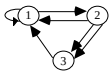
TEMA 5
El tipo grafo

PROGRAMACIÓN Y ESTRUCTURAS DE DATOS

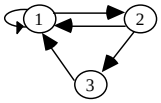
Tipo grafo

1. Concepto de grafo y terminología (I)

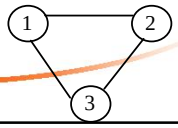
- Estructura de los grafos
  - cada nodo puede tener más de un sucesor y más de un predecesor
- Clasificación
  - MULTIGRAFO, no tiene ninguna restricción (arcos reflexivos y múltiples ocurrencias del mismo arco)



  - DIGRAFO, no hay múltiples ocurrencias del mismo arco



  - GRAFO NO DIRIGIDO, hay relaciones irreflexivas y simétricas entre nodos (todos sus arcos no orientados→aristas)



1. Concepto de grafo y terminología (II)

- Definición de grafo
  - Un **grafo G** consiste en dos conjuntos V y A donde: G=(V, A)
    - 1) V es un conjunto finito no vacío de vértices o nodos.
    - 2) A es un conjunto de aristas o arcos, tales que cada arista a_i de A está identificada por un único par (v_j, v_k) de nodos de V, denotada por $a_i = (v_j, v_k)$
- Un Arco Orientado o ARCO es aquél en que el orden de los vértices es importante: $<v_j, v_k>$
- Arco No Orientado o ARISTA: orden de los vértices no importa. Relaciones simétricas: (v_j,v_k)
- El **número máximo de aristas o arcos** que pueden existir en un grafo de *n* vértices sin contar las aristas que unen un vértice consigo mismo (v_i, v_i) son:

  Grafo no dirigido: $n\,(n-1)\,/\,2$

  Grafo dirigido: $n\,(n-1)$

1. Concepto de grafo y terminología (III)

- Los vértices ***v1*** y ***v2*** son adyacentes si *(v1,v2)* es una arista en *A(G)*, la cual diremos que es **incidente sobre** ambos vértices.
- En el caso de grafos dirigidos <*v1,v2*,> será **incidente a** *v1* y *v2*. Además diremos que *v1* es **adyacente hacia** *v2*, y *v2* es **adyacente desde** *v1*.
- Adyacencia de un vértice: es el conjunto de nodos del grafo tales que existe un arco que los relacione:

  $Ay(x) = \{vi \in V / \exists (x, vi) \in A\}$
- Para los digrafos podemos hablar de:

  - **Adyacencia de entrada**, es el conjunto de nodos para los que hay una arista en el grafo que relaciona a ambos, con destino el vértice *x*:

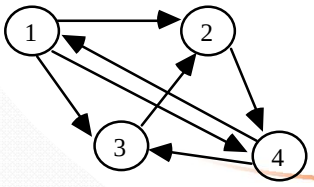
  $AyE(x) = \{vi \in V / \exists \langle vi, x\rangle \in A\}$

  -**Adyacencia de salida** es:

  $AyS(x) = \{vi \in V / \exists \langle x, vi\rangle \in A\}$
- **Grado de entrada o ingrado (gradoE):** cardinal del conjunto *Adyacencia de Entrada.*
- **Grado de salida (gradoS):** cardinal del conjunto *Adyacencia de Salida.*
- **Grado de un Vértice:** $grado(v) = CARD(Ay(v)) = gradoE + gradoS$

1. Concepto de grafo y terminología (IV)

- Un **camino desde *vp* hasta *vq*** en el grafo *G* es una secuencia de vértices *vp, v1,v2, ..., vn, vq* tal que *(vp, v1) (v1,v2), (v2, v3), ..., (vn, vq)* son aristas de *A(G).*
- Se llama **longitud de un camino** al número de aristas del mismo.
- Se llama **camino simple** a aquél en el que todos los vértices, excepto el primero y el último, son distintos.
- Un **ciclo** es un camino simple en el que el vértice primero y último coinciden.

