internos del árbol.

ÁRBOL BINARIO

Una clase especial de árboles raíz, son los llamados árboles binarios, es de mucha utilidad en la ciencia computacional.

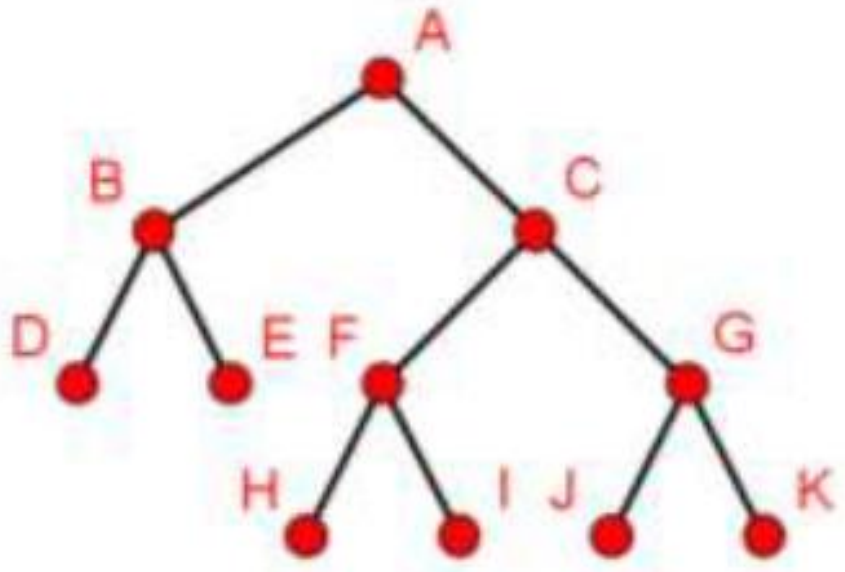
Definición: si todo vértice interno de todo árbol raíz tiene exactamente a lo más de 2 hijos, el árbol se denomina árbol binario completo o árbol binario.

En otras palabras, un árbol binario completo es aquel en el cual hay exactamente un vértice (raíz) de grado 2 y cada uno de los vértices restantes es de grado 1 o 3. En las siguientes figuras T_1 es un árbol binario, en tanto que T_2 es un árbol binario completo.



T_1

Figura 8 12



T_2

Figura 8 11

PROPIEDADES DE LOS ÁRBOLES BINARIOS

Propiedad 1

El número n de vértices del árbol binario completo es impar y el número de vértices colgantes (hojas de árbol es igual:

$$\frac{(n+1)}{2}$$

Propiedad 2

La altura mínima de un árbol binario de n vértices es igual a $[\log_2(n+1)-1]$, donde $[x]$ denota el entero más pequeño mayor que o igual a x

Ejemplo: Dibuje los árboles binarios de 11 vértices con altura mínima y máxima. Determine también la longitud de la trayectoria de ambos árboles.

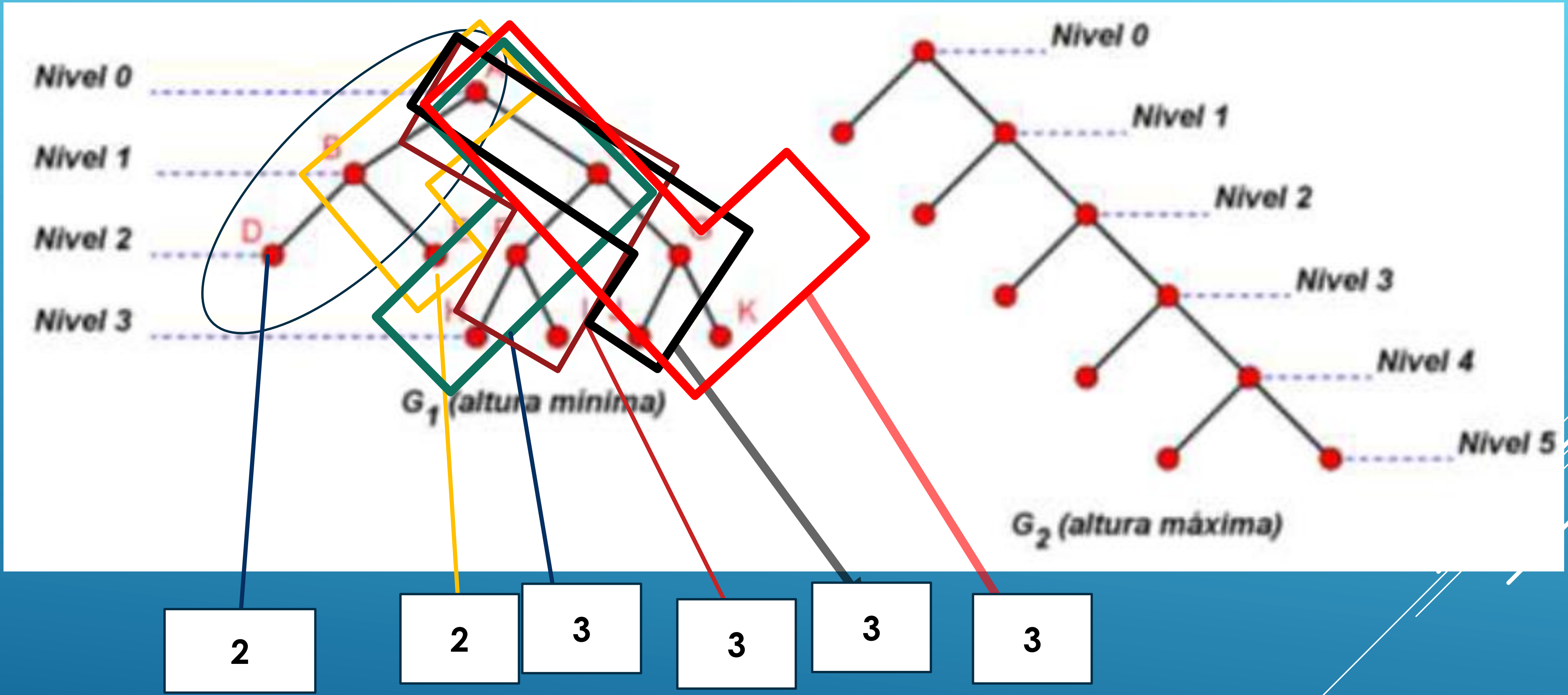
Por la propiedad (2) de los árboles binarios, la altura mínima del árbol binario de 11 vértices = $[\log_2(12) - 1] = [3{,}5850 - 1] = [2{,}5850] = 3$, (ya que $[x]$= el entero menor $\geq x$)

Para dibujar el árbol binario con la altura máxima se deben tener exactamente dos vértices en cada nivel (excepto en el nivel cero)

$$\therefore Altura\ máxima = \frac{11 - 1}{2} = 5$$

Los árboles binarios requeridos se indican en la siguiente figura.

La suma de las longitudes de la trayectoria desde la raíz hasta los vértices terminales de un árbol binario recibe el nombre de longitud de la trayectoria del árbol.



Para G_1 la longitud de la trayectoria = 2 + 2 + 3 + 3 + 3 + 3 = 16

Para G_2 la longitud de la trayectoria = 1 + 2 + 3 + 4 + 5 + 5 = 20

RECORRIDO DE ÁRBOLES

Una de las operaciones más comunes efectuadas sobre gráficas de árboles es la del recorrido. El recorrido de un árbol es un proceso para recorrer (desplazarse a lo largo) un árbol de manera sistemática a fin de que cada vértice se visite y procese exactamente una vez. Hay tres métodos para recorrer un árbol binario, a saber, recorridos de preorden, de inorden y de posorden.

Definiciones

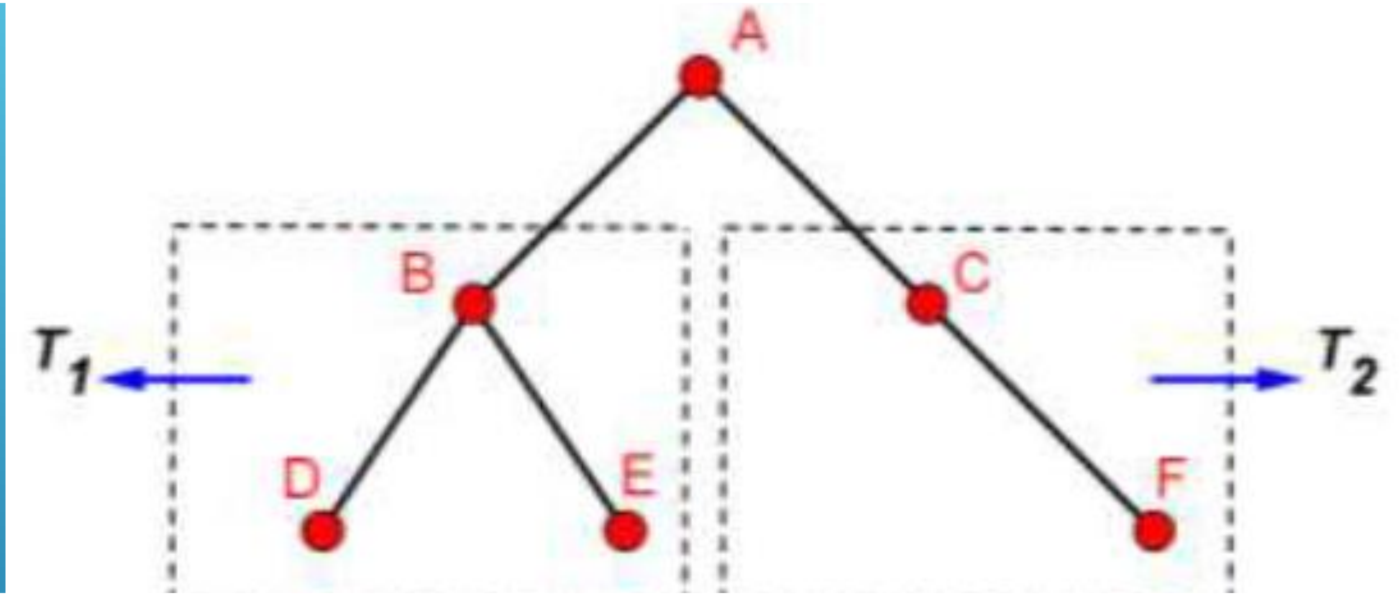
Sean T_1, T_2, ..., T_n los subárboles de un árbol binario dado y la raíz R de izquierda a derecha. El proceso de visitar la raíz R primero y de recorrer T_1 en preorden, después T_2 en preorden y así hasta llegar a que T_n se recorrerá en preorden recibe el nombre de recorrido preorden.

El proceso de recorrer T_1 primero en inorden y después visitar la raíz R y continuar el recorrido de T_2 en inorden, T_3 en inorden, etc., hasta que T_n se recorra en inorden, recibe el nombre de recorrido inorden.

El proceso de recorrer T_1 primero en posorden, después T_2 en posorden, etc., T_n en posorden y por último visitar la raíz R se llama recorrido posorden.

Ejemplo, considere los tres métodos de recorrido de árboles binarios que se muestran en la siguiente figura.

Así, el recorrido posorden de T es $D\ E\ B\ F\ C\ A$.



De tal modo, el recorrido inorden de T es $D\ B\ E\ A\ C\ F$.

De tal modo, el recorrido de preorden de T es $A\ B\ D\ E\ C\ F$.

T_1 y T_2 son los subárboles del árbol binario dado T con B y C como las raíces respectivas.

- El recorrido preorden de T visita la raíz A primero y después recorre T_1 y T_2 en preorden. El recorrido de preorden de T_1 visita la raíz B y después D y E en ese orden.

El recorrido de preorden de T_2 visita la raíz C y después F.

De tal modo, el recorrido de preorden de T es A B D E C F.

- El recorrido inorden de T recorre T_1 primero en inorden, después visita la raíz A y finalmente recorre T_2 en inorden.

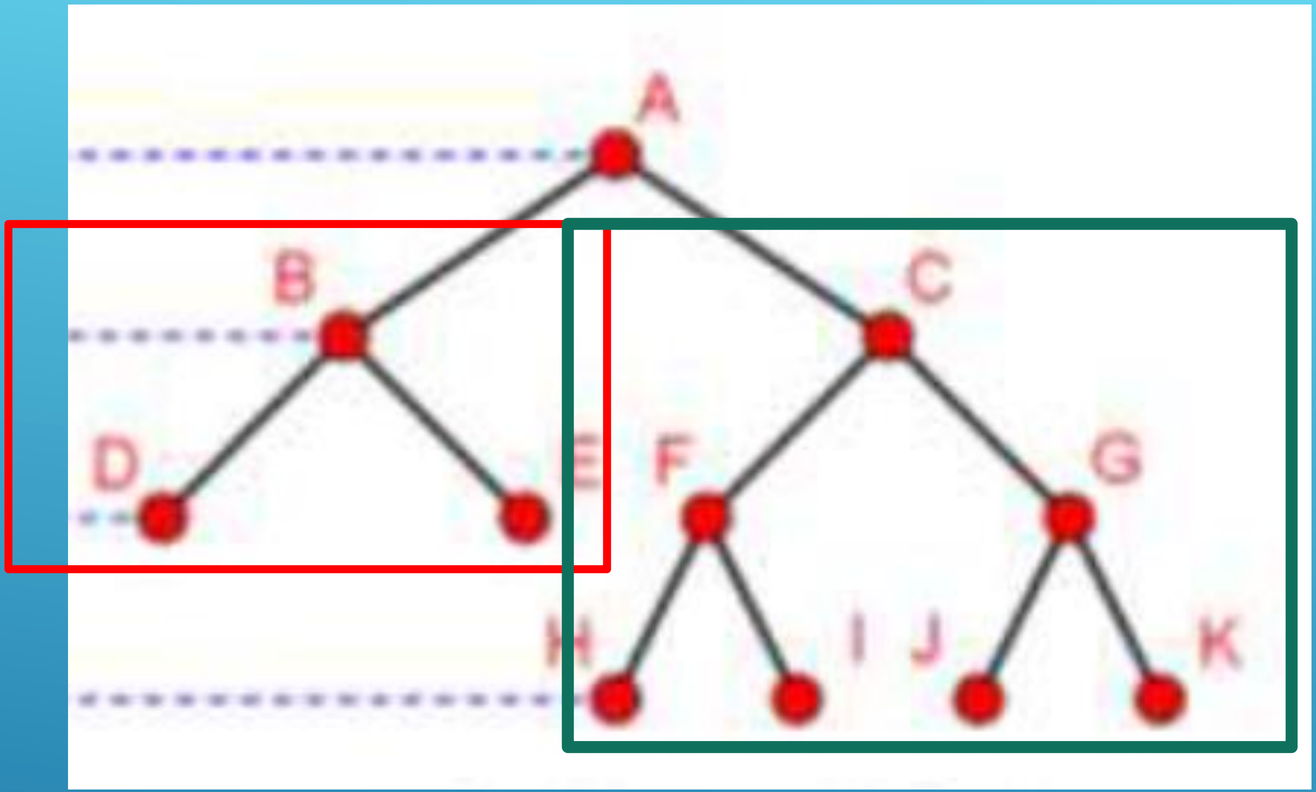
Sin embargo, el recorrido inorden de T_1 procesa D, B y E en ese orden y después el recorrido inorden de T_2 procesa C y luego F.

De tal modo, el recorrido inorden de T es $D\ B\ E\ A\ C\ F$.

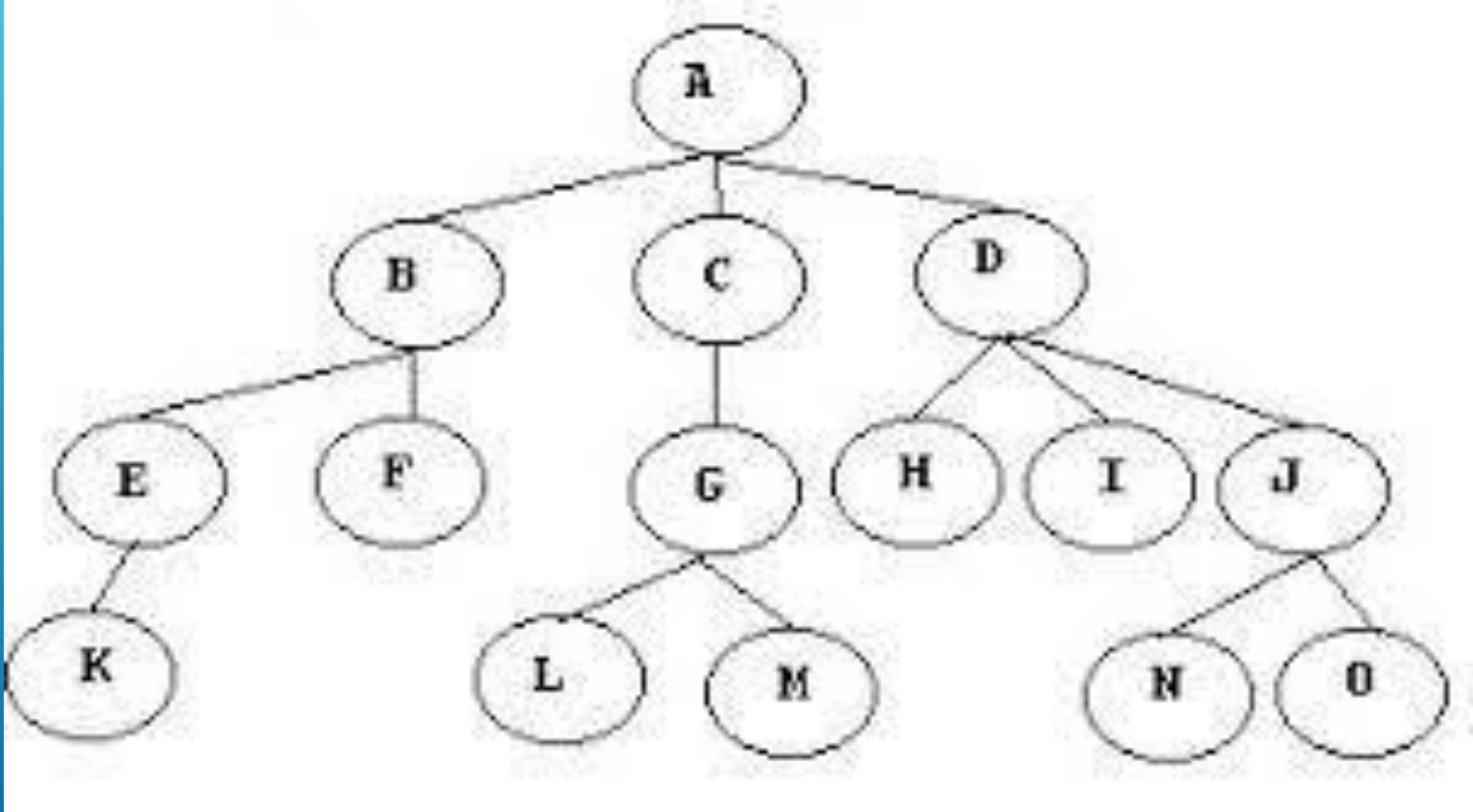
- El recorrido posorden de T procesa T_1, luego T_2 en posorden y finalmente visita A.

