



# Belépő a tudás közösségébe

## Informatika szakköri segédanyag



## Visszalépéses kiválogatás

**Heizlerné Bakonyi Viktória, Horváth Győző, 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 2017-ben.



Eötvös Loránd Tudományegyetem  
Informatikai Kar

**SZÉCHENYI** 2020



MAGYARORSZÁG  
KORMÁNYA

Európai Unió  
Európai Szociális  
Alap



**BEFEKTETÉS A JÖVŐBE**

## 1. Az összes lehetséges sorrend

Sokszor előfordul feladatként, hogy elő kell állítani valamilyen elemek összes lehetséges sorrendjét. Az egyszerűség kedvéért legyenek az elemek az  $1, \dots, N$  sorszámmok. Ha ugyanis bármi más elemekről lenne szó, akkor azokat helyezzük el egy tömbbe és a sorszámmukat használjuk indexelésre!

Így például 4 elem összes lehetséges sorrendje:

1.: 1 2 3 4	7.: 2 1 3 4	13.: 3 1 2 4	19.: 4 1 2 3
2.: 1 2 4 3	8.: 2 1 4 3	14.: 3 1 4 2	20.: 4 1 3 2
3.: 1 3 2 4	9.: 2 3 1 4	15.: 3 2 1 4	21.: 4 2 1 3
4.: 1 3 4 2	10.: 2 3 4 1	16.: 3 2 4 1	22.: 4 2 3 1
5.: 1 4 2 3	11.: 2 4 1 3	17.: 3 4 1 2	23.: 4 3 1 2
6.: 1 4 3 2	12.: 2 4 3 1	18.: 3 4 2 1	24.: 4 3 2 1

Látszik ezen a számsorozaton, hogy mindegyik tagja 1 és 4 közötti (általános esetben 1 és  $N$  közötti), de minden szám pontosan egyszer fordul elő egy sorozatban. Másképp ránézve: Vannak 1-essel, 2-essel, 3-assal és 4-essel kezdődő típusúak. A megoldás is erre fog építeni, adjuk meg az összes 1-essel kezdődőt, az összes 2-essel kezdődőt, ...

Az alábbi programban  $N$  az elemek száma,  $i$  pedig azon helyek száma, ahova már tettünk elemet, és  $x$ -ben tároljuk az aktuális permutációt.

```
Megoldás (N, Db, Y) :
  Db:=0
  Összes_sorrend(1, N, Db, Y, x)
Eljárás vége.

Összes_sorrend(i, N, Db, Y, x) :
  Ha  $i > N$  akkor  $Db := Db + 1$ ;  $Y(Db) := x$ 
  különben Ciklus  $j = 1$ -től  $N$ -ig
    Ha nem Volt( $i, j, x$ )
      akkor  $x[i] := j$ ; Összes_sorrend( $i + 1, N, Db, Y, x$ )
    Ciklus vége
  Elágazás vége
Eljárás vége.

Volt( $i, j, x$ ) :
   $k := 1$ 
  Ciklus amíg  $k < i$  és  $x[k] \neq j$ 
     $k := k + 1$ 
  Ciklus vége
  Volt:=( $k < i$ )
Függvény vége.
```

Minta kódok

C++ [cpp/1\\_Osszes\\_lehetseges\\_sorrend/feladat.cpp](#)C# [cs/1\\_Osszes\\_lehetseges\\_sorrend/feladat.cs](#)Java [java/1\\_Osszes\\_lehetseges\\_sorrend/feladat.java](#)Pascal [pas/1\\_Osszes\\_lehetseges\\_sorrend/feladat.pas](#)Python [py/1\\_Osszes\\_lehetseges\\_sorrend/feladat.py](#)

## 2. Az összes lehetséges speciális sorrend

Az  $1, \dots, N$  sorozat összes olyan permutációját állítsuk elő, ahol minden elem maximum 1 hellyel mozdul el a helyéről, azaz  $i-1 \leq x[i] \leq i+1$ !

Így például 4 elem összes lehetséges olyan sorrendje, ahol az elemek a sorszámuktól legfeljebb 1 távolságra lehetnek:

```
1.: 1 2 3 4
2.: 1 2 4 3
3.: 1 3 2 4
4.: 2 1 3 4
5.: 2 1 4 3
```

Kiszámolhatjuk egy egyszerű rekurzív függvénnyel, hogy hány ilyen sorozat lehet. Egy ilyen sorrend vagy 1-essel kezdődik és mögötte  $N-1$  szám összes lehetséges jó sorrendje van, vagy 2 1-gyel kezdődik és mögötte  $N-2$  szám összes lehetséges jó sorrendje van.

