

CUPRINS

1. Algoritmi elementari
2. Vectori
3. Matrici
4. Firiere
5. Subprograme
6. Stivă . Coadă
7. Recursivitate
8. Siruri de caractere
9. Grafuli neorientate
10. Grafuli orientate
11. Arbozi
12. Backtracking
13. Metoda Greedy
14. Programare dinamică

ALGORITMI ELEMENTARI

1. Cifre, oglindit, palindrom, construire număr
2. Cîndră, cîmpe: rădări, Euclid
3. Număr prim
4. Divizori: sume, produse
5. Descompunere în factori primi
6. Înd. în Fibonaci

① OGLINDIT

int oglindit (int n)

{

 int c, ogl = 0;

 while (n > 10)

 {

 c = n % 10;

 ogl = ogl * 10 + c;

 return ogl;

}

PALINDROM

if (n == oglindit (n))

 cout << "palindrom";

CONSTRUCȚIE NUMAR

int construit (int n)

{

 int c, nr, nr;

 nr = 0;

 n = 1;

 while (n != 0)

 {

 c = n % 10; n / = 10;

 nr = nr + c * n;

 n * = 10;

 }

 return nr;

}

②. CMMDC

a) Sčádavý repeat

int umnoz (int a, int b)

}

while (a != b)

{

if (a > b) a = a - b;

else b = b - a;

}

}

b) Euklid

int umnoz (int a, int b)

{

int r = a % b;

while (r != 0)

{

a = b;

b = r;

r = a % b;

}

} return b;

CMMMC

a) $\text{ummc}(a, b) = \frac{a * b}{\text{umnoz}(a, b)}$

b) A dvojicí repeat

int ummc (int a, int b)

{

int x = a, y = b;

while (x != y)

{

if (x < y) x = x + a;

else y = y + b;

}

} return x;

}

③ NUMÄR PRIM

bool prim(int n)

{

 if (n <= 1) return false;

 if (n == 2) return true;

 if (n > 2 & n % 2 == 0) return false;

 for(int i = 3; i * i <= n; i++)

{

 if (n % i == 0) return false;

 return true;

}

④ DIVIZORI

void div(int n)

{

 for(int i = 1; i * i <= n; i++)

{

 if (n % i == 0)

{

 premaga(k++, s = s + i);

 int j = n / i;

 if (i != j) premaga(...);

y

y

y

5 DESCOMPOUNEREA IN FACTORI PRIMI

```

f = 2;
while (n != 1)
{
    n = 0;
    while (n * f == 0)
    {
        n / = f;
        n + +;
    }
    if (n > 0) * and << f << " " << n << endl;
    f + +;
}

```

NUMĂRUL DE DIVIZORI

$$\text{nr. div} = (n_1 + 1)(n_2 + 1) \dots (n_k + 1)$$

$$* K = K * (n + 1)$$

SUMA DIVIZORILOR

$$\text{suma div} = \frac{t_1^{n_1+1} - 1}{t_1 - 1} + \frac{t_2^{n_2+1} - 1}{t_2 - 1} + \dots + \frac{t_k^{n_k+1} - 1}{t_k - 1}$$

$$* S = S * (pow(t, n+1) - 1) / (t - 1)$$

⑤ FIBONACCI

$$f(n) = \begin{cases} 1, & n \leq 2 \\ f(n-2) + f(n-1), & n \geq 3 \end{cases}$$

Primi n termini ai simili li fibonacci

$a = 1;$ $b = 1;$

if ($n == 1$) cout « a;

else if ($n == 2$) cout « a « " " « b;

else

{

cout « a « " " « b « " ";

for (int i=3; i <= n; i++)

{

$c = a + b;$

cout « c « " ";

$a = b;$

$b = c;$

}

}

VECTORI

1. Minime / Maxime simple și multiple
2. Min / Max și numărare
3. Ordinări : bubble, interschimbare directă, selecție, găsire, raportă, prin interclasare
4. Căutare sequentială, căutare binară
5. Operări cu multimi : verificare multime, intersecție, reunire, diferență
6. Interclasare simplă, cu selecție
7. Vectori de frecvență
8. Operări cu numere mari
9. Încercări / Eliminări din vector

① MIN / MAX SIMPLE

include <limits.h>

maxi = INT_MIN;

mini = INT_MAX;

for(int i=1; i<=n; i++)

{

if(v[i]>maxi) maxi = v[i];

if(v[i]<mini) mini = v[i];

}

MIN / MAX MULTIPLE

maxi1 = 0; maxi2 = 0;

for(int i=1; i<=n; i++)

{

if(v[i]>maxi1)

{

maxi2 = maxi1;

}

maxi1 = v[i];

else if(v[i]>maxi2)

maxi2 = v[i];

}

② MIN / MAX SI NUMARARE

maxi = 0; K = 1;

for(int i=1; i<=n; i++)

{

if(v[i]>maxi)

{

maxi = v[i];

K = 1;

else if(v[i] == maxi)

K = K + 1;

}

3. ORDONĂRI

INTERSCHIMBARE DIRECTĂ

```
for (int i = 1; i < n; i++)
```

```
{
```

```
    for (int j = i + 1; j <= n; j++)
```

```
{
```

```
        if (v[i] > v[j])
```

```
            aux = v[i];
```

```
v[i] = v[j];
```

```
} v[j] = aux;
```

```
}
```

```
}
```

SORTARE PRIN SELECTIE

```
for (int i = 1; i < n; i++)
```

```
{
```

```
    min = v[i];
```

```
    p = i;
```

```
    for (int j = i + 1; j <= n; j++)
```

```
{
```

```
        if (v[j] < min) { min = v[j];  
                           p = j; }
```

```
    aux = v[i];
```

```
v[i] = v[p];
```

```
v[p] = aux;
```

```
}
```

$$O(n \cdot (n-1)/2) \approx O(n^2)$$

BUBBLE SORT

$O(n^2)$

do

{

$ok = 1;$

for (int $i = 1; i < n; i++$)

{ if ($v[i] > v[i+1]$)

{

$aux = v[i];$

$v[i] = v[i+1];$

$v[i+1]$

$ok = 0;$

}

}

} while ($ok == 0$);

SORTARE PRIN INTERCLASARE (MERGE SORT)

void inter (int v[], int st, int dr)

$O(n \log_2 n)$

{

int w[10], i, j, k, m

$m = (st + dr) / 2;$

$i = st; j = m + 1;$

$k = st - 1;$ // contor de numărare $\Leftrightarrow k=0;$

while ($i \leq m \wedge j \leq dr$)

{

if ($v[i] < v[j]$)

$w[k++k] = v[i++];$

}

else $w[k++k] = v[j++];$

while ($i \leq m$) $w[i++k] = v[i++];$

while ($j \leq dr$) $w[j++k] = v[j++];$

for ($i = st; i \leq dr; i++$)

{

$v[i] = w[i];$

void MergeSort (int v[], int st[], int dr[])

{

if ($st < dr$)

{

int m = $(st + dr) / 2;$

MergeSort (v, st, m);)

MergeSort (v, m+1, dr);

} inter (v, st, dr);

}

$O(n \log_2 n)$

SORTARE RAPIDA (QUICK SORT)

```
void pivot ( int v[], int st, int dr )
{
    int n, q, x;
    n = st; q = dr;
    x = v[n];
    while (n < q)
    {
        while (n < q && v[q] >= x) q--;
        v[n] = v[q];
        while (n < q && v[n] <= x) n++;
        v[q] = v[n];
    }
    v[n] = x;
    return n;
}
```

```
void Quicksort ( int v[], int st, int dr )
{
    int m
    if (st < dr)
    {
        m = pivot ( v, st, dr );
        quicksort ( v, st, m - 1 );
        quicksort ( v, m + 1, dr );
    }
}
```

Exemplu Quicksort

n	n	n	n	n	n	q	q	q	q	q
20	4	19	12	17	20	100	61	92	100	58
1	2	3	4	5	6	7	8	9	10	

$$x = 20;$$

Quicksort

1. Se plasează un pivot pe poziția i (p) și un punct pe mijloc x
2. Se salvează în variabila x prima valoare a intervalului
3. Algoritmul interschimbă valorile din prima jumătate mai mari decât valoarea x cu cele mai mici din a doua jumătate
=> valoarea x ajunge pe poziția sa în următorul sortat
iar toate valorile dinaintea sa sunt mai mari decât x

4 CĂUTAREA UNUI ELEMENT

CĂUTARE SECVENTIALĂ

$O(n)$

`ok = false;`

```
for ( int i=1; i <= n && !ok; i++ )
```

{

```
    if ( v[i] == x ) ok = true;
```

}

CĂUTARE BINARĂ

// vectorul trebuie să fie ordonat

!

`rt = 1; dr = n;`

`ok = false;`

```
while ( rt <= dr )
```

{

`m = (rt + dr) / 2;`

`if (x == v[m]) ok = true;`

`else if (x < v[m]) dr = m - 1;`

`else rt = m + 1;`

}

$O(\log_2 n)$

5. OPERAȚII PE MULTIMI

INTERSECTIONA

A ∩ B

```
for( int i = 1; i <= n; i++ )  
{  
    ok = false;  
    for( int j = 1; j <= m && !ok; j++ )  
    {  
        if( a[i] == b[j] ) ok = true;  
    }  
    if( ok == true ) c[ ++kj ] = a[i];  
}  
y
```

DIFERENȚA

A \ B

```
for( int i = 1; i <= n; i++ )  
{  
    ok = false;  
    for( int j = 1; j <= m && !ok; j++ )  
    {  
        if( a[i] == b[j] ) ok = true  
    }  
    if( ok == false ) c[ ++kj ] = a[i];  
}  
y
```

REUNIUNEA

```
for( int i = 1; i <= m; i++ )
```

```
    c[ ++k ] = a[ i ];
```

```
for( int j = 1; j <= m; j++ )
```

```
{
```

ok = false;

```
for( int i = 1; i <= m && !ok; i++ )
```

```
{
```

if(a[i] == b[j]) ok = true;

```
}
```

if(ok == false) c[++k] = b[j];

```
}
```

VERIFICARE MULTIME

ok = 1;

```
for( int i = 1; i < m && ok; i++ )
```

```
{
```

```
    for( j = i + 1; j <= m && ok; j++ )
```

```
{
```

if(vr[i] == vr[j]) ok = 0; // int-o multime el. sunt
// distinse

```
}
```

```
}
```

6. INTERCLASARI

! vectorii trebuie să fie ordonate (vectorii nu devin)

INTERCLASARE SIMPLĂ

$i = 1; j = 1; k = 0;$

while ($i \leq n \wedge j \leq m$)

}

if ($a[i] < b[j]$) $c[+ + k] = a[i + +];$
else $c[+ + k] = b[j + +];$

}

while ($i \leq n$) $c[+ + k] = a[i + +];$

while ($j \leq m$) $c[+ + k] = b[j + +];$

INTERCLASARE CU SELECTIE

$K_a = K_b = 0;$

for ($i = 1; i \leq n; i + +$)

{ am $\rightarrow x;$

y if (conditie) $a[+ + K_a] = x;$

for ($i = 1; i \leq m; i + +$)

{ am $\rightarrow y;$

y if (conditie) $b[+ + K_b] = y;$

$n = K_a; m = K_b;$

ANALOG

7. VECTORI DE FRECUENȚĂ

```
int fr[100];
for( int i=0; i <= 99; i++)
    fr[i] = 0;
for( int i = 1; i <= n; i++)
{
    cin >> x;
    fr[x]++;
}
for( int i = 0; i <= 99; i++)
{
    if( fr[i] > 0) ...
}
```

VECTORI DE FRECUENȚĂ CU STĂRI

Exemplu

a:

10	7	8	12	5	13
----	---	---	----	---	----

b:

8	10	3	2
---	----	---	---

c:

9	8	11	7	2
---	---	----	---	---



Se dau 3 vectori:

+

Afișați el comunul din ce 3 vectori

```
for( int i = 0; i <= 99; i++ )  
    f[i] = 0;
```

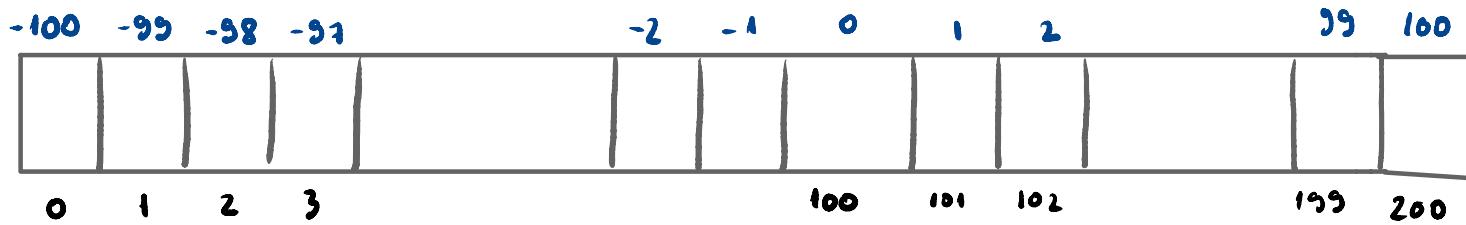
```
for( int i = 1; i <= m; i++ )  
{  
    cin >> x;  
    f[x] = 1;
```

```
for( int i = 1; i <= m; i++ )  
{  
    cin >> y;  
    if( f[y] == 1 ) f[y] = 2;  
}
```

```
for( int i = 1; i <= n; i++ )  
{  
    cin >> z;  
    if( f[z] == 2 ) f[z] = 3;  
}
```

```
for( int i = 0; i <= 99; i++ )  
{  
    if( f[i] == 3 ) cout << i << " ";  
}
```

FRECUENȚĂ PE NUMERE ÎNTREGI



$$x \in [-100, 100]$$

```
int f[201];
```

```

for( int i=1; i <= m; i++)
{
    cin >> x;
    f[x+100]++;
}

```

```

for( int i=0; i <= 200; i++)
{
    if( f[i]>0) cout << i-100;
}

```

FRECUENȚĂ PE LITERE

I Ineficient

```

for( int i=1; i <= m; i++)
{
    char c;
    cin >> c;
    x = (int) c;
    f[x]++;
}

```

```

int a, z;
a = (int)'a';
z = (int)'z';

```

```

for( int j=a; j <= z; j++)
{
    if( f[j]>0) ...
}

```

II Eficient

```

for( int i=1; i <= m; i++)
{
    char c;
    cin >> c;
    x = c - 'a';
    f[x]++;
}

```

```

for( int i=0; i <= 26; i++)
{
    if( f[i]>0)
        cout << 'a' + i;
}

```

MATRICI

1. Marginile matricei. Cifre și Bordare. Vîrni
2. Matrice pătratice : elemente situate pe diagonala principală, secundară, deasupra, sub cele două diagonale, în poziții de nord, sud, est, vest
3. Pătrate concentrice. Spirale
4. Operații pe linii, coloane : sume, produse, numărări, elemente maxime, minime, ordonări
5. Funcții pentru vectori în calculul operațiilor pe linii
6. Rotiri de matrice

① CHEMICAL MATRIX

$i = 1 \dots n$ $j = 1 \dots m$

	1	2	3	4	5	6	7
1	10	20	4	7	61	5	21
2	7	5	200	5	12	43	
3	65	50	2	0	40	56	91
4	2	3	7	71	71	2	20
5	19	8	33	100	60	52	7

BORDARE MATRIX

for ($i = 0; i <= n+1; i++$)

}

 for ($j = 0; j <= m+1; j++$)

 if ($i == 0 \text{ or } i == n+1 \text{ or } j == 0 \text{ or } j == m+1$)

$a[i][j] = 0;$

}

0 1 2 3 4 5 6

0	0	0	0	0	0	0	0
1	0	7	5	200	5	12	0
2	0	50	2	0	40	56	0
3	0	3	7	71	71	2	0
4	0	0	0	0	0	0	0

② MATRICE PĂTRATICE

DIAGONALA PRINCIPALĂ

$$i = j \quad (i, j = \overline{1, n})$$

a) SUB DIAG. PRINCIPALĂ

$$i > j$$

b) DEASUPRA DIAG. PRINCIPALE

$$i < j$$

	1	2	3	4	5	6
1	██████					██████
2		██████				██████
3			██████		██████	
4				██████	██████	
5					██████	
6	██████					██████

DIAGONALA SECUNDARĂ

$$i + j = n + 1 \quad (i, j = \overline{1, n})$$

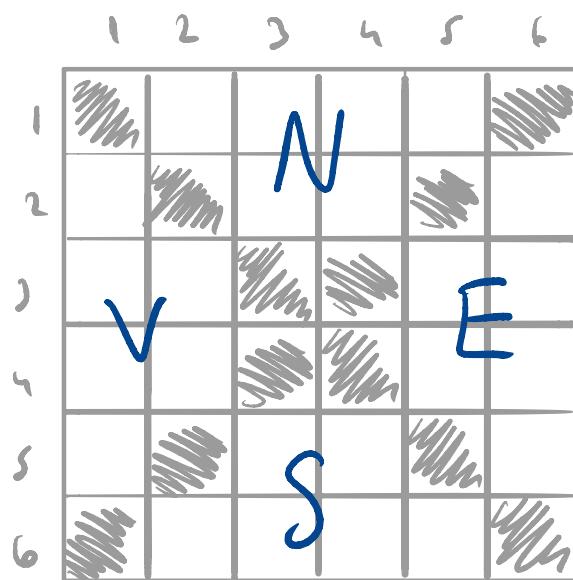
a) SUB DIAG. SEC

$$i + j > n + 1$$

b) DEASUPRA DIAG. SEC.

$$i + j < n + 1$$

ZONE IN MATRICEA PATRATICA



Nord :
$$\left\{ \begin{array}{l} i < j \\ i + j < n + 1 \end{array} \right.$$

Sud :
$$\left\{ \begin{array}{l} i > j \\ i + j > n + 1 \end{array} \right.$$

Est :
$$\left\{ \begin{array}{l} i < j \\ i + j > n + 1 \end{array} \right.$$

Vest :
$$\left\{ \begin{array}{l} i > j \\ i + j < n + 1 \end{array} \right.$$

③ PĂTRATE CONCENTRICE

Numește de patrate concentrice $\frac{n+1}{2} = n/2 + n \cdot 1/2$

	1	2	3	4	5	6
1						
2						
3						
4						
5						
6						

$a[i][j]$ \dots \dots $a[n-i][n+1-i]$

•

•

•

•

•

•

•

•

•

•

•

•

•

•

$a[n+1-i][i]$ \dots \dots $a[n+1-i][n+1-i]$

$$i = \overline{1, (n+1)/2}$$

FINISIERE

1. Citire când se cunoaște / nu se cunoaște numărul de elemente
2. Determinarea celor mai mari / mică două / trei numere
3. Determinarea celor mai măre / mic număr + nr. operații
4. Det. lungimii maxime a sevenței de elemente alăturate cu o anumită proprietate
5. Det sevenței de lungime maximă cu o anumită proprietate
6. Afisarea în ordine cresc./descresc. a numerelor în ceea ce care anotă îndeplinește o anumită condiție (sunt formate din una / două cifre - și anotă)

STIVĀ
COADA

GRAFURI NEORIENTATE

1. Grad, vîrf izolat, vîrf terminal;

Suma gradelor = 2 m

2. Funcții pentru determinarea gradului

3. Matrice de adiacență, liste de adiacență

4. Graf parțial, subgraf

5. Graf complet - numărul de muchii

6. Lant, cale + elementar

7. Graf conex, componente conexe. Determinarea numărului maxim de componente conexe

8. Cale eulerian, graf eulerian. Teorema. Algoritmi pentru determinarea unei cale eulerian

9. Cale hamiltonian, graf hamiltonian. Teorema. Determinarea unei componente hamiltonian (program C++)

10. Graf turcesc. Determinare drum în graf turcesc

11. Parcurgeri

- Parcurgerea în lățime

- Parcurgerea în adâncime

- Corectate: verificare, det. componente conexe, componente conexe maxime

- Verificare graf bipartit

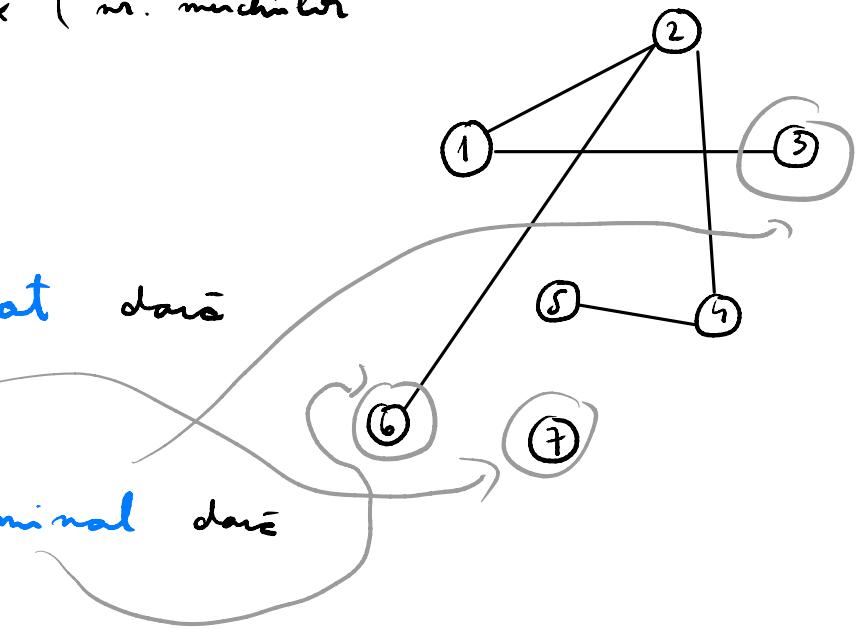
- Distanțe minime folosind BFS (adaptare Lee pe grafuri)

① GRAD, VÂRZ / ZOLAT / ZOLAT

Numele gradul varfului, $d(x)$ numărul
varfurilor adiacente cu x (nr. muchiilor
incidente)

Varful x nu e n. izolat deci
gradul său este zero

Varful x nu e n. terminal deci
gradul său este 1



② FUNCTIE PT. DET. GRADULUI

```
int grad ( int x )  
{  
    int s = 0;  
    for( int i = 0; i <= n; i++ )  
    {  
        s += adj[x][i];  
    }  
    return s;  
}
```

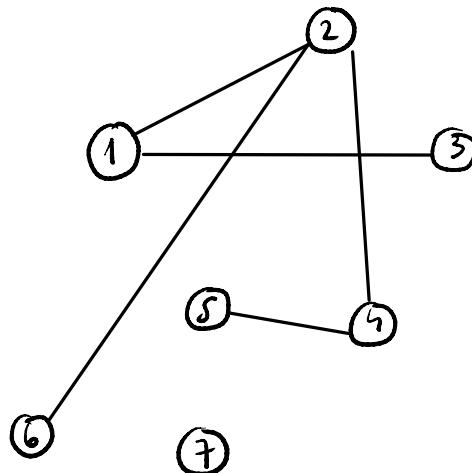
Torema Suma gradelor = $2 * m$ (nr. muchii)

③ MATRICE DE ADIACENȚĂ, LISTE DE ADIACENȚĂ

MATRICE DE ADIACENȚĂ

$$a_{i,j} = \begin{cases} 1, & \text{dă muchie de la } i \text{ la } j \\ 0, & \text{în } \{i, j\} \notin U \end{cases}$$

a	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0
2	1	0	0	1	0	1	0
3	1	0	0	0	0	0	0
4	0	1	0	0	1	0	0
5	0	0	0	1	0	0	0
6	0	1	0	0	0	0	0
7	0	0	0	0	0	0	0



Obs

1. $a_{i,j} \neq a_{j,i} \Rightarrow a_{i,j} = 0$
2. matricea **simetrică** fără de diag principală
3. complexitate matrică $O(n^2)$
4. suma el. de pe linia i = suma el. de pe coloana i = $d(i)$

VECTORI DE MUCHII

struct muchie

```

    {
        int x, y
    }
    vector<muchie> v;

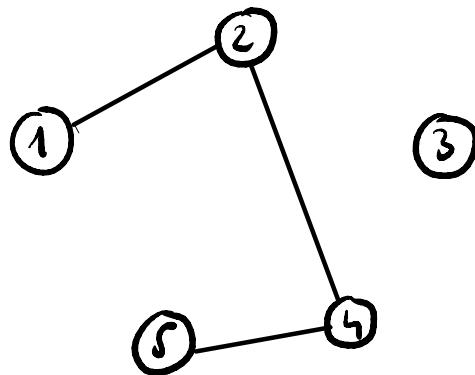
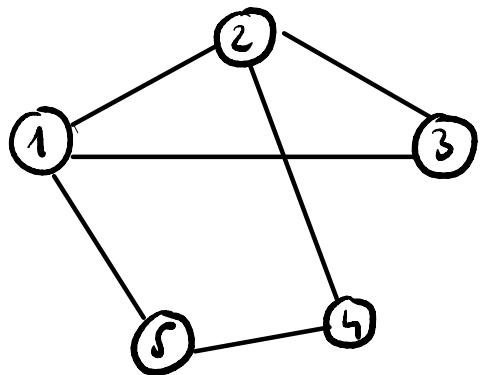
```

v:

1	2	1	3	2	4	2	6	4	5
---	---	---	---	---	---	---	---	---	---

4 GRAF PARTIAL, SUBGRAF

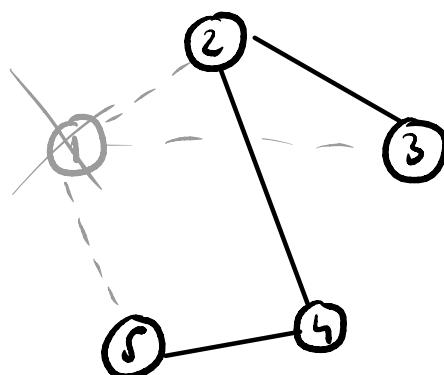
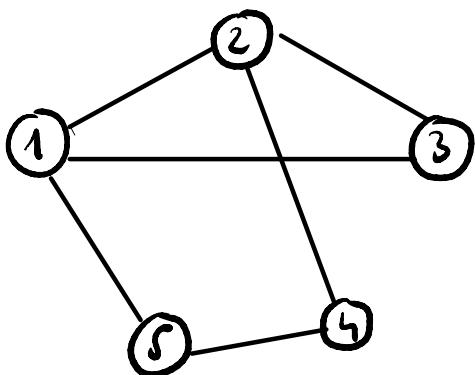
Un graf parțial se obține prin răstrângere varfurilor grafului initial și eliminarea unei muchii.



Teorema Fie G un graf cu n varfuri și m muchii.

$$\begin{aligned} \text{Nr. grafuri parțiale} &= C_m^0 + C_m^1 + \dots + C_m^{m-1} \\ &= 2^m \end{aligned}$$

Un subgraf se obține prin eliminarea unei varfuri și a tuturor muchiilor adiacente acestora.



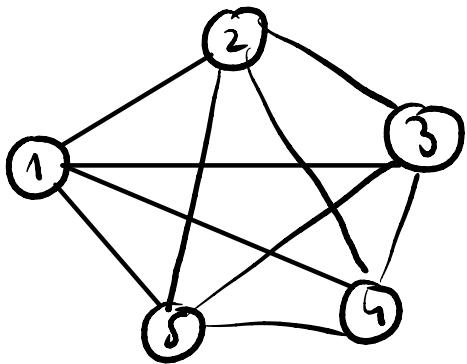
Teorema Fie G un graf cu n varfuri și m muchii.

$$\begin{aligned} \text{Nr. subgrafuri} &= C_m^0 + C_m^1 + \dots + C_m^{m-1} \\ &= 2^m - 1 \end{aligned}$$

5 GRAF COMPLET

Un graf G re n. graf complet dñs dicere
dñs v. sunt adiacente

Nr. maxim de muchii intr-un G_f complet = $\frac{n(n-1)}{2}$



Un graf complet cu $n > 2$

contine $\frac{(n-1)!}{2}$ cicluri hamiltoniene

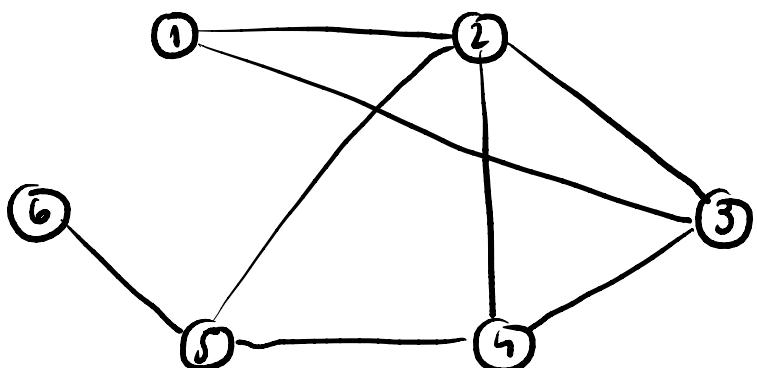
⑥ LANT, CICLU + ELEMENTAR

Se n. lant în graful G o succesiune de varfuri an nezidătoare cînd oricare 2 varfuri vecini sunt adiacente

Se n. lant elementar un lant în care v.f. sunt distincte doar căte două

Se n. ciclu un lant an proprietatea cînd primul varf este egal cu ultimul.

Se n. ciclu elementar un ciclu în care varfurile, an excepția primei și ultimului, sunt distincte doar căte două.



$L = [1, \underline{2}, 4, 3, \underline{2}, 5]$ lant nesim.

