Belépő a tudás közösségébe

Informatika szakköri segédanyag

Kombinatorikai algoritmusok

Bende Imre, Heizlerné Bakonyi Viktória, Menyhárt László,
Szlávi Péter, Törley Gábor, Zsakó László

Szerkesztő: Abonyi-Tóth Andor, Zsakó László

A kiadvány „A felsőoktatásba bekerülést elősegítő készségfejlesztő és kommunikációs programok megvalósítása, valamint az MTMI szakok népszerűsítése a felsőoktatásban" (EFOP-3.4.4-16-2017-006) című pályázat keretében készült 2018-ban.

SZÉCHENYI 2020

MAGYARORSZÁG KORMÁNYA

Európai Unió
Európai Szociális Alap

BEFEKTETÉS A JÖVŐBE

Eötvös Loránd Tudományegyetem
Informatikai Kar

Áttekintés

A kombinatorika: véges halmazok elmélete, egy véges halmaz elemeinek valamilyen szabály alapján történő csoportosításával, kiválasztásával, sorrendbe rakásával foglalkozik.

Tipikus kombinatorikai feladatok:

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

Ha tudjuk az adott képletet, akkor csak optimálisan ki kell számolni, ha nem tudjuk, akkor meg kell keresni egy rekurzív képletet.

Ismétlés nélküli kombinációk

Ismert képlet:

$$\binom{N}{K}=\frac{N!}{K!*(N-K)!}=\frac{N*(N-1)*\ldots*1}{K*(K-1)*\ldots 1*(N-K)*(N-K-1)*\ldots*1}=\frac{N*(N-1)*\ldots*(K+1)}{(N-K)*(N-K-1)*\ldots>1}$$

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

Rekurzív definíció[1]: N elemből K elem választása kétféleképpen indulhat:

- az első elemet választjuk, majd még N-1 elemből választunk K-1 elemet; vagy
- az első elemet nem választjuk és a maradék N-1 elemből választunk K elemet.

→ B(N,K)=B(N-1,K-1)+B(N-1,K).

```
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.
```

Minta kódok.

C++	cpp/1_Kombinacio_Rekurziv1/feladat.cpp
C#	cs/1_Kombinacio_Rekurziv1/feladat.cs
Java	java/1_Kombinacio_Rekurziv1/feladat.java
Pascal	pas/1_Kombinacio_Rekurziv1/feladat.pas
Python	py/1_Kombinacio_Rekurziv1/feladat.py

Ebben az esetben arra kell figyelni, hogy ugyanazt az értéket ne számoljuk ki kétszer, azaz vagy ciklussal számoljuk ki az értékeket jól megválasztott sorrendben, vagy pedig a rekurzív megoldást memorizálással egészítjük ki.

Táblázatkitöltéssel hatékonyabb:

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

Minta kódok.

C++	cpp/2_Kombinacio_Tablazatkitoltessel1/feladat.cpp
C#	cs/2_Kombinacio_Tablazatkitoltessel1/feladat.cs
Java	java/2_Kombinacio_Tablazatkitoltessel1/feladat.java
Pascal	pas/2_Kombinacio_Tablazatkitoltessel1/feladat.pas
Python	py/2_Kombinacio_Tablazatkitoltessel1/feladat.py

Rekurzív definíció$_2$: N elemből K elem választása így is indulhat:

- 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

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

Minta kódok.

C++	cpp/3_Kombinacio_Rekurziv2/feladat.cpp
C#	cs/3_Kombinacio_Rekurziv2/feladat.cs
Java	java/3_Kombinacio_Rekurziv2/feladat.java
Pascal	pas/3_Kombinacio_Rekurziv2/feladat.pas
Python	py/3_Kombinacio_Rekurziv2/feladat.py

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.
```

Minta kódok.

C++	cpp/4_Kombinacio_Tablazatkitoltessel2/feladat.cpp
C#	cs/4_Kombinacio_Tablazatkitoltessel2/feladat.cs
Java	java/4_Kombinacio_Tablazatkitoltessel2/feladat.java
Pascal	pas/4_Kombinacio_Tablazatkitoltessel2/feladat.pas
Python	py/4_Kombinacio_Tablazatkitoltessel2/feladat.py

Elsőfajú Euler számok

Az Euler-háromszög hasonló a Pascal-háromszöghöz, az ún. Euler számokat tartalmazza: E(n,k) az első n természetes szám azon permutációi száma, ahol pontosan k emelkedés van (emelkedés van az i-edik helyen, ha $x_i<x_{i+1}$).

A legegyszerűbb a Pascal háromszögnél egy addíciós szabály volt, próbálkozzunk itt is ilyennel! Az n elem összes olyan permutációja, ahol pontosan k emelkedés van, kétféleképen állítható elő n-1 elem összes valamilyen permutációjából:

- n-1 elem összes olyan permutációja, ahol pontosan k növekedés van: az n-edik számot a sorozat elejére vagy egy emelkedésbe tesszük – ezzel a növekedések száma nem változik;
- n-1 elem összes olyan permutációja, ahol pontosan k-1 emelkedés van: n-edik elemet a sorozat végére vagy egy nem emelkedő helyre tesszük – ezzel a növekedések száma eggyel nő.

$$E(n,k)=(k+1)E(n-1,k)+(n-k)E(n-1,k-1)$$

A feladat megoldása tesztelhető az elkészült forráskód feltöltésével itt:

Weboldal	https://mester.inf.elte.hu/
Szint	Haladó
Téma	Kombinatorikai algoritmusok
Feladat	27. Ügyességi verseny

Másodfajú Euler számok

A másodfajú Euler számok (E(n,k)) kiszámolási szabálya: 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 emelkedése van.

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

Az elsőfajú Euler számokhoz hasonlóan itt is két esetre bonthatjuk az n-1 elemű permutációkra visszavezetést.

- n-1 elempár összes olyan permutációja, ahol pontosan k növekedés van: az n-edik számpárt a sorozat elejére vagy egy emelkedésbe tesszük (ez azonos az elsőfajú Euler számokkal) ;
- n-1 elempár 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).

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

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

A feladat megoldása tesztelhető az elkészült forráskód feltöltésével itt:

Weboldal	https://mester.inf.elte.hu/
Szint	Haladó
Téma	Kombinatorikai algoritmusok
Feladat	

Ismétlés nélküli permutációk száma megszorítással

Ebben a speciális esetben 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.

Tehát a megoldás egy Fibonacci szám előállítása:

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

A feladat megoldása tesztelhető az elkészült forráskód feltöltésével itt:

Weboldal	https://mester.inf.elte.hu/
Szint	Haladó
Téma	Kombinatorikai algoritmusok
Feladat	

Kombinatorikai algoritmusok – az összes előállítása

Az összes permutáció, kombináció, … előállítása lényegében abból áll, hogy elő kell állítanunk az összes, valamely feltételnek eleget tevő olyan sorozatot, amelyben az 1,2,…,N számok szerepelhetnek. A feltételek teljesülése visszalépéses keresés (backtrack) algoritmussal minden esetben ellenőrizhetők.

Összes ismétlés nélküli permutáció

- Backtrack: $\forall i$ $(1 \le i \le n)$: $\forall j$ $(1 \le j < i)$: $X_j \ne X_i$ – olyan N elemű sorozatok, ahol minden elem különböző.

Összes ismétlés nélküli kombináció

- Backtrack: $\forall i$ $(1 \le i \le k)$: $\forall j$ $(1 \le j < i)$: $X_j < X_i$ – olyan K elemű sorozatok, ahol minden elem különböző és növekvő sorrendben szerepel.

Összes ismétléses kombináció

- Backtrack: $\forall i$ $(1 \le i \le k)$: $\forall j$ $(1 \le j < i)$: $X_j \le X_i$ – olyan N elemű sorozatok, ahol lehetnek egyforma elemek, de növekvő sorrendben szerepelnek.

Összes partíció

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

Összes diszjunkt felbontás

- N felbontása pozitív (>0) számok összegére.

Összes ismétlés nélküli permutáció

```
1 2 3 … N-1 N
1 2 3 … N N-1
…
N 1 2 3 … N-1
…
N N-1 … 3 1 2
N N-1 … 3 2 1
```

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.

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.

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

A feladat megoldása tesztelhető az elkészült forráskód feltöltésével itt:

Weboldal	https://mester.inf.elte.hu/
Szint	Haladó
Téma	Kombinatorikai algoritmusok
Feladat	

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.

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

A feladat megoldása tesztelhető az elkészült forráskód feltöltésével itt:

Weboldal	https://mester.inf.elte.hu/
Szint	Haladó
Téma	Kombinatorikai algoritmusok
Feladat	

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.
```

A feladat megoldása tesztelhető az elkészült forráskód feltöltésével itt:

Weboldal	https://mester.inf.elte.hu/
Szint	Haladó
Téma	Kombinatorikai algoritmusok
Feladat	

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 $i-1 \le X(i) \le i+1$!

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! 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.
```

A feladat megoldása tesztelhető az elkészült forráskód feltöltésével itt:

Weboldal	https://mester.inf.elte.hu/
Szint	Haladó
Téma	Kombinatorikai algoritmusok
Feladat	17. Sorrend-változás

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
…
```

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.
```

A feladat megoldása tesztelhető az elkészült forráskód feltöltésével itt:

Weboldal	https://mester.inf.elte.hu/
Szint	Haladó
Téma	Kombinatorikai algoritmusok
Feladat	

Összes részhalmaz (halmazként)

```
Részhalmazok(n,Y):
  Ciklus i=0-tól 2ⁿ-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)
```

```
Variációk(n,k,Y):
  Ciklus i=0-tól nᵏ-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.
```

A feladat megoldása tesztelhető az elkészült forráskód feltöltésével itt:

Weboldal	https://mester.inf.elte.hu/
Szint	Haladó
Téma	Kombinatorikai algoritmusok
Feladat	

Összes partíció

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

Felbontás(N): {X | N=Σ X_i, ahol X_i>0}, azaz olyan X sorozatok halmaza, amelyek összege éppen N.

A megoldások alakja:

```
1,x,…,x
2,x,…,x
…
N,x,…,x
```

```
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.
```

A feladat megoldása tesztelhető az elkészült forráskód feltöltésével itt:

Weboldal	https://mester.inf.elte.hu/
Szint	Haladó
Téma	Kombinatorikai algoritmusok
Feladat	

Összes partíció

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

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

```
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.
```

A feladat megoldása tesztelhető az elkészült forráskód feltöltésével itt:

Weboldal	https://mester.inf.elte.hu/
Szint	Haladó
Téma	Kombinatorikai algoritmusok
Feladat	

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.
```

A feladat megoldása tesztelhető az elkészült forráskód feltöltésével itt:

Weboldal	https://mester.inf.elte.hu/
Szint	Haladó
Téma	Kombinatorikai algoritmusok
Feladat	

Inverziós táblázat: $a_1,\ldots,a_n \rightarrow b_1,\ldots,b_n$, ahol b_i jelentse az i.-től balra levő, nála nagyobb elemek számát (ez pl. rendezett vektor esetén csupa 0 elemet tartalmazó táblázat lesz, illetve egyetlen, faktoriális számrendszerben felírt szám), ekkor egy b inverziós táblával leírt permutáció előállítása:

```
Permutáció(b):
  sorozat:=[N]
  ciklus i=N-1-től 1-ig -1-esével
    beilleszt(i,b[i]. helyre)
  ciklus vége
Eljárás vége.
```

A feladat megoldása tesztelhető az elkészült forráskód feltöltésével itt:

Weboldal	https://mester.inf.elte.hu/
Szint	Haladó
Téma	Kombinatorikai algoritmusok
Feladat	5. Dekódolás

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

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

```
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.
```

A feladat megoldása tesztelhető az elkészült forráskód feltöltésével itt:

Weboldal	https://mester.inf.elte.hu/
Szint	Haladó
Téma	Kombinatorikai algoritmusok
Feladat	

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.
```

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

Az X_1, …X_i, X_{i+1},… X_n rákövetkezője, ha X_{i+1},… X_n monoton csökkenő és X_i<X_{i+1}: X_1, …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

```
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 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.
```

A feladat megoldása tesztelhető az elkészült forráskód feltöltésével itt:

Weboldal	https://mester.inf.elte.hu/
Szint	Haladó
Téma	Kombinatorikai algoritmusok
Feladat	10. Következő azonosító

Előző permutáció

Az X_1, …X_i, X_{i+1},… X_n rákövetkezője, ha X_{i+1},… X_n monoton növekvő és X_i>X_{i+1}: X_1, …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

```
Előző(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 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.
```

A feladat megoldása tesztelhető az elkészült forráskód feltöltésével itt:

Weboldal	https://mester.inf.elte.hu/
Szint	Haladó
Téma	Kombinatorikai algoritmusok
Feladat	

Következő ismétlés nélküli kombináció

$X_1, \ldots X_i, X_{i+1}, \ldots X_K$ rákövetkezője

$X_1, \ldots X_i, X_{i+1}, \ldots X_K+1$, ha $X_K<N$ vagy

$X_1, \ldots X_i+1, X_i+2, \ldots X_i+K-i+1$, ahol i az utolsó olyan hely, ahol $X_{i+1}-X_i>1$.

Példa: (N=9-re)

X X X X 6 8 ⇒ X X X X 6 9

X X X 5 8 9 ⇒ X X X 6 7 8

A feladat megoldása tesztelhető az elkészült forráskód feltöltésével itt:

Weboldal	https://mester.inf.elte.hu/
Szint	Haladó
Téma	Kombinatorikai algoritmusok
Feladat	34. Csapat összeállítás

Kombinatorikai algoritmusok – egy véletlen előállítása

Véletlen permutáció (keverés)

Ha N elem véletlenszerű sorrendjére van szükségünk, pl. programok tesztelésekor, akkor használhatjuk az alábbi algoritmust:

```
Véletlen permutáció:  {minimumkiválasztásos rendezésre építve}
  Ciklus i=1-től N-1-ig
    j:=véletlen(i..N); Csere(X(i),X(j))
  Ciklus vége
Eljárás vége.
```

Az algoritmus alkalmazása – N*K elemű halmaz K egyenlő részre osztása véletlenszerűen:

```
Véletlen részekre osztás:
  Ciklus i=1-től K*N-1-ig
    j:=véletlen(i..N); Csere(X(i),X(j))
    H((i-1) div K):=H((i-1) div K) ∪ X(i)
  Ciklus vége
Eljárás vége.
```

Véletlen ismétlés nélküli kombináció

Szintén véletlenszerű tesztesetek előállításakor merül fel az a feladat, hogy N lehetséges elemből pontosan K elemet válasszunk ki.

Itt az alapelv, hogy egy elem kiválasztása valószínűsége: hány elem kell még/hányból kell még választani. Belátható, hogy az első elemet K/N valószínűséggel választjuk. A második elemet úgy is választhatjuk, hogy az elsőt is választottuk és úgy is, hogy az elsőt nem. A kettő valószínűségének összege K/N, azaz beláthatjuk, hogy minden elem K/N eséllyel kerül a kiválasztottak közé.

```
Véletlen kombináció(N,K):
  DB:=0
  Ciklus i=1-től N-ig
    Ha véletlenszám<(K-DB)/(N+1-i) akkor DB:=DB+1; Y(DB):=i
  Ciklus vége
Eljárás vége.
```