Pero el recorrido posorden de T_1, procesa D, E y B en ese orden y el recorrido posorden de T_2 procesa F y luego C.

Así, el recorrido posorden de T es $D\ E\ B\ F\ C\ A$.



PREORDEN: ABDECFHIGJK
INORDEN: DBEACFHIGJK
POSORDEN: DEBHIFJKGCA


A
B
C
D
E
F
G
H
I
J
K
L
M
N
O

ÁRBOLES DE EXPRESIÓN

Los árboles binarios pueden utilizarse para representar expresiones algebraicas, ya que tales representaciones facilitan la evaluación por computadora de las expresiones. En la representación de expresiones mediante árbol binario, los vértices terminales (hojas) se etiquetan con números o variables, mientras que los vértices internos se etiquetan con las operaciones, por ejemplo, suma (+), resta (-), multiplicación (*), división (/) y exponenciación (↑). La operación en cada vértice interno opera sobre sus subárboles izquierdo y derecho de izquierda a derecha.

Es posible representar expresiones de tres maneras diferentes utilizando árboles binarios. Éstas se conocen como formas infijo, prefijo y posfijo de una expresión.

Notación infijo

La manera estándar de representar una expresión en la cual el operador se ubica entre sus operadores se denomina la forma infijo de la expresión.

La forma infijo de una expresión algebraica corresponde al recorrido inorden del árbol binario que representa a la expresión; proporciona la expresión original con los operadores y operaciones en las mismas posiciones. Para evitar ambigüedad en la notación infijo se incluyen un par de paréntesis en cada operación.

Ejemplo, la expresión $((A + B)*(C/D))$ se representa mediante el árbol binario que se muestra en la siguiente figura.

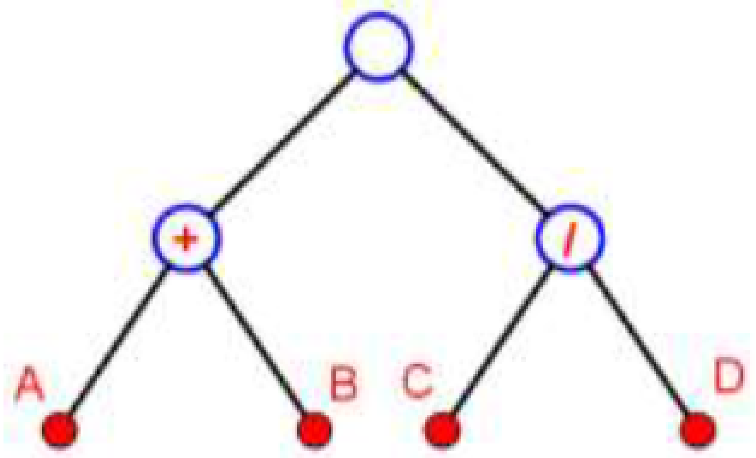


Figura 8 20

Notación prefijo

La forma prefijo de una expresión algebraica representada por un árbol binario corresponde al recorrido preorden del árbol. La expresión en la notación prefijo es inambigua y por ello no se necesita utilizar paréntesis en esta forma. Las expresiones escritas en forma prefijo se dice que están en notación polaca, nombre que se asigna en honor al lógico polaco Jan Lukasiewicz.

Ejemplo, la forma prefijo de la expresión representada por el árbol binario del anterior ejemplo es: $*+AB/CD$.

Notación posfijo

La forma posfijo de una expresión algebraica representada por un árbol binario corresponde al recorrido posorden del árbol. Como la expresión en la notación posfijo es inambigua, no se requieren paréntesis en esta forma. También se dice que las expresiones escritas en la forma posfijo están en notación polaca inversa.

Ejemplo, la forma posfijo de la expresión representada por el árbol binario dado en la figura anterior es $AB + CD/*$

Ejemplo, Representa la expresión *((a – c)*d)/(a + (b – d))* como un árbol binario y escriba las formas prefijo y posfijo de la expresión.