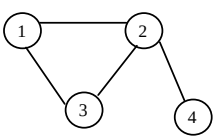


Camino 3,2,4,3
es un ciclo de longitud 3

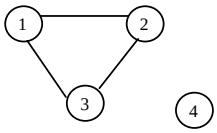
Camino 3,2,4,1,4,3,
no es un ciclo

1. Concepto de grafo y terminología (V)

- Un **grafo no dirigido** se dice que es **conexo**: si $\forall vi, vj \in V(G)$ existe un camino de *vi* a *vj* en *G*

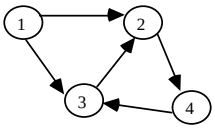


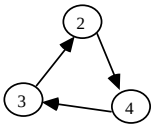
CONEXO



NO CONEXO

- Un **subgrafo** de un grafo *G=(V, A)*, es un grafo *G'=(V', A')* tal que $V' \subset V$ y $A' \subset A$





SUBGRAFO

- Un **árbol extendido** de un grafo *G=(V, A)* es un subgrafo *T= (V', A')* de *G* tal que *T* es un árbol y *V'=V*

2. Especificación algebraica (I)

MODULO GENERICO ModuloGrafo
MODULO Grafo **USA** Boolean
PARAMETRO
TIPO Vértice, Ítem
SINTAXIS
ErrorÍtem ( ) → Ítem
TIPO
Grafo
SINTAXIS
Crear ( ) → Grafo
EsVacíoGrafo(Grafo) → Boolean
InsertarArista(Grafo, Vértice, Vértice, Ítem) → Grafo
RecuperarArista(Grafo,Vértice, Vértice) → Ítem
BorrarArista(Grafo, Vértice, Vértice) → Grafo
VAR G: Grafo; x, y, z, t: Vértice;
p, q: Ítem;

2. Especificación algebraica (II)

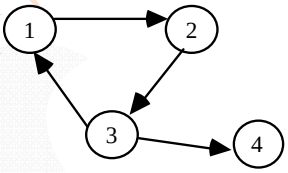
SEMANTICA

EsVacíoGrafo(Crear) ⇔ Cierto
EsVacíoGrafo(InsertarArista (G, x, y, p)) ⇔ Falso
BorrarArista(Crear, z, t) ⇔ Crear
BorrarArista(InsertarArista(G,x,y,p), z, t) ⇔
si (x == z) y (y == t)
entonces G //Multigrafo: BorrarArista(G, z, t)
sino InsertarArista(BorrarArista(G, z, t), x, y, p)
RecuperarArista(Crear, x, y) ⇔ ErrorÍtem
RecuperarArista(InsertarArista(G,x,y,p), z, t) ⇔
si (x == z) y (y == t)
entonces p
sino RecuperarArista(G, z, t)
InsertarArista (InsertarArista (G,x,y,p), z,t,q) ⇔
si (x == z) y (y == t)
entonces InsertarArista(G,z,t,q)
sino InsertarArista(InsertarArista(G,z,t,q), x, y, p)

FIN MODULO

3. Representación de grafos (I)

- Dado un grafo *G* con *n* vértices, *n≥1*, la **matriz de adyacencia de G** es una matriz *A* cuadrada bidimensional *n x n* tal que:
  A(i, j) = 1 ó CIERTO si existe *(vi, vj)*
  A(i, j) = 0 ó FALSO si no existe *(vi, vj)*

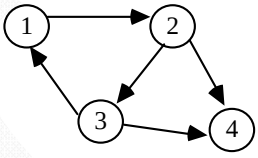


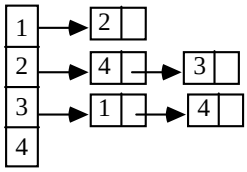
MATRIZ ADYACENCIA

$$\begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

3. Representación de grafos (II)

- En el caso que representásemos un grafo no dirigido podríamos observar cómo la matriz de adyacencia sería simétrica respecto la diagonal ante la equivalencia de cada arista con sus dos arcos. Además, la diagonal principal tendrá todos sus valores a cero, a causa de no permitirse relaciones reflexivas entre nodos.
- Una segunda posibilidad de representación es la **lista de adyacencia** donde se cambia la matriz por una lista para cada vértice *vi* en *G*, y en la que los nodos de cada lista contienen los vértices que son adyacentes desde el vértice *vi*.





LISTA DE ADYACENCIA

4. Grafos dirigidos

RECORRIDO EN PROFUNDIDAD O DFS (I)

- Generalización del preorden
- Recorre todos los vértices que sean accesibles desde v
- Cuando se hayan visitado todos se continúa desde otro vértice que no haya sido visitado

```
ALGORITMO DFS
ENTRADA : v : Vértice ; G : Grafo ;
ENTRADA-SALIDA:Visitados: conjunto vertices;
VAR : w: Vértice;
METODO
INSERTAR (visitados, v)
VISITAR (v)
Para todo w ∈ AyS (v)
  Si NO PERTENECE (w, visitados)
      DFS (w, G, visitados)
  fsi
fpara
fMETODO
```

4. Grafos dirigidos

RECORRIDO EN PROFUNDIDAD O DFS (II)

- Tipos de arcos:
- **DEL ARBOL**: los que conducen a nodos no visitados, y forman un **BOSQUE EXTENDIDO EN PROFUNDIDAD**.
- **DE RETROCESO**: los que van desde un vértice a uno de sus antecesores en el bosque extendido
- **DE AVANCE**: los que van de un vértice a un descendiente propio y no son arcos DEL ARBOL
- **DE CRUCE**: los que van de un vértice a otro que no es un antecesor ni descendiente

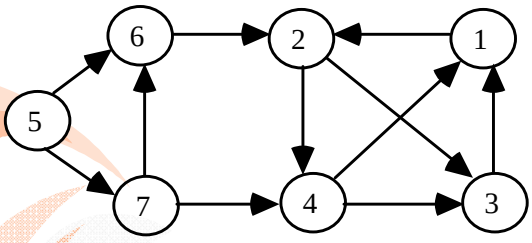
4. Grafos dirigidos

COMPONENTES FUERTEMENTE CONEXOS

- **Prueba de Aciclidad**: Si se encuentra un arco de retroceso durante la búsqueda en profundidad el grafo no será acíclico.
- **Componente Fuertemente conexo de un grafo dirigido**: Conjunto maximal de vértices en el cual existe un camino que va desde cualquier vértice del conjunto hasta cualquier otro vértice también del conjunto.
- **GRAFO DIRIGIDO FUERTEMENTE CONEXO**: aquel que tiene un sólo componente fuerte.

4. Grafos dirigidos

RECORRIDO EN PROFUNDIDAD O DFS (III)



(Lista de adyacencia de menor a mayor)

Recorridos DFS:

1,2,3,4
2,3,1,4
3,1,2,4
4,1,2,3
5,6,2,3,1,4,7

4. Grafos dirigidos

EJERCICIO RECORRIDO EN PROFUNDIDAD O DFS (IV)

- Dado el grafo anterior realizar el recorrido DFS partiendo del vértice 5 (suponed la lista de adyacencia ordenada de mayor a menor). Clasificar los arcos y dibujar el bosque extendido en profundidad

4. Grafos dirigidos

EJERCICIO 1. RECORRIDO EN PROFUNDIDAD O DFS

Sobre el siguiente grafo dirigido, responder:

A) Realiza el recorrido DFS(1), la adyacencia de salida ordenada de menor a mayor, y obtén el árbol extendido en profundidad.

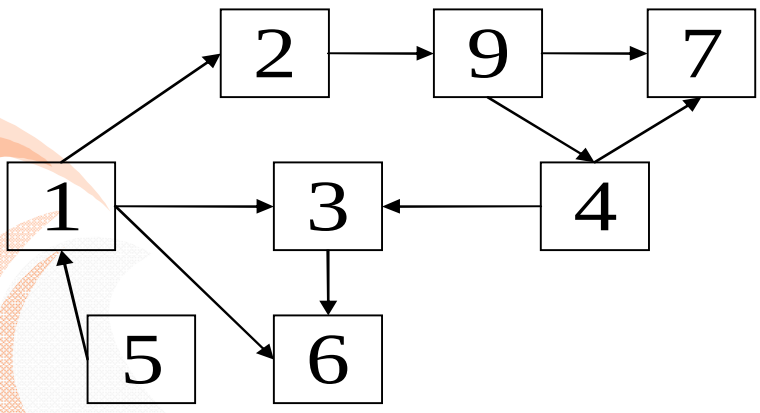
B) Etiqueta los arcos

C) ¿Este grafo tiene ciclos? ¿Por qué?

D) ¿Es un grafo fuertemente conexo? Justifica tu respuesta.

4. Grafos dirigidos

EJERCICIO 1. RECORRIDO EN PROFUNDIDAD O DFS



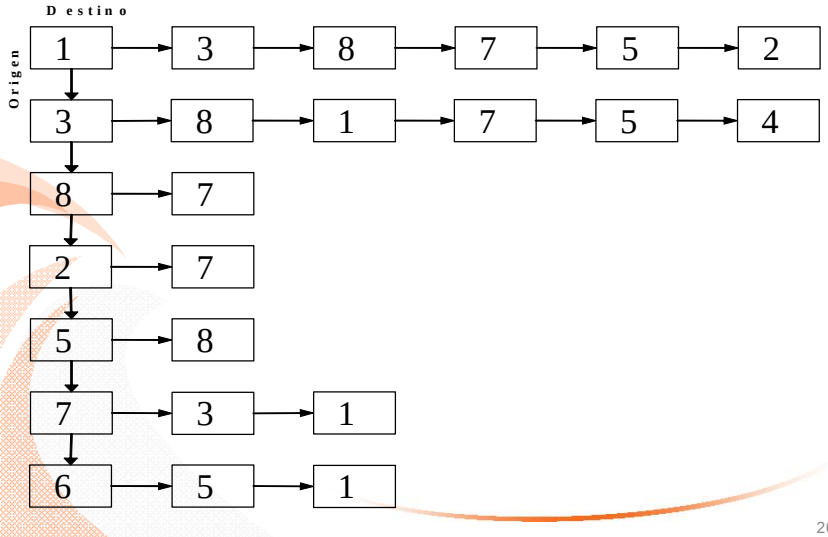
4. Grafos dirigidos

EJERCICIO 2. RECORRIDO EN PROFUNDIDAD O DFS

- Dado el grafo dirigido representado por la lista de listas que se muestra a continuación, calcular el árbol extendido en profundidad que parte del vértice 1. Para recorrer las listas de adyacencia, seguir el orden izquierda a derecha de cada lista.

4. Grafos dirigidos

EJERCICIO 2. RECORRIDO EN PROFUNDIDAD O DFS



4. Grafos dirigidos

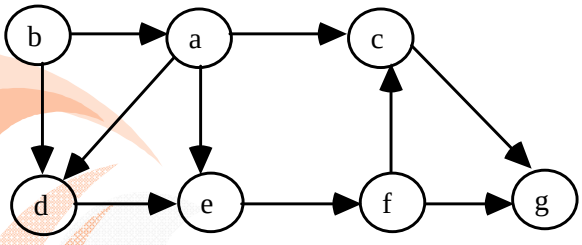
RECORRIDO EN ANCHURA O BFS (I)

- Primero visita todos los adyacentes a *vi* y después los adyacentes a los visitados

```
ALGORITMO BFS
ENTRADA   v: TVértice; G: TGrafo;
VAR       w1, w2: TVértice;   visitados: TConjunto( TVértice );Q: TCola( TVértice );
METODO
    INSERTAR( visitados, v );
    ENCOLAR( Q, v );
    VISITAR( v );
    mientras no EsVacia( Q ) hacer
        w1  = CABEZA( Q ); DESENCOLAR( Q );
        para todo w2  ∈ L( w1 ) hacer          // Lista adyacencia
            si no PERTENECE( w2 , visitados ) entonces
                VISITAR( w2 );
                ENCOLAR( Q, w2 );
                INSERTAR( w2, visitados );
            fsi
        fpara
    fmientras
fMETODO
```


4. Grafos dirigidos

RECORRIDO EN ANCHURA O BFS (II)



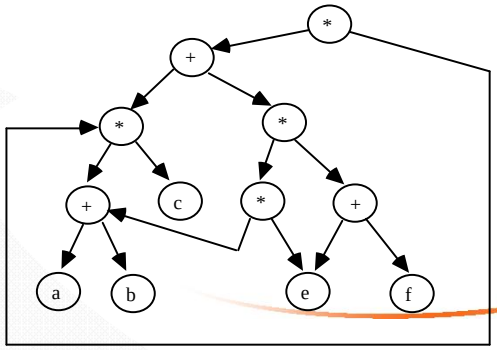
Recorridos BFS:

a,c,d,e,g,f
b,a,d,c,e,g,f

4. Grafos dirigidos

GRAFOS ACÍCLICOS DIRIGIDOS O GAD (I)

- Digrafo que no tiene ciclos.
- Son más generales que los árboles, pero menos que los grafos dirigidos arbitrarios.
- Son útiles para representar la estructura sintáctica de expresiones aritméticas con subexpresiones comunes.
- Ejemplo: ((a+b) * c + ((a+b) * e) * (e+f))* ((a+b) * c)



4. Grafos dirigidos

GRAFOS ACÍCLICOS DIRIGIDOS O GAD (II)

- Otra aplicación de los GAD consiste en la representación de órdenes parciales:
- Un **orden parcial *R* en un conjunto *S*** es una relación binaria tal que:

  $\forall a \in S$, $a\ R\ a$ es FALSO (*R* es irreflesiva)

  $\forall$ a, b, c $\in$ S, **si** $a\ R\ b$ y $b\ R\ c$ **entonces** $a\ R\ c$ (*R* es transitiva)
- **Ejemplo**. Sea el conjunto *S={1, 2, 3}*, y sea el conjunto de todos los subconjuntos de *S*:

  P(S) = {Ø, {1}, {2}, {3}, {1, 2}, {1, 3}, {2, 3}, {1, 2, 3}}

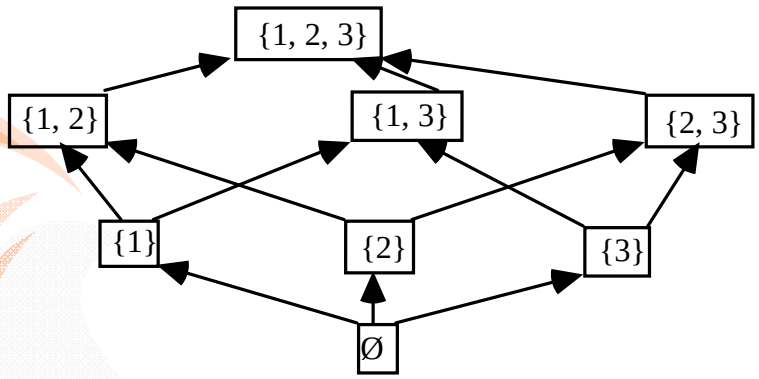
  La relación $\subset$ es un orden parcial en P(S), ya que cumple:

  $\forall$ A $\in$ P(S), A $\subset$ A es FALSO (ireflexividad)

  Si A $\subset$ B y B $\subset$ C $\rightarrow$ A $\subset$ C (transitividad)

4. Grafos dirigidos

GRAFOS ACÍCLICOS DIRIGIDOS O GAD (III)



5. Grafos no dirigidos

ALGORITMOS DE RECORRIDO

- **EN PROFUNDIDAD (DFS):**
- Se puede usar el mismo algoritmo que en grafos dirigidos

  TIPOS DE ARCOS:
  - Del Arbol
  - De Retroceso (= avance. No existen de cruce)

- **EN AMPLITUD (BFS):**
- Igual que para digrafos
- Se puede construir un bosque de extensión:

  (x,y) será del árbol si "y" (no se había visitado anteriormente) se visitó primero partiendo de "x" del ciclo interno de BFS.

  Toda arista que no es del árbol es una arista de retroceso (conecta dos vértices ninguno de los cuales es antecesor del otro).