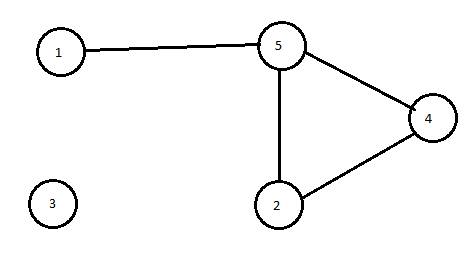
**Grafuri neorientate** ***(a good tool for drawing graphs: https://graphonline.ru/)***

Un graf este structură de date formată dintr-o mulțime de noduri (notată de regulă cu X, iar n= nr. de noduri = - nodurile se numerotează de regulă cu numere de la 1 .. n) și o mulțime de muchii. O muchie reprezintă o legătură între două noduri, din acest motiv mulțimea muchiilor (notată de regulă cu U, iar m=nr. de muchii = ) este formată din elemente de forma [u,v] cu u,v ∈ U. Dacă [u,v] este muchie, atunci se consideră implicit că și [v,u] există și NU se mai repetă scrierea sa.

Ex: pentru graful de mai jos avem



X = {1, 2, 3, 4, 5} cu n=5

U = { [1,5], [2,5], [2,4], [4,5] }

**Noțiuni din grafuri**:

**noduri adiacente** = legate printr-o muchie.

Ex: nodurile 1 și 5 sunt adiacente. Nodurile 1,4 NU sunt adiacente.

**muchie incidentă într-un nod** = muchia are nodul respectiv pe post de unul dintre capete

Ex: muchia [2,5] este incidentă în nodurile 2 și 5.

**gradul unui nod d(x)** = degree of x = numărul de muchii incidente în acel nod.

Dacă gradul unui nod este 0 - se numește nod izolat (EMO :D)

Dacă gradul este 1 - se numește nod terminal.

Ex: nodul 5 are gradul 3

nodul 1 are gradul 1 (→ este terminal)

nodul 3 are gradul 0 (→ este izolat)

**Teoremă**: Suma gradelor tuturor nodurilor unui graf neorientat este egală cu dublul numărului de muchii:

Dacă avem muchiile unui graf, putem scrie direct care este șirul gradelor, fără a mai desena graful :

Ex: fie graful de mai jos, cu n=11 noduri și muchiile următoare:

[1,4],[1,5],[3,5], [3,4], [3,6],[4,5],[4,6],[7,8],[7,9],[8,9],[9,10]

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| nodul x | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
| gradul lui x - d(x) | \*\* |  | \*\*\* | \*\*\*\* | \*\*\* | \*\* | \*\* | \*\* | \*\*\* | \* |  |

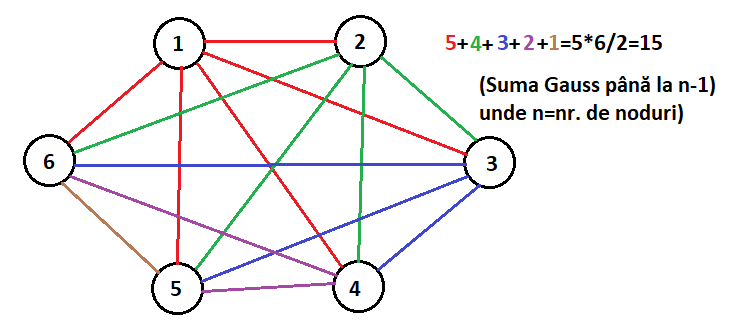
Se observă că avem m=11 muchii iar suma gradelor tuturor nodurilor este 22.

→ nodul 2 și nodul 11 sunt EMO (izolate) și nodul 10 este terminal.

Acesta din stânga este graful dat în exemplu de mai sus.

**Graf complet** : un graf în care oricare două noduri sunt adiacente.

**Teoremă**: un graf complet cu n noduri are muchii.



**Graf parțial** : se obține dintr-un graf păstrând TOATE nodurile și DOAR o submulțime a muchiilor.

**Teoremă** : O mulțime cu **n** elemente are **2n** submulțimi (incluzând-o și p-aia vidă).

(teoremă de la mate)

**Corolar din ultimele două :** Un graf cu **m** muchii are **2m** grafuri parțiale.

**Subgraf** : se obține dintr-un graf păstrând o submulțime de noduri și DOAR acele muchii care leagă nodurile păstrate și existau și înainte în graf.

**Corolar** : Un graf cu **n** noduri admite **2n**­ subgrafuri. (din mulțimea celor n noduri păstrăm o submulțime)

**Lanț** : Un șir de noduri legate prin muchii (vecinii în lanț)

Ex: pe graful din poza de la primul exemplu, lanț: (4, 5, 3, 4, 6, 3)

**Lungimea unui lanț** : numărul de muchii ale sale ( = nr. de noduri minus 1)

Lanț **elementar** : nu se repetă niciun nod.

Lanț **simplu** : nu se repetă nicio muchie (au voie să se repete noduri).

Lanț **neelementar** : se repetă vreun nod.

**Graf conex**: Între oricare două noduri există un lanț.

Evident, graful nostru din poză NU este conex.

**Componentă conexă** este un subgraf al unui graf dat, care este conex și este maximal cu această proprietate (orice alt nod am fi păstrat în subgraf, el NU mai rămânea conex)

Pe graful nostru avem 4 componente conexe: {1, 3, 4, 5, 6} {2} {7, 8, 9, 10} {11}

**Ciclu** : Un lanț în care (!!) **NU SE REPETĂ MUCHII** (!!) și în care nodul inițial coincide cu cel final.

Dacă se repetă noduri → ciclul este neelementar. În caz contrar este elementar (într-un ciclu elementar, în scrierea sa are voie să se repete primul și ultimul nod).

Exemplu de ciclu elementar în graful nostru (primul exemplu) :

1, 5, 3, 6, 4, 1 - de lungime 5

Exemplu de ciclu neelementar în același graf: 1, 4, 3, 6, 4, 5, 1 - lungime 6

**Ciclu hamiltonian**  - un ciclu **elementar** care trece prin toate nodurile grafului. În cazul în care există, și graful s.n. hamiltonian

Există o teoremă perfect inutilă de caracterizare a unui graf hamiltonian: DACĂ într-un graf neorientat cu n noduri gradul fiecărui nod este mai mare sau egal cu [n/2] ATUNCI graful este hamiltonian.

Teorema asta zice de fapt că avem foarte multe muchii, atât de multe muchii încât sigur putem să formăm un ciclu hamiltonian.

Inutilitatea acestei teoreme vine din faptul că ea NU caracterizează graful prin "dacă și numai dacă" - deci este cu unic sens (deci reciproca NU e adevărată)

**Ciclu eulerian** - un ciclu care trece prin toate muchiile grafului (nu neapărat elementar). În cazul în care există, și graful s.n. eulerian.

Există o teoremă puternică de caracterizare a unui graf eulerian:

**Un graf este eulerian dacă și numai dacă este conex (abstracție făcând de nodurile izolate) și gradul fiecărui nod al său este par.**

**Arbore** d.p.d.v. al teoriei grafurilor (se mai numesc și "arbori fără rădăcină") = un graf conex fără cicluri.

**Teoremă**: un arbore cu **n** noduri are exact **n-1** muchii.

Orice muchie i-am scoate, NU mai e conex

Orice muchie i-am adăuga, se formează un ciclu.

**Metode de reprezentare ale grafurilor în memorie**

1) Cu ajutorul unei **matrice de adiacență**: o matrice pătratică, de dimensiuni n x n, simetrică (pentru că grafurile sunt neorientate) în care

|0 dacă i=j sau dacă NU există muchie între i și j

a[i][j]=|

|1 dacă există muchie de la i la j.

Ex: Graful din primul exemplu are următoarea matrice de adiacență:

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
| 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 3 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| 4 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| 5 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 6 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 7 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 |
| 8 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 |
| 9 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 |
| 10 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| 11 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**2) Liste de adiacență:**

Pentru fiecare nod se reține o listă în care sunt trecuți toți vecinii săi (nodurile cu care acesta este adiacent). Dacă [x,y] este o muchie atunci atât y apare în lista de vecini ai lui x, cât și x apare în lista de vecini ai lui y.

Pe graful dat, aceste liste sunt:

Listele:

|  |  |
| --- | --- |
| Nodul | Vecinii |
| 1  2  3  4  5  6  7  8  9  10  11 | 4, 5  - (sau NULL)  4, 5, 6  1, 3, 5, 6  1, 3, 4  3, 4  8, 9  7, 9  7, 8, 10  9  - (sau NULL) |

**3) Vector de muchii:** se face fie un vector de struct, fie o matrice cu două coloane, în care trecem pur și simplu capetele muchiilor. Pentru o muchie [u,v] se trece doar o singură dată

Pe graful din exemplu, varianta cu matrice cu două coloane:

|  |  |  |
| --- | --- | --- |
| indice muchie | coloana 1=  unul dintre  capete | coloana 2=  celălalt dintre  capete |
| 1 | 1 | 4 |
| 2 | 1 | 5 |
| 3 | 3 | 4 |
| 4 | 3 | 5 |
| 5 | 3 | 6 |
| 6 | 4 | 5 |
| 7 | 4 | 6 |
| 8 | 7 | 8 |
| 9 | 7 | 9 |
| 10 | 8 | 9 |
| 11 | 9 | 10 |