El árbol binario para la expresión puede construirse de abajo hacia arriba. Primero se construyen los subárboles para las expresiones dentro de los paréntesis internos, a saber, $a - c$ y $b - d$ como se indica en la figura 8.21

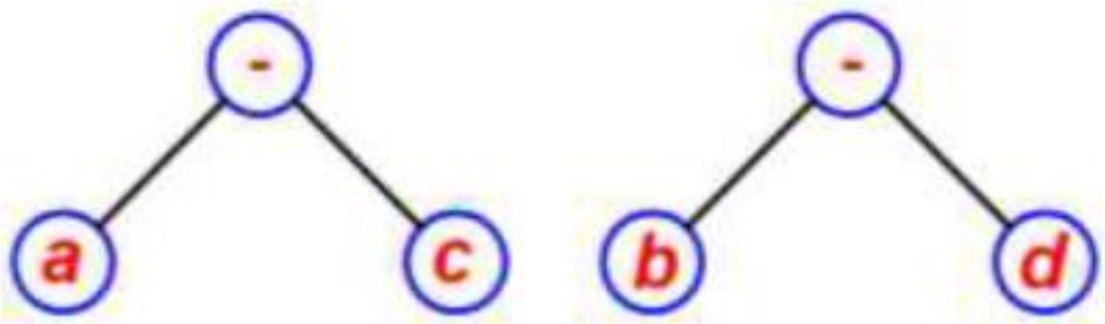


Figura 8 21

Luego éstas se incorporan como parte del subárbol más grande que representa a (a – c)*d y a + (b – d), lo que se muestra en la figura 8.22

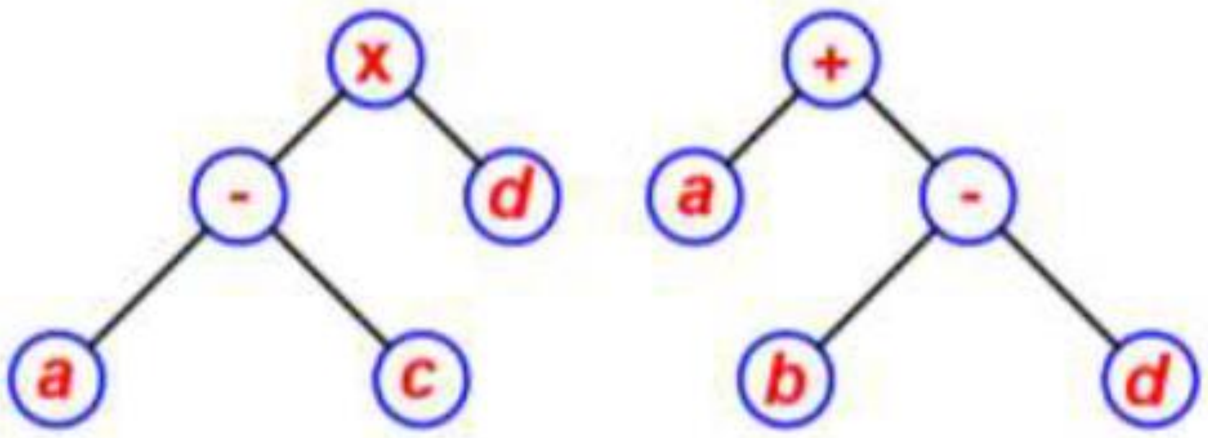


Figura 8 22

Por último, los subárboles dados en la figura 8.22 se combinan para formar el árbol requerido que representa la expresión dada. El árbol binario requerido se ilustra en la figura 8.23

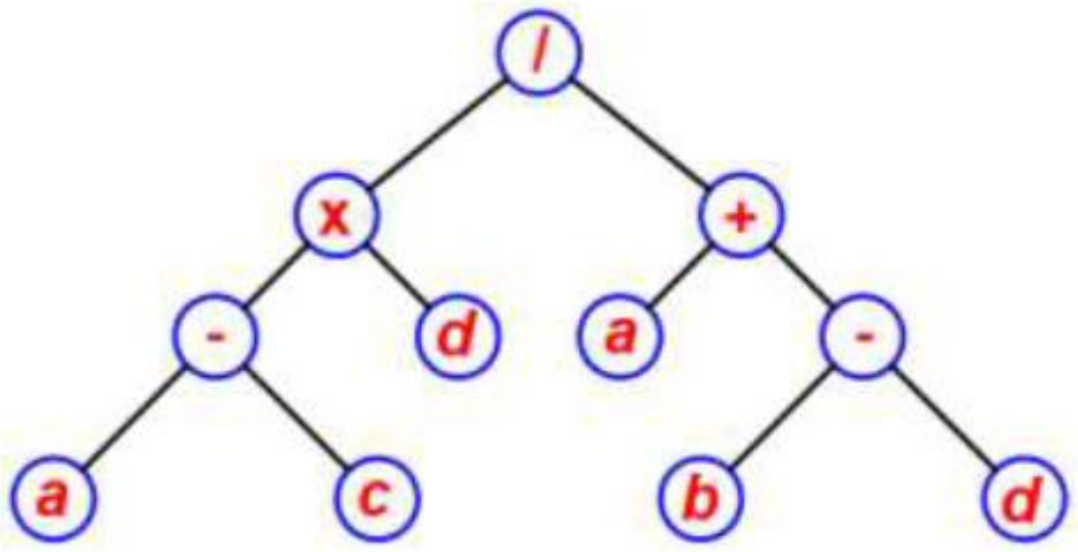


Figura 8 23

La expresión totalmente entre paréntesis, a saber (((a – c)*d)/(a + (b – d))), es la forma infijo de la expresión.

Forma prefijo, esta se obtiene visitando los vértices utilizando el recorrido preorden.

Etapa (1) /, *, +

Etapa (2) /, *, -, d, +, a, -

Etapa (3) /*-acd+a-bd, que es la forma prefijo-requerida.

Forma posfijjo, ésta se obtiene visitando los vértices utilizando el recorrido posorden.

Etapa (1) *, +, /

Etapa (2) -, d, *, a, -, +, /

Etapa (3) ac – d*abd-+/, que es la forma posfijo-requerida

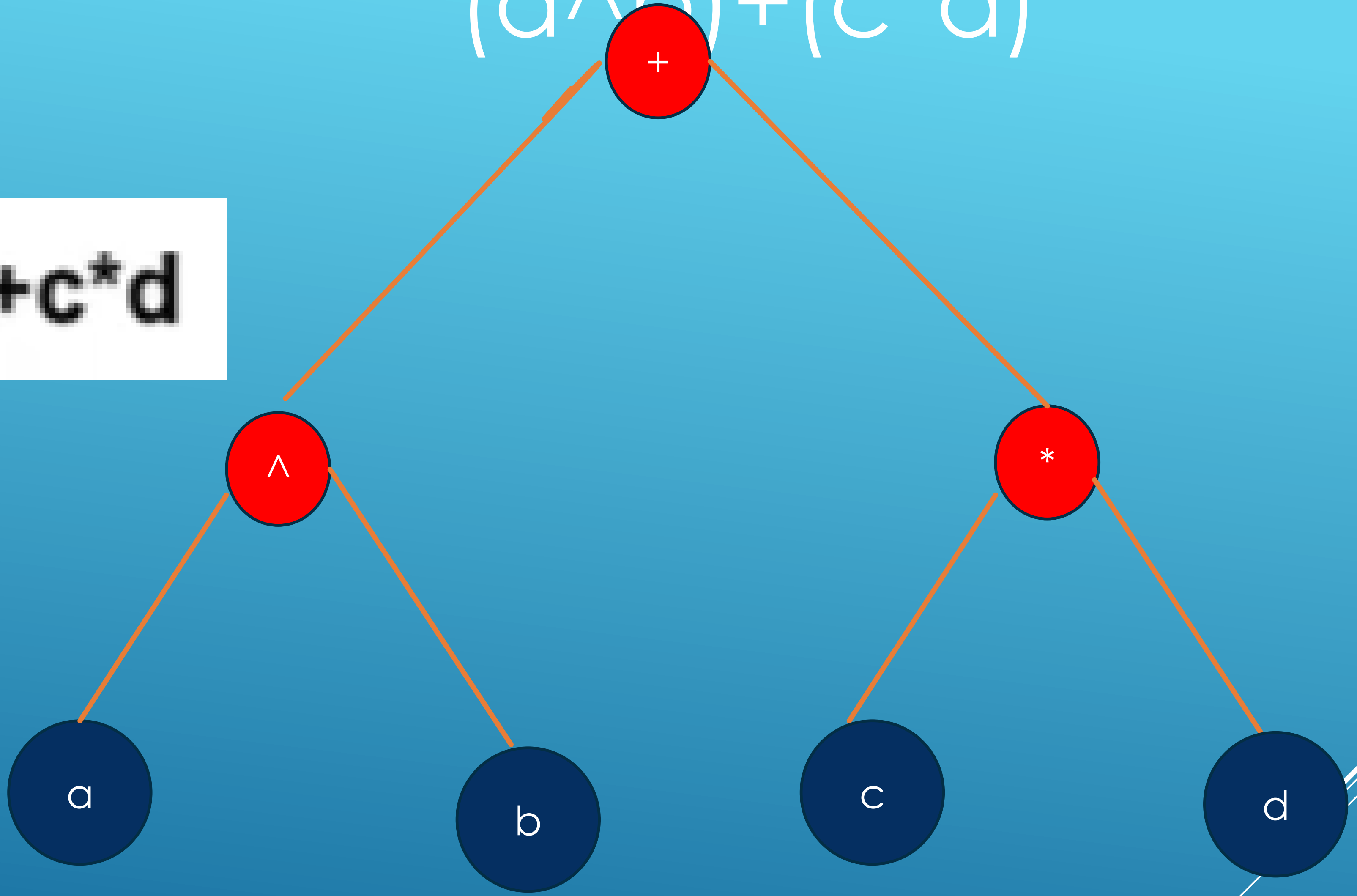
CARACTERÍSTICAS A TOMARSE EN CUENTA EN LA CONSTRUCCIÓN DE UN ÁRBOL DE EXPRESIONES

- La raíz siempre debe ser un operador.
- Las hojas siempre deben ser operandos.
- Los nodos deben estar etiquetados por operadores.
- Si un operador tiene mayor prioridad que la raíz se coloca como hijo.
- Si un operador tiene igual o menor prioridad que un nodo se coloca como padre.
- Un nodo puede contener como hijo otro subárbol que contiene una pequeña expresión.

EJEMPLO REPRESENTE

a^b+c*d

(a^b)+(c*d)



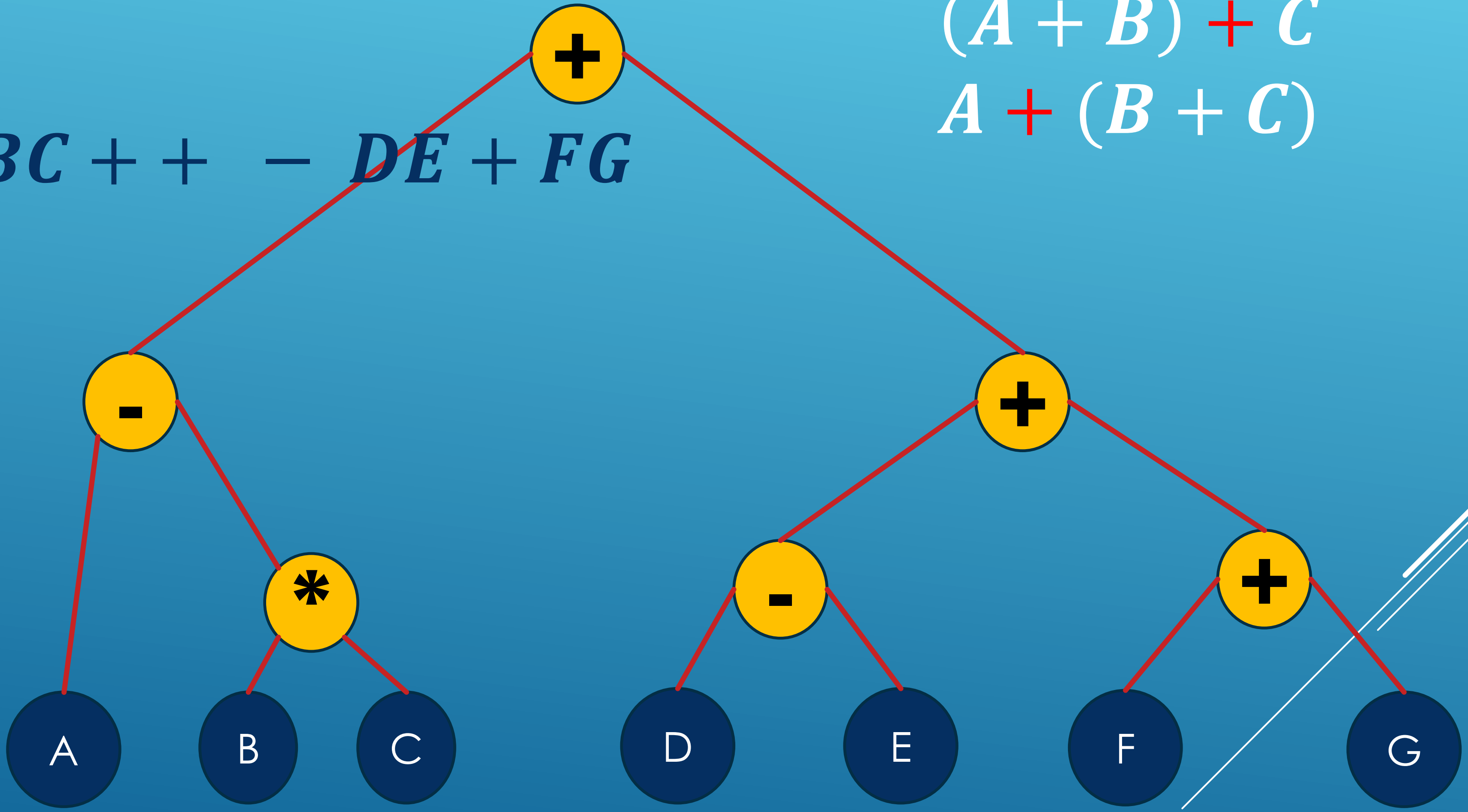
((A-(B*C))+((D-E)+(F+G)))

$Posfijo:\ ABC * -DE - FG ++ +$

$Prefijo$ =+-A*BC+-DE+FG

$$A + B + C$$
$$(A + B) + C$$
$$A + (B + C)$$

$Infijo:\ A -* BC + + \ \ - \ DE + FG$



- $\left((a+c)-\left(\frac{b}{d}\right)\right)*\left(f-\left(\frac{g}{h}\right)\right)$