Azaz az  $N$  szám összes lehetséges jó sorrendje megegyezik  $N-1$  szám összes lehetséges jó sorrendje +  $N-2$  szám összes lehetséges jó sorrendjével: ami ezek szerint egy Fibonacci-szám.

A rekurzív megoldásban tehát  $j$  értéke csak  $i-1$ ,  $i$  vagy  $i+1$  lehet, két kivétellel, az 1-est nem lehet előre hozni, az  $N$  értéket nem lehet hátrább tenni.

```
Összes_jó_sorrend(i,N,Db,Y,x):
  Ha i>N akkor Db:=Db+1; y[Db]:=x
  különben Ciklus j=i-1-től i+1-ig
    Ha j≥1 és j≤N és nem Volt(i,j,x)
      akkor x[i]:=j; Összes_jó_sorrend(i+1,N,Db,Y,x)
  Ciklus vége
  Elágazás vége
Eljárás vége.
```

A feladat megoldása tesztelhető az elkészült forráskód feltöltésével itt:

Weboldal	<a href="https://mester.inf.elte.hu/">https://mester.inf.elte.hu/</a>
Szint	
Téma	
Feladat	

### 3. Szóösszeadó fejtörő

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 szolgáltatson a szavakon.

Pl.  $SEND + MORE = MONEY$ , vagy hagyományosan leírva:

$\begin{array}{r} SEND \\ + MORE \\ \hline MONEY \end{array}$
---

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

1. megoldási ötlet („algebrai hozzáállás”):

$$D+E=Y, N+R=E, \dots M=1$$

$$D+E=10+Y, N+R+1=E, \dots M=1$$

2. megoldási ötlet:

Az összes permutáció algoritmusára építünk. A betűkhöz számjegyeket rendelünk az összes lehetséges sorrendben, majd megnézzük, hogy ez a hozzárendelés jó-e az összegzés szerint.

A JÓ eljárás ellenőrzi a permutáció – a feladat szempontjából való – helyességét, és gondoskodik az esetleges megoldás gyűjtéséről vagy kiírásáról.

A megfelelés a (\*)  $SEND + MORE - MONEY = 0$  egyenletre. Ha

- 'S'  $\times [1]$  értékű, akkor a (\*)-ban  $\times [1] \cdot 1000$ -rel van jelen;
- 'E'  $\times [2]$  értékű, akkor  $\times [2] \cdot [100+1-10] = \times [2] \cdot 91$ -gyel;
- 'N'  $\times [3]$  értékű, akkor  $\times [3] \cdot [10-100] = \times [3] \cdot [-90]$ -nel;
- 'D'  $\times [4]$  értékű, akkor  $\times [4] \cdot [1]$ -gyel;
- 'M'  $\times [5]$  értékű, akkor  $\times [5] \cdot [1000-10000] = \times [5] \cdot [-9000]$ -rel;
- 'O'  $\times [6]$  értékű, akkor  $\times [6] \cdot [100-1000] = \times [6] \cdot [-900]$ -zal;
- 'R'  $\times [7]$  értékű, akkor  $\times [7] \cdot 10$ -zel;
- 'Y'  $\times [8]$  értékű, akkor  $\times [8] \cdot [-1]$ -gyel van jelen.
- továbbá az S és az M betűhöz nem rendelhetünk nullát, azaz  $\times [1] \neq 0$  és  $\times [5] \neq 0$ !

Megoldás (Db, Y) :

Db:=0

Összes\_sorrend(1, 8, Db, Y, x)

**Eljárás vége.**

```
Összes_sorrend(i,N,Db,Y,x):
  Ha i=N+1 akkor Ha Jó(x) akkor Db:=Db+1; Y[db]:=x; Kiírás(x)
    Elágazás vége
  különben Ciklus j=0-tól 9-ig
    Ha nem Volt(i,j,x)
      akkor x[i]:=j; Összes_sorrend(i+1,N,Db,Y,x)
    Ciklus vége
  Elágazás 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.
```

Egy megoldás kiírása a következőképpen nézhet ki:

**Konstans szöveg='SENDMORY'**

```
Kiírás(x):
  Ciklus i=1-től 8-ig
    Ki: szöveg[i],x[i]
  Ciklus vége
Eljárás vége.
```

Szebben kiírva:

SEND	9567
+ MORE	+ 1085
MONEY	10652

Megjegyzés: Hatékonyabb lenne, ha az utolsó feltételt kