Kombinatorikai algoritmusok

(Horváth Gyula és Szlávi Péter előadásai felhasználásával)

Kombinatorikai algoritmusok

A kombinatorika: „*egy véges halmaz elemeinek valamilyen szabály alapján történő csoportosításával, kiválasztásával, sorrendbe rakásával foglalkozik*” (Wikipedia)

Tipikus kombinatorikai feladatok:

- részhalmazok képzése
  - **összes** felsorolása
  - **partícionálás** céljából – halmazfelbontás
  - **adott tulajdonságú** (pl. adott számosságú) megadása

Kombinatorikai algoritmusok

Tipikus kombinatorikai feladatok:

- halmazok **számosságá**nak a meghatározása
- halmazok (~sorozatok) felsorolása – **összes eleme**
- halmazok (~sorozatok) valamely elemének generálása – **egy eleme**
  - **szabályos** sorrend szerinti i. elem
  - adottra **következő**, **előző**
  - „**véletlen**” elem

Kombinatorikai algoritmusok

Kombinatorikai feladattípusok:

- permutációk (ismétlés nélküli, ismétléses)
- kombinációk (ismétlés nélküli, ismétléses)
- variációk (ismétlés nélküli, ismétléses)
- részhalmazok
- diszjunkt felbontások (N elemű halmaz diszjunkt felbontásai K halmazra, vagy természetes számra: $N=x_1+\ldots x_k$, ahol $x_i \geq 0$)
- partíciók (N elemű halmaz diszjunkt felbontásai, vagy természetes számra: $N=x_1+\ldots x_m$, ahol $x_i>0$)

Kombinatorikai algoritmusok – az elemszám kiszámítása

Ismétlés nélküli kombinációk

- Ismert képlet: $\binom{N}{K} = \frac{N!}{K!*(N-K)!} = \frac{N*(N-1)*\ldots*1}{K*(K-1)*\ldots*1*(N-K)*(N-K+1)*\ldots*1}$
- Rekurzív definíció 1: N elemből K elem választása
  - az első elemet választjuk, majd még N-1 elemből válasz-tunk K-1 elemet. vagy
  - az első elemet nem választjuk és a maradék N-1 elemből választunk K elemet.

$\rightarrow$ B(N,K)=B(N-1,K-1)+B(N-1,K) .

Kombinatorikai algoritmusok – az elemszám kiszámítása

Ismétlés nélküli kombinációk

```
B(N,K):
  Ha K=0 vagy K=N akkor B:=1
  különben B:=B(N-1,K-1)+B(N-1,K)
Függvény vége.
```

De táblázatkitöltéssel hatékonyabb!

```
Binom(N,K):
 Szélek feltöltése
 Ciklus i=1-től N-ig
  Ciklus j=1-től i-ig
   B(i,j):=B(i-1,j-1)+B(i-1,j)
  Ciklus vége
 Ciklus vége
Eljárás vége.
```

Kombinatorikai algoritmusok – az elemszám kiszámítása

Ismétlés nélküli kombinációok

- Rekurzív definíció 2: N elemből K elem választása
  - először kiválasztunk K-1 elemet, majd
  - a maradék N-K+1 elemből kell egyet választani (de így minden kombináció pontosan K-féleképpen áll elő), tehát jön még egy K-val osztás

→ B(N,K)=B(N,K-1)*(N-K+1)/K

Kombinatorikai algoritmusok – az elemszám kiszámítása

Ismétlés nélküli kombinációk

```
B(N,K):
  Ha K=0 akkor B:=1
  különben B:=B(N,K-1)*(N-K+1)/K
Függvény vége.
```

Táblázatkitöltéssel lényegében nem hatékonyabb!

```
Binom(N,K):
 B(0):=1
 Ciklus i=1-től N-ig
   B(i):=B(i-1)*(N-i+1)/i
 Ciklus vége
Függvény vége.
```

Kombinatorikai algoritmusok – az elemszám kiszámítása

Elsőfajú Euler számok

- E(n,k) az első n természetes szám azon permutációi száma, ahol pontosan k emelkedés van (emelkedés van az i-edik helyen, ha $x_i<x_{i+1}$).
  - n-1 elem összes olyan permutációja, ahol pontosan k növekedés van: az n-edik számot a sorozat elejére vagy egy emelkedésbe tesszük;
  - n-1 elem összes olyan permutációja, ahol pontosan k-1 emelkedés van: n-edik elemet a sorozat végére vagy egy nem emelkedő helyre tesszük.

$$\rightarrow \quad E(n,k)=(k+1)E(n-1,k)+(n-k)E(n-1,k-1)$$

Kombinatorikai algoritmusok – az elemszám kiszámítása

Másodfajú Euler számok

- E(n,k) az {1,1,2,2,…,n,n} sorozat olyan permutációi száma, amelyekben tetszőleges **m** szám két előfordulása között csak náluk nagyobb szám fordulhat elő és pontosan k emelkedő részsorozata van.

Vegyük észre, hogy amikor áttérünk n-1-ről n-re, akkor a két n-edik számot csak egymás mellé szúrhatjuk be!

Kombinatorikai algoritmusok – az elemszám kiszámítása

Másodfajú Euler számok

- n-1 elem összes olyan permutációja, ahol pontosan k növekedés van: az n-edik számpárt a sorozat elejére vagy egy emelkedésbe tesszük;
- n-1 elem összes olyan permutációja, ahol pontosan k-1 emelkedés van: n-edik számpárt a sorozat végére vagy egy nem emelkedő helyre tesszük (ebből 2*n-1-k van).

$$\rightarrow \quad E(n,k)=(k+1)E(n,k-1)+(2n-1-k)E(n-1,k-1)$$

Kombinatorikai algoritmusok – az elemszám kiszámítása

n\k	0	1	2	3	4	5	6	7
0	1							
1	1	0						
2	1	2	0					
3	1	8	6	0				
4	1	22	58	24	0			
5	1	52	328	444	120	0		
6	1	114	1452	4400	3708	720	0	
7	1	240	5610	32120	58140	33984	5040	0

Kombinatorikai algoritmusok – az elemszám kiszámítása

Ismétlés nélküli permutációk száma megszorítással – minden elem a sorszám szerinti helyétől legfeljebb egyet távolodhat el

- Amikor a permutációk előállításában az I. elemnél tartunk, akkor két döntési lehetőségünk van:
  - az I marad a helyén, majd permutációk I+1-től,
  - az I+1 és az I helyet cserél , majd permutációk I+2-től.

```
Permszám(X,n):
  Ha n<2 akkor Permszám:=n
      különben Permszám:=Permszám(n-1)+
                         Permszám(n-2)
Eljárás vége.
```

Kombinatorikai algoritmusok – az összes előállítása

Összes ismétlés nélküli permutáció

- Backtrack: $\forall i$ $(1 \leq i \leq n)$: $\forall j$ $(1 \leq j < i)$: $X_j \neq X_i$

Összes ismétlés nélküli kombináció

- Backtrack: $\forall i$ $(1 \leq i \leq k)$: $\forall j$ $(1 \leq j < i)$: $X_j < X_i$

Összes ismétléses kombináció

- Backtrack: $\forall i$ $(1 \leq i \leq k)$: $\forall j$ $(1 \leq j < i)$: $X_j \leq X_i$

Összes partíció

- Olyan K-jegyű számok, ahol a számjegyek összege pontosan N.

Összes diszjunkt felbontás

- N felbontása pozitív (>0) számok összegére.

Kombinatorikai algoritmusok – az összes előállítása

Összes ismétlés nélküli permutáció

```
1  2  3   …   N-1   N
```

1,2,3,…,N-1,N

1,2,3,…,N,N-1

…

N,1,2,3,…,N-1

…

N,N-1,…,3,2,1

Kombinatorikai algoritmusok – az összes előállítása

Összes ismétlés nélküli permutáció ciklussal

```
Permutációk(N):
  i:=1; X():=(0,…,0)
  Ciklus amíg i≥0 és i≤N
    Ha vanjóeset(i,van,j)
       akkor X(i):=j
             Ha i<n akkor i:=i+1
             különben Kiír: X; X(i):=0; i:=i-1
    különben X(i):=0; i:=i-1
  Ciklus vége
Eljárás vége.
```

Kombinatorikai algoritmusok – az összes előállítása

Összes ismétlés nélküli permutáció ciklussal

```
vanjóeset(i,van,j):
  j:=X(i)+1
  Ciklus amíg j≤N és rossz(i,j)
    j:=j+1
  Ciklus vége
  van:=j≤N
Eljárás vége.
```

Kombinatorikai algoritmusok – az összes előállítása

Összes ismétlés nélküli permutáció ciklussal

```
rossz(i,j):
  k:=1
  Ciklus amíg k<i és j≠X(k)
    k:=k+1
  Ciklus vége
  rossz:=k<i
Eljárás vége.
```

Kombinatorikai algoritmusok – az összes előállítása

Összes ismétlés nélküli permutáció rekurzív backtrack

- Az összes olyan szám-n-es előállítása visszalépéses kereséssel, ahol $i \neq j \rightarrow x_i \neq x_j$

```
Perm(i,n,X,db,Y):
  Ha i=n+1 akkor db:=db+1; Y(db):=X
  különben
    Ciklus j=1-től n-ig
      Ha nem Rossz(i,j)
         akkor X(i):=j; Perm(i+1,n,X,Db,Y)
    Ciklus vége
Eljárás vége.
```

Kombinatorikai algoritmusok – az összes előállítása

Összes ismétlés nélküli permutáció rekurzív backtrack

```
Rossz(i,j):
  k:=1
  Ciklus amíg k<i és X(k)≠j
    k:=k+1
  Ciklus vége
  Rossz:=(k<i)
Függvény vége.
```

Kombinatorikai algoritmusok – az összes előállítása

Összes ismétlés nélküli permutáció rekurzívan

- Ha N-1 elem összes permutációja kész, akkor szúrjuk be az N-et minden lehetséges helyre, mindegyikbe!

```
Permutáció(x,i,n):
  Ha i>n akkor Ki: x
  különben x(i):=i; Permutáció(x,i+1,n)
           Ciklus j=i-1-től 1-ig -1-esével
             Csere(x(j),x(j+1))
             Permutáció(x,i+1,n)
           Ciklus vége
Eljárás vége.
```

Kombinatorikai algoritmusok – az összes előállítása

Összes ismétlés nélküli permutáció megszorítással

- Az 1,…,n sorozat összes olyan permutációját állítsuk elő, ahol minden elem maximum 1 hellyel mozdul el a helyéről, azaz $i-1 \leq X(i) \leq i+1$!

A backtrack-es megoldást könnyű módosítani:

`Ciklus j=1-től n-ig` helyett

`Ciklus j=i-1-től i+1-ig` kell a programba!

A rekurzív megoldást ezzel szemben újra kell gondolni!

Kombinatorikai algoritmusok – az összes előállítása

Összes ismétlés nélküli permutáció megszorítással

- Amikor a permutációk előállításában az I. elemnél tartunk, akkor két döntési lehetőségünk van:
  - az I marad a helyén, majd permutációk I+1-től,
  - az I+1 és az I helyet cserél , majd permutációk I+2-től.

```
Permutáció(X,i,n):
 Ha i>n akkor Ki: X
  különben X(i):=i; Permutációk(X,i+1,n)
           Ha i<n akkor X(i):=i+1; X(i+1):=i
                        Permutációk(X,i+2,n)
Eljárás vége.
```

Az összes permutáció alkalmazása

Feladat:

Jól ismert fejtörő, amelyben egy aritmetikai művelet kapcsol egybe szavakat. A feladat az, hogy a szavak egyes betűinek feleltessünk meg egy számjegyet úgy, hogy a művelet helyes eredményt szolgáltasson a szavakon.

Pl. SEND + MORE = MONEY.

Megoldás:

A szavakban előforduló jelekhez (SENDMORY) keressük a 0..9 számjegyek egyértelmű hozzárendelését.

Az összes permutáció alkalmazása

- **1. megoldási ötlet (**„algebrai hozzáállás”**):**

  D+E=Y, N+R=E, … M=1

  D+E=10+Y, N+R+1=E, … M=1

- **2. megoldási ötlet:**
  - Az összes permutáció algoritmusára építünk.
  - A Jó eljárás ellenőrzi a permutáció – a feladat szempontjából való – helyességét, és gondoskodik az esetleges megoldás gyűjtéséről vagy kiírásáról.

Az összes permutáció alkalmazása

A megfelelőség a (*) SEND + MORE – MONEY = 0 egyenletre. Ha

- 'S‘ X(1) értékű, akkor a (*)-ban X(1) *1000-rel van jelen;
- 'E' X(2) értékű, akkor X(2)*(100+1-10)=X(2)*91-gyel;
- 'N' X(3) értékű, akkor X(3)*(10-100)= X(3)*(-90)-nel;
- 'D' X(4) értékű, akkor X(4)*(1)-gyel;
- 'M' X(5) értékű, akkor X(5)*(1000-10000)=X(5)*(-9000)-rel;
- 'O' X(6) értékű, akkor X(6)*(100-1000)= X(6)*(-900)-zal;
- 'R' X(7) értékű, akkor X(7)*10-zel;
- 'Y' X(8) értékű, akkor X(8)*(-1)-gyel van jelen.

- továbbá az S és az M betűhöz nem ren-delhetünk nullát, azaz $X(1)\neq 0$ és $X(5)\neq 0$!

Az összes permutáció alkalmazása

```
Megoldás(i,n,X,db,Y):
  Ha i=n+1 és jó(X) akkor db:=db+1; Y(db):=X
  különben
    Ciklus j=0-tól 9-ig
      Ha nem Rossz(i,j)
         akkor X(i):=j; Megoldás(i+1,n,X,Db,Y)
    Ciklus vége
Eljárás vége.
jó(X):
  jó:= (X(1) *1000+X(2)*91+X(3)*(-90)+X(4)*1+X(5)*(-9000)+
        X(6)*(-900)+X(7)*10+X(8)*(-1)=0 és X(1)≠0 és X(5)≠0)
Függvény vége.
```

Ha a konstansokat egy Z vektorban tárolnánk, akkor a Jó függvényben X és Z skaláris szorzatát kellene kiszámolnunk.

Összes részhalmaz

- Feleltessük meg a részhalmazokat kettes számrendszerbeli számoknak:

{}	→	0...0000
{1}	→	0...0001
{2}	→	0...0010
{1,2}	→	0...0011
{3}	→	0...0100
{1,3}	→	0...0101
{2,3}	→	0...0110
{1,2,3}	→	0...0111
…		

Kombinatorikai algoritmusok – az összes előállítása

Összes részhalmaz (kettes számrendszerbeli számként)

```
Részhalmazok(n,Y):
  Ciklus i=0-tól 2ⁿ-1-ig
    k:=i
    Ciklus j=1-től n-ig
      X(j):=k mod 2; k:=k div 2
    Ciklus vége
    Y(i+1):=X
  Ciklus vége
Eljárás vége.
```

Kombinatorikai algoritmusok – az összes előállítása

Összes részhalmaz (részhalmazként)

```
Részhalmazok(n,Y):
  Ciklus i=0-tól 2^n-1-ig
    X.db:=0
    Ciklus j=1-től n-ig
      Ha k mod 2=1 akkor X.db:=X.db+1; X.t(X.db):=j
      k:=k div 2
    Ciklus vége
    Y(i+1):=X
  Ciklus vége
Eljárás vége.
```

Összes ismétléses variáció

- Feleltessük meg a variációkat N alapú számrendszerbeli K jegyű számoknak!

```
1,...,1,1     → 0...0000
1,...,1,2     → 0...0001
1,...,1,n     → 0...001(n-1)
1,...,2,1     → 0...0010
1,...,2,n     → 0...001(n-1)
n,...,n,n     → (n-1)...(n-1)
```

Kombinatorikai algoritmusok – az összes előállítása

Összes ismétléses variáció

```
Variációk(n,k,Y):
  Ciklus i=0-tól n^k-1-ig
    m:=i
    Ciklus j=1-től n-ig
      X(j):=m mod n; m:=m div n
    Ciklus vége
    Y(i+1):=X
  Ciklus vége
Eljárás vége.
```

Kombinatorikai algoritmusok – az összes előállítása

Összes partíció

- Az N számot bontsuk fel az összes lehetséges módon K nemnegatív szám összegére!

Felbontás(N): $\left\{X \middle| N=\sum X_i, \quad ahol \quad X_i>0\right\}$, azaz olyan X sorozatok halmaza, amelyek összege éppen N.

A megoldások alakja:

1,x,…,x

2,x,…,x

…

N,x,…,x

Kombinatorikai algoritmusok – az összes előállítása

Összes partíció

```
Felbontás(n,k,i,db,Y):
  Ciklus j=1-től n-1-ig
    X(i):=j; Felbontás(n-X(i),i+1)
  Ciklus vége
  X(i):=n; X(i+1..k):=0
  db:=db+1; Y(db):=X
Eljárás vége.
```

Kombinatorikai algoritmusok – az összes előállítása

Összes partíció

- Az N számot bontsuk fel az összes lehetséges módon K monoton csökkenő sorrendű nemnegatív szám összegére!

Felbontás(N): $\left\{ X \middle| N = \sum X_i, \quad ahol \quad X_i > 0 \right\}$, azaz olyan X sorozatok halmaza, amelyek összege éppen N és $X_i \geq X_{i+1}$.

A megoldások alakja:

1,x,…,x

2,x,…,x

…

N,x,…,x

Kombinatorikai algoritmusok – az összes előállítása

Összes partíció

```
Felbontás(n,k,i,db,Y):
  Ciklus j=1-től min(n-1,X(i-1))-ig
    X(i):=j; Felbontás(n-X(i),i+1)
  Ciklus vége
  Ha X(i-1)≥n akkor X(i):=n; X(i+1..k):=0
                    db:=db+1; Y(db):=X
Eljárás vége.
```

Kombinatorikai algoritmusok – az I. előállítása

I-edik permutáció

- Vegyük pl. a minimum-kiválasztásos rendezést!
- Jelentse F(j) a j-edik lépésbeli csere távolságát!
- Az F vektor alapján az eredeti sorrend visszaállítható!
- Minden i természetes számhoz ($0\le i<n!$) különböző F vektor tartozik, az i szám felírása faktoriális számrendszerben.

Kombinatorikai algoritmusok – az I. előállítása

Rendezés

```
Rendezés(n):
  Ciklusi=1-től n-ig
    min:=i
    Ciklus j=i+1-től n-ig
      Ha X(j)<X(min) akkor min:=j
    Ciklus vége
    Csere(X(i),X(min)); Táv(i):=min-i
  Ciklus vége
Eljárás vége.
```

Kombinatorikai algoritmusok – az I. előállítása

Rendezés: a Táv vektor alapján az eredeti sorrend visszaállítható

```
Rendezés vissza(n):
  Ciklus j=n-1-től 1-ig -1-esével
    Csere(x(j),x(j+Táv(j)))
  Ciklus vége
Eljárás vége.
```

Kombinatorikai algoritmusok – az I. előállítása

A Táv vektor elemei:

Táv(n-1)∈{0,1}

Táv(n-2)∈{0,1,2}

…

Táv(1)∈{0,1,…,n-1}

Ez egy olyan n-1 számjegyből álló szám, amelynek helyiérté-kenként más a számrendszer alapszáma – faktoriális szám-rendszer.

Kombinatorikai algoritmusok – az I. előállítása

Az i-edik permutáció:

Az i felírása faktoriális számrendszerben, majd cserék.

```
Permutáció(i,n):
  x:=(1,2,...,n); K:=2
  Ciklus j=n-1-től 1-ig -1-esével
    t:=i mod K; i:=i div K
    Csere(x(j),x(j+t-1))
    K:=K+1
  Ciklus vége
Eljárás vége.
```

Probléma: nem lexikografikus sorrend!

Kombinatorikai algoritmusok – az I. előállítása

Az i-edik permutáció:

Tudjuk, hogy pontosan (n-1)! 1-gyel, 2-vel, ... kezdődő permutáció van.

```
Permutáció(i,n):
  x:=(1,2,...,n); i:=i-1; m:=n-1
  Ciklus j=1-től n-1-ig
    k:=i div fakt(m); i:=i mod fakt(m)
    Eltol(j,k); m:=m-1
  Ciklus vége
Eljárás vége.
```

Kombinatorikai algoritmusok – az I. előállítása

Az i-edik permutáció, amelyben minden elem legfeljebb eggyel mozdulhatott el:

```
Permutáció(i,n):
  m:=n-1; j:=1;
  Ciklus amíg j<n
    Ha i<Fib(m) akkor x(j):=j; m:=m-1; j:=j+1
    különben i:=i-Fib(m); x(j):=j+1; x(j+1):=j
             m:=m-2; j:=j+2
  Ciklus vége
Eljárás vége.
```

Tudjuk, hogy pontosan Fib(n-1) (1)-gyel, illetve Fib(n-2) (2,1)-gyel, ... kezdődő ilyen permutáció van.

Kombinatorikai algoritmusok – az I. előállítása

Az i-edik ismétléses variáció:

Az i felírása K alapú számrendszerben.

```
Variáció(i,n,k):
  ciklus j=n-től 1-ig -1-esével
    x(j):=i mod k; i:=i div k
  ciklus vége
Eljárás vége.
```

Kombinatorikai algoritmusok – az I. előállítása

Az i-edik részhalmaz:

Az i felírása kettes számrendszerben.

```
Részhalmaz(i,n):
  ciklus j=n-től 1-ig -1-esével
    x(j):=i mod 2; i:=i div 2
  ciklus vége
Eljárás vége.
```

Kombinatorikai algoritmusok – a következő előállítása

Következő permutáció

- $X_1, \ldots X_i, X_{i+1}, \ldots X_n$ rákövetkezője, ha $X_{i+1}, \ldots X_n$ monoton csökkenő és $X_i < X_{i+1}$: $X_1, \ldots X_{i-1}$, a régi X_i-nél nagyobbak közül a legkisebb, majd a többiek monoton növekvően.

Példa: X X X 4 7 5 3 → X X X 5 3 4 7

Kombinatorikai algoritmusok – a következő előállítása

Következő permutáció

```
Következő(P):
  i:=N
  Ciklus amíg i>0 és P(i)<P(i-1)
    S(i):=P(i); i:=i-1
  Ciklus vége
  Ha i>0 akkor …
```

Kombinatorikai algoritmusok – a következő előállítása

…

```
Ha i>0 akkor j:=N
             Ciklus amíg P(j)<P(i)
               j:=j-1
             Ciklus vége
             S(j):=P(i); P(i):=P(j); j:=N
             Ciklus amíg i<N
               i:=i+1; P(i):=S(j); j:=j-1
             Ciklus vége
Eljárás vége.
```

Kombinatorikai algoritmusok – a következő előállítása

Előző permutáció

- $X_1, \ldots X_i, X_{i+1}, \ldots X_n$ rákövetkezője, ha $X_{i+1}, \ldots X_n$ monoton növekvő és $X_i > X_{i+1}$: $X_1, \ldots X_{i-1}$, a régi X_i-nél kisebbek közül a legnagyobb, majd a többiek monoton csökkenően.

Példa: X X X 5 3 4 7 → X X X 4 7 5 3

Kombinatorikai algoritmusok – a következő előállítása

Előző permutáció

```
Következő(P):
  i:=N
  Ciklus amíg i>0 és P(i)>P(i-1)
    S(i):=P(i); i:=i-1
  Ciklus vége
  Ha i>0 akkor …
```

Kombinatorikai algoritmusok – a következő előállítása

```
…
Ha i>0 akkor j:=N
             Ciklus amíg P(j)>P(i)
               j:=j-1
             Ciklus vége
             S(j):=P(i); P(i):=P(j); j:=N
             Ciklus amíg i<N
               i:=i+1; P(i):=S(j); j:=j-1
             Ciklus vége
Eljárás vége.
```

Kombinatorikai algoritmusok

HALADÓ!!!!!

A binomiális együtthatók felhasználhatók számok speciális számrendszerben, az un. *binomiális számrendszer*ben való felírására. Rögzített m esetén minden nemnegatív n szám egyértelműen felírható az alábbi formában:

$$n=\binom{a_1}{1}+\binom{a_2}{2}+\ldots\binom{a_m}{m} \quad \text{, ahol} \quad 0 \le a_1 < a_2 < \ldots < a_m.$$

Kombinatorikai algoritmusok

HALADÓ!!!!!

A Zeckendorf tétel alapján minden természetes szám egyértelműen előállítható Fibonacci számok összegeként úgy, hogy

$$n = F_{k_1} + F_{k_2} + \ldots + F_{k\,r}$$

ahol $\forall i(1 \le i < r): k_i \ge k_{i+1} + 2$, és $k_r \ge 2$. A Fibonacci számokat az alábbi képlettel számolhatjuk:

$$F_k = \begin{cases} 0 & ha \quad k = 0 \\ 1 & ha \quad k = 1 \\ F_{k-1} + F_{k-2} & ha \quad k > 1 \end{cases}$$

Kombinatorikai algoritmusok előadás vége