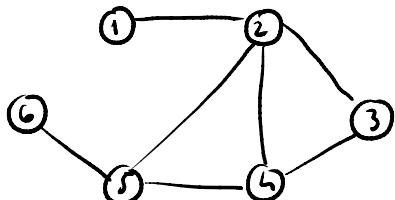
$L = [1, 3, 2, 4, 5]$

$C = [1, 2, 3, 1] \rightarrow$ ciclu elementar

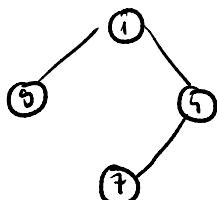
$C = [3, 4, 5, 2, 1, 3]$

⑦ GRAF CONEX

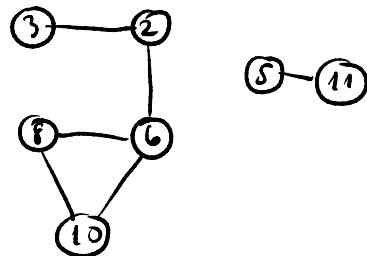
Un graf G este **conex** dacă între oricare două varfuri ale sale există un lanț.



G_f conex



G_f nu este conex
⇒ 3 componente conexe

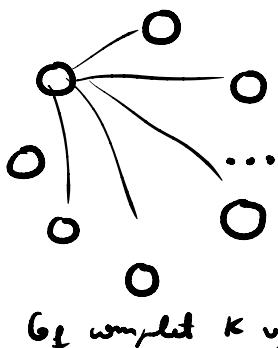


Dacă un graf G are **p** componente conexe atunci **nr. minim de muchii** cu trebui adăugate pentru ca G_f să fie **conex** este **$p-1$**

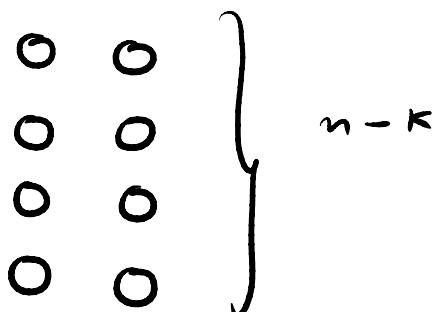
APLICATIE:

Fie G un graf cu n varfuri
și m muchii

Nr. **MAXIM** de componente conexe



G_f are cel mult K cf



$$\frac{k(k-1)}{2} \geq m, k \text{ min}$$

⇒ **$n - K + 1$** componente conexe

BEST OF *

1. Ridicare la putere logaritmice
2. Ultima cifră a lui x^y
3. Cifra de control
4. Triunghiul lui Pascal
5. Turnurile lui Hanoi
6. Codul lui Caesar
7. Numere primăvara
8. Polinoame
9. Ordinea operărilor
10. Expresie condițională
11. Bits
12. Tipul long
13. Operări pe bits

14. Semideagonale
15. Determinantul unei matrice
16. Numere divizibile dintr-un interval
17. Lățimea de divizibilitate cu $b-1$
18. Puterea la care apare un nr prim x în numera lui $n!$
19. Ultima cifră numără de la finalul lui $n!$
20. Permutări cu repetitii / puncte fixe
21. A K -a subordonare în ordine lexicografică
22. Grafuri parțiale și subgrafuri
23. Numărul căderilor hamiltoniene într-un graf complet
24. Numărul maxim de muchii într-un graf cu p componente conexe
25. Baza 16 (hexazecimală)
26. Algoritmul lui Bresenham
27. Al K -lea termen din sirul lui Fibonacci

28. Fonction injective
29. Principale de Dirichlet
30. Projet de Cantor
31. Stars and bars
32. Immunités à la Rousse

① RIDICARE LA PUTERE LOGARITMICA

$$x^y = \begin{cases} (x^2)^{\frac{y}{2}}, & y \text{ par} \\ (x^2)^{\frac{y-1}{2}} \cdot x, & y \text{ impar} \end{cases}$$

$O(\log_2 n)$

$n = 1;$

$$3^{21} = ?$$

while ($y > 0$)

{
 if ($y \cdot 1 \cdot 2 == 1$)

$$n = n * x;$$

$$x = x * x;$$

} $y = y / 2;$

x	y	n
3	21	3
3^2	10	
3^4	5	$3 \cdot 3^4 = 3^5$
3^8	2	
3^{16}	1	$3^5 \cdot 3^{16} = \underline{\underline{3^{21}}}$
3^{32}	0	

② ULTIMA CIFRĂ A LUI x^y

$y = y \cdot 1 \cdot 4 \rightarrow$ ultima cifră nu repetată

din 4 în 4

$$\text{uc}(x) = 1 \rightarrow \underline{1}$$

$$5 \rightarrow \underline{5}$$

$$6 \rightarrow \underline{6}$$

$$2 \rightarrow \underline{2, 4, 6, 8}, 2, 4, 6, 8, 2 \dots$$

$$4 \rightarrow \underline{4, 6, 4, 6}, 4, 6, 4, 6, 4 \dots$$

int m(int x, int y)

{

 int n = 1;

 x = x + 10;

 y = y + 4;

 if (y == 0) y = 4;

 for (int i = 1; i <= y; i++)

 n = (n * x) + 10;

 return n;

}

Exemplos

$$m(1472^{16}) = m(2^{16})$$

$$= m(2^4)$$

$$= 6$$

$$m(27^{2007}) = m(7^{2007})$$

$$= m(7^3)$$

$$= 3$$

③ CIFRA DE CONTROLE

I) Neficiente

while (n > 9)

{

 s = 0;

 while (n > 0)

{

 s += n % 10;

 n /= 10;

 m = s;

}

II) Puting mai eficiente

while (n > 9)

{

 n = n / 10 + n % 10;

}

III) Eficiente

if (n % 9 == 0) n = 9;

else n = n % 9;

$$Cifcontrol(n) = Cifcontrol(n + 9k)$$

$$\text{ex. } cifctrl(17) = cif(26) = cif(35) = \dots = cif(17 + 9k)$$

$$! \text{ consequência: } cif(x) = cif(y) \Rightarrow |y - x| \cdot 9 = 0$$

④ TRIUNGHIUL LUI PASCAL

$(a+b)^n \rightarrow$ det. coef. cunihor

A. MATRICE

	0	1	2	3	4	5	6			
$(a+b)^0$	1									
$(a+b)^1$		1	1							
$(a+b)^2$			1	2	1					
$(a+b)^3$				1	3	3	1			
4		1	4	6	4	1				
5			1	5	10	10	5	1		
6				1	6	15	20	15	6	1

$$C_m^k = C_{m-1}^{k-1} + C_{m-1}^k$$

$$\Downarrow$$

$$a_{i,j} = a_{i-1,j-1} + a_{i-1,j}$$

B. CU 2 VECTORI

	0	1	2	3	4	
w	0	1				
w	x	1	1	1		
w	x	2	1	2	1	
w	x	3	1	3	3	1
v	4	1	4	6	4	1

C. CU UN SINGUR VECTOR

U:

$i = 0;$	X					
$i = 1;$	X	X				
$i = 2;$	X	X	X			
$i = 3;$	1	3	3	1		
$i = 4;$	1	4	6	4	1	

0 1 2 3 4 5

for ($i = 0; i < n; i++$)

}

$v[0] = v[i] = 1;$

for ($j = i+1; j >= 1; j--$)

{

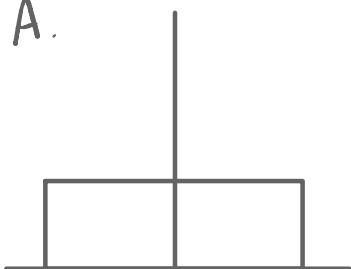
$v[j] = v[j] + v[j-1];$

}

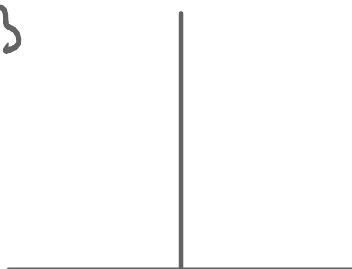
}

⑤ TURNURILE LUI HANOI

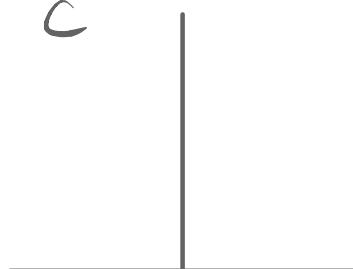
A.



B



C



$n = 1 \Rightarrow 1$ mutare



$n = 3 \Rightarrow 7$ mutări

$2^m - 1$ - nr minimum de mutări
pentru n discuri

$O(2^m)$

⑥ CODUL LUI CEZAR

0	1	2	3	4	5	6	...	22	23	24	25
A	B	C	D	E	F	G	...	W	X	Y	Z
↓	↓	↓	↓	↓	↓	↓		↓	↓	↓	↓
D	E	F	G	H	I	J		Z	A	B	C

$$C = (C + F) \cdot 1 \cdot 25$$

⑦ NUMERE FRUMOASE

- nr care are ca factor primi doar $\neq 2, 3, 5$

1	2	3	4	5	6	7	8	9	10	11
1	2	3	4	5	6	7	8	9	10	12

$$K_2 \quad \text{---} \overset{\times 2}{\bullet} \quad \overset{\times 2}{\bullet} \quad \overset{\times 2}{\bullet} \quad \overset{\times 2}{\bullet} \quad \overset{\times 2}{\bullet}$$

$$K_3 \quad \text{---} \overset{\times 3}{\bullet} \quad \overset{\times 3}{\bullet}$$

$$K_5 \quad \text{---} \overset{\times 5}{\bullet} \quad \overset{\times 5}{\bullet}$$

⑧ POLINOAME

$$P(x) = a_0 + a_1 x + a_2 x^2 + \dots + a_n x^n$$

$$\Leftrightarrow a_n \cdot x^n + a_{n-1} \cdot x^{n-1} + \dots + a_1 \cdot x + a_0$$

$$\Leftrightarrow (((a_n \cdot x) + a_{n-1}) \cdot x + a_{n-2}) \cdot x + \dots a_1 \cdot x + a_0$$

$$n = 3$$

$$P(x) = a_0 + a_1 \cdot x + a_2 \cdot x^2 + a_3 \cdot x^3$$

$$P(x) = (((a_3 \cdot x) + a_2) \cdot x + a_1) \cdot x + a_0$$

$$n = 0;$$

for (i = n; i >= 0; i--)

$$n = n^* x + a[i];$$

Nr. minun de inmultiri:

$$= n$$

⑦ ORDINEA OPERAȚIILOR

$$\begin{aligned} a &= x + + + + + y \\ &= (x++) + (\underline{++y}) \end{aligned}$$

$$\left\{ \begin{array}{l} y = y + 1; \\ a = x + y; \\ x = x + 1; \end{array} \right.$$

⑧ EXPRESIA CONDITIONATĂ

$$1) \quad x = (n \cdot 1 \cdot a == 0) ? 1 : 0;$$

(\Rightarrow)

if ($n \cdot 1 \cdot a == 0$) $x = 1;$

else $x = 0;$

$$2) \quad \text{int } f(\text{int } x) \Rightarrow |x|$$

{

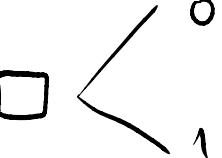
return ($x > 0$) ? $x : -x;$

}

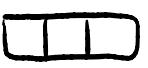
3)

$$\max = (a > b) ? ((a > c) ? a : c) : ((b > c) ? b : c);$$

⑤ BITI

$n = 1$  $\{0, 1\}$
 $\Rightarrow 2 \text{ valori} = 2^1$

$n = 2$  $\{0, 1\} \times \{0, 1\}$
 $\begin{array}{l} 00 \\ 01 \\ 10 \\ 11 \end{array}$
 $\Rightarrow 4 \text{ valori} = 2^2$

$n = 3$  $\{0, 1\} \times \{0, 1\} \times \{0, 1\}$
 $\begin{array}{l} 000 \\ 001 \\ 010 \\ 011 \\ 100 \\ 101 \\ 110 \\ 111 \end{array}$
 $\rightarrow 8 \text{ valori} = 2^3$
 $111_2 \rightarrow 7 = 2^0 \cdot 1 + 2^1 \cdot 1 + 2^2 \cdot 1$

$$1732 = 2 \cdot 10^0 + 3 \cdot 10^1 + 7 \cdot 10^2 + 1 \cdot 10^3$$

$$11_2 = 1 \cdot 2^0 + 1 \cdot 2^1 = 3_{10}$$

n	Nr. Valori	Domäne in \mathbb{N}	Domäne in \mathbb{Z}
1	$2 = 2^1$	$[0, 1]$	
2	$4 = 2^2$	$[0, 2^2 - 1]$	$[-2, 1]$
3	$8 = 2^3$	$[0, 2^3 - 1]$	$[-4, 3] = [-2^2, 2^2 - 1]$
4	$16 = 2^4$	$[0, 2^4 - 1]$	$[-8, 7] = [-2^3, 2^3 - 1]$
...			
n	2^n	$[0, 2^n - 1]$	$[-2^{n-1}, 2^{n-1} - 1]$

⑩ TIPUL LONG

1 byte = 8 biti

long = 4 B = 32 biti $\rightarrow 2^{32}$

$[0, 2^{32}-1] \in \mathbb{N}$
unsigned

$[-2^{31}, 2^{31}-1] \in \mathbb{Z}$
int

⑪ OPERATII PE BITI

OPERATORUL DE NEGATIE \sim

\rightarrow bitul 0 devine 1

\rightarrow bitul 1 devine 0

Ex. $\sim 133 = -134$

$$133_{10} = 0000000010000101_2$$

$$\sim 133 = 111111101111010_2$$

OPERATORUL DE CONJUNCȚIE $\&$ (\wedge)

Ex. $13 \& 151 = 5$

$$0 \& 0 = 0$$

$$0 \& 1 = 0$$

$$1 \& 0 = 0$$

$$1 \& 1 = 1$$

$$\begin{array}{r} 1101 \\ 10010111 \\ \hline 00000101 \end{array} \rightarrow 5$$

OPERATORUL DE DISJUNCȚIE \mid (sau)

$$0 \mid 0 = 0$$

ex

$$0 \mid 1 = 1$$

$$13 \mid 151 = 159$$

$$1 \mid 0 = 1$$

$$1 \mid 1 = 1$$

OPERATORUL DE DISJUNCȚIE EXCLUSIVĂ

\wedge (sau excluderă)

$$0 \wedge 0 = 0$$

ex

$$0 \wedge 1 = 0$$

$$13 \wedge 151 = 154$$

$$1 \wedge 0 = 0$$

$$1 \wedge 1 = 0$$

OPERATORUL DEPLASARE LA STÂNGA \ll

$\rightarrow n \ll k$ deplasează pe stânga bitul n în n cu k pozitii

$$\text{ex. } 13 \ll 3 = 104 = 13 \cdot 2^3$$

$$1101 \rightarrow 1101000$$

$$n \ll k = n \cdot 2^k$$

OPERATORUL DEPLASARE LA DREAPTA \gg

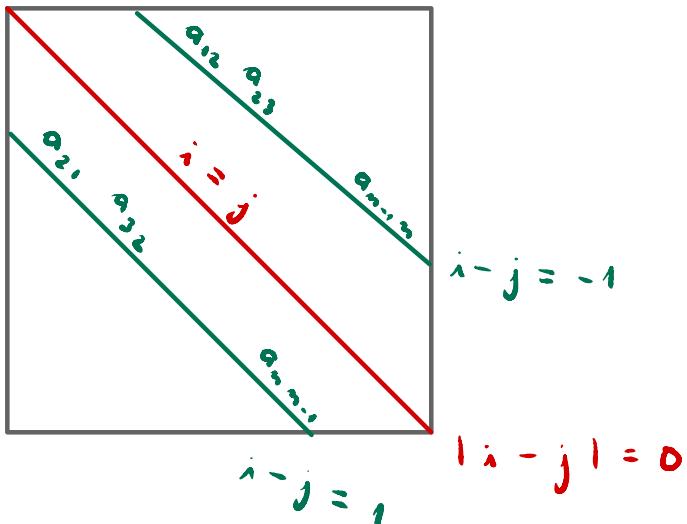
$\rightarrow n \gg k$ deplasează cu dreapta lită m m
cu k pozitie

$$\text{ex. } 133 \gg 3 = 16$$

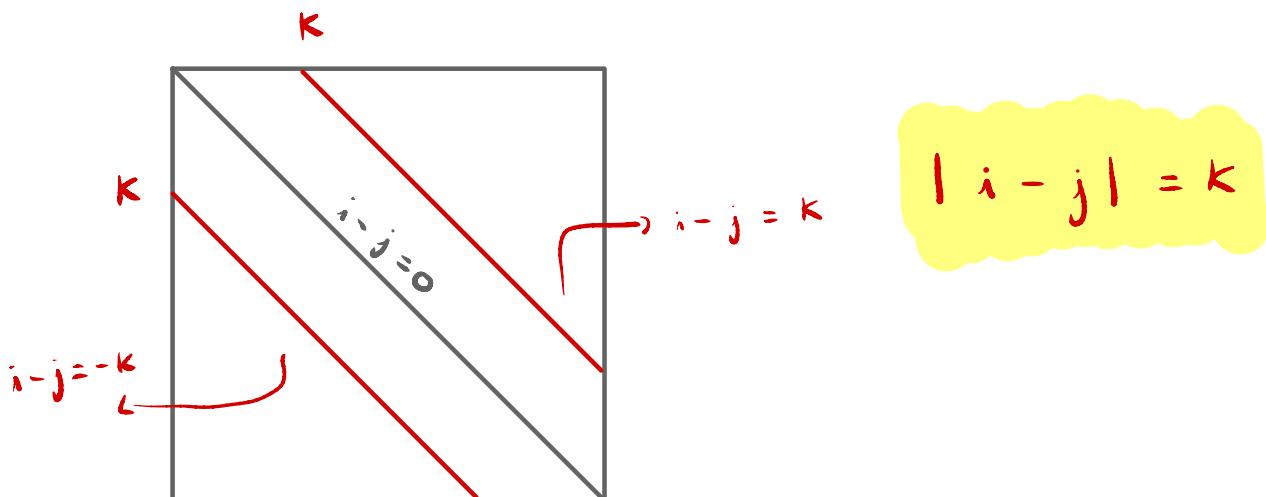
$$10000101 \rightarrow 10000$$

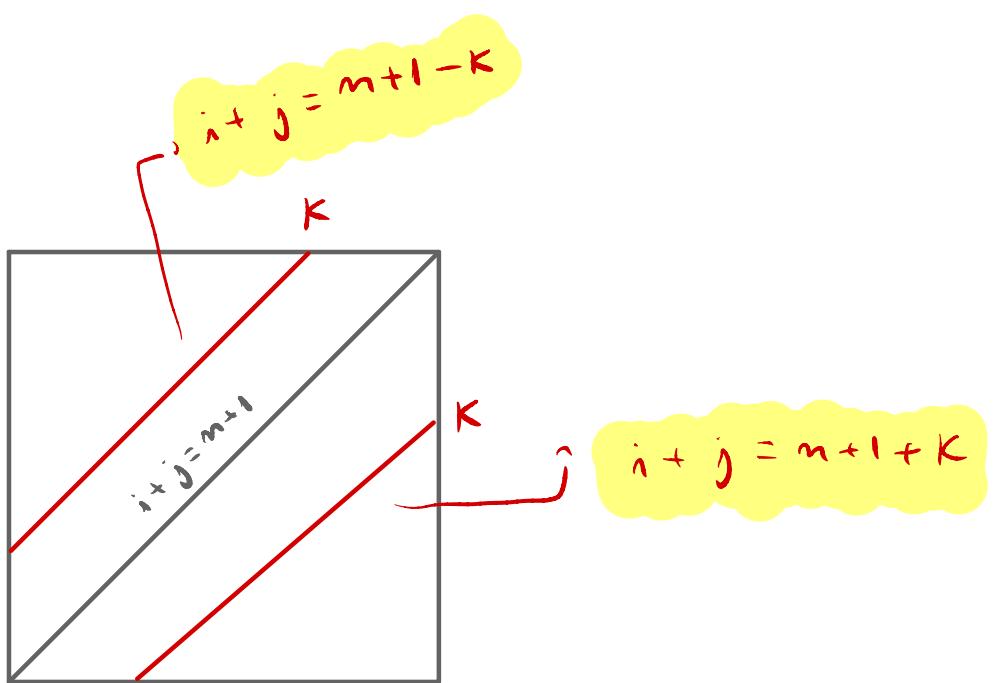
$$n \gg k = n / 2^k$$

(12) SEMIDIAGONALE



Generalizare a K diagonale





⑬ DETERMINANTUL UNEI MATRICI

$$\begin{vmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \\ 13 & 14 & 15 & 16 \end{vmatrix} = 1 \cdot (-1)^{1+1} \begin{vmatrix} 6 & 7 & 8 \\ 10 & 11 & 12 \\ 11 & 15 & 16 \end{vmatrix}$$

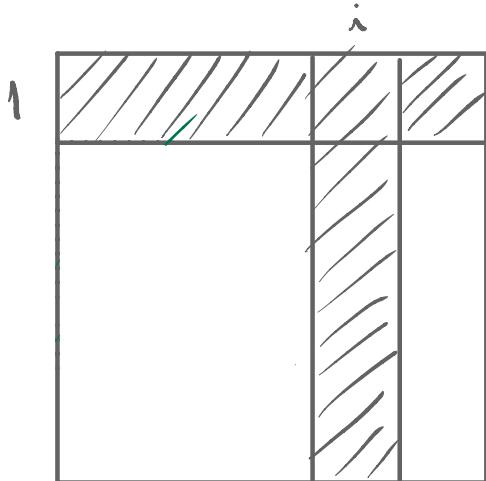
$$+ 2 \cdot (-1)^{1+2} \begin{vmatrix} 5 & 7 & 8 \\ 9 & 11 & 12 \\ 13 & 15 & 16 \end{vmatrix}$$

$$+ 3 \cdot (-1)^{1+3} \begin{vmatrix} 5 & 6 & 8 \\ 9 & 10 & 12 \\ 13 & 14 & 16 \end{vmatrix}$$

$$+ 4 \cdot (-1)^{1+4} \begin{vmatrix} 5 & 6 & 7 \\ 9 & 10 & 11 \\ 13 & 14 & 15 \end{vmatrix}$$

$$\det(A, n) = a_{11} \cdot (-1)^{1+1} \begin{vmatrix} (1, 1) \end{vmatrix} + a_{12} \cdot (-1)^{1+2} \cdot \begin{vmatrix} (1, 2) \end{vmatrix} \\ + \dots + a_{1n} \cdot (-1)^{n+1} \cdot \begin{vmatrix} (1, n) \end{vmatrix}$$

↙ el. linia 1^a cu dimensiune n



$$b_{i-1, j} = a_{i, j}$$

$$b_{i-1, j-1} = a_{i, j}$$

!

14 NUMERE DIVIZIBILE DINTR-UN INTERVAL

$$[1, n] : x \rightarrow \left[\frac{n}{x} \right]$$

$$[a, b] : x \rightarrow \underbrace{\left[\frac{1}{x}, \frac{2}{x}, \dots, \frac{a-1}{x} \right]}_{\left[\lceil (a-1)/x \rceil \right]} \cup \underbrace{\left[\frac{a}{x}, \frac{a+1}{x}, \dots, \frac{b}{x} \right]}_{\left[\lceil b/x \rceil - \lceil (a-1)/x \rceil \right]}$$

$$[a, b] : x \rightarrow \left[\frac{b}{x} \right] - \left[\frac{a-1}{x} \right]$$

(15) Criteriu de divizibilitate cu $b - 1$

Un număr este divizibil cu $b - 1$ dacă suma cifrelor numărului în baza b este divizibilă cu $b - 1$.

Ex.

$$12051 \div (4 - 1)$$

— //

1	2	0	5	1	3
3	0	1			
7	5	3			
1	8	8			
4	7	3			
1	1	3			
2	2				
0					

$$12051_{10} = 2330103_4$$

$$\text{Suma } cf = 12 \div 3$$

$$\Rightarrow n \div 13$$

(18) PUTEREA LA CARE APARE UN NR. PRIM x IN SCRIREA LUI $n!$

$x - nr\ prim$

$$\left[\frac{n}{x} \right] + \left[\frac{n}{x^2} \right] + \left[\frac{n}{x^3} \right] + \dots + \left[\frac{n}{x^k} \right], \quad x^k \leq n$$

Ex.

$$n = 33$$

$$x = 2$$

2	4	6	8	10	12	14	16	18	20
22	24	26	28	30	32				

$$\left[\frac{33}{2} \right] + \left[\frac{33}{4} \right] + \left[\frac{33}{8} \right] + \left[\frac{33}{16} \right] + \left[\frac{33}{32} \right]$$

$$= 16 + 8 + 4 + 2 + 1 = 31$$

19) ULTIMA CIFRA NENULĂ DE LA SFÂRȘITUL

LUI m!

$$m = 52$$

	0	1	2	3	4	5	6	7	8	9
0	1	2	6	4	2	2	4	2	8	.
10	8	8	6	8	2	3	8	6	8	2
20	4	4	8	4	6	5	3	1	8	2
30	6	6	2	6	4	4	8	4	6	.
40	4	4	8	4	6	7	2	4	2	8
50	4	4	8							

$$15 \cdot 2 = \underline{30}$$

20) PERMUTĂRI CU REPETIȚIE

$$\frac{n!}{n_1! n_2! \dots n_k!} \quad \text{a.i. } n_1 + n_2 + \dots + n_k = n$$

Ex. $\{A, A, A, B, B, C, C, C, C, D\}$

nr de permutări distincte:

$$\frac{10!}{3! \cdot 2! \cdot 4! \cdot 1!} = \frac{10 \cdot 9 \cdot 8 \cdot 7 \cdot 6 \cdot 5}{6 \cdot 2} = \dots$$

PERMUTĂRI CU PUNCTE FIXE

→ o permutare cu puncte fixe dăza $\exists 1 \leq i \leq n$
așă $p_i(i) = i$

nr. permutări cu puncte fixe :

$$C_m^1 (m-1)! - C_m^2 (m-2)! + \dots + (-1)^{m-1} C_m^m$$

$$\frac{m!}{1!} - \frac{m!}{2!} + \frac{m!}{3!} + \dots + (-1)^{m-1} \frac{m!}{m!}$$

PERMUTĂRI FĂRĂ PUNCTE FIXE

- $!n$ ($left factorial$) nr. de permutări fixe
- $!n = n! - \text{nr. permutărilor CU puncte fixe}$

$$!n = n! - \frac{n!}{1!} + \frac{n!}{2!} - \frac{n!}{3!} + \dots + (-1)^n \frac{n!}{n!}$$

$$!n = n! \left[1 - \frac{1}{1!} + \frac{1}{2!} - \frac{1}{3!} + \dots + \frac{(-1)^n}{n!} \right]$$

(21) A K-a SUBMULTIME IN ORDINE LEXICOGRAFICA

$\{1, 2, 3, \dots n\}$

1. $\{\}$

2. $\{1\} C_{n-3}^0 + C_{n-3}^1 + C_{n-3}^2 + C_{n-3}^3 + \dots + C_{n-3}^{n-3} = 2^{n-3}$

3. $\{1, 2\}$

4. $\{1, 2, 3\} \quad \emptyset \quad | \quad 4 \quad | \quad 4, 5 \quad | \quad 4, 5, 6 \quad | \quad \dots$

5. $5 \quad | \quad 4, 6 \quad | \quad \dots$

6. $6 \quad | \quad \dots$

... $| \quad n-1, n$

$$3 + 2^{n-3}$$

$$4 + 2^{n-3} \quad \{1, 2, 4\}$$

Ex. $n = 10$

1. $\{\}$

2. $\{1\}$

3. $\{1, 2\}$

4. $\{1, 2, 3\}$

$$\dots 2^7 = 128$$

131

132. $\{1, 2, 4\}$

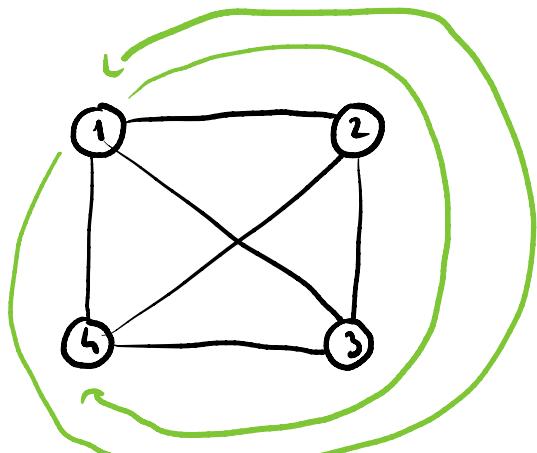
195 $\dots 2^6 = 64$

196. $\{1, 2, 5\}$

23 NUMARUL CICLURILOR HAMILTONIENE INTR-UN

GRAF COMPLET

$$n = 4$$



$$\frac{(n-1)!}{2}$$

$$1234 = 2341 = 3412 = 4123$$

$$\Rightarrow \frac{4!}{4} = 3!$$

$$1234 = 1432 \Rightarrow \frac{3!}{2}$$

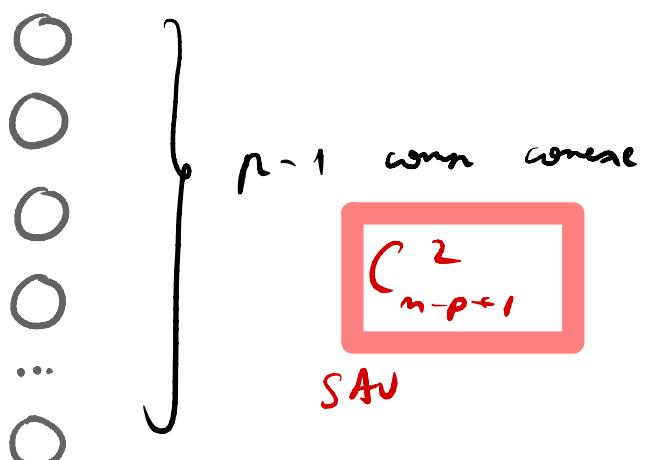
24 NR. MAXIM DE MUCHII INTR-UN GRAF

CU n COMPONENTE CONEXE

$$\begin{aligned} n - (n-1) \\ = n - n + 1 \\ \text{moduri} \end{aligned}$$

în subgraf complet

$$\frac{k(k-1)}{2} \text{ muchii} \Rightarrow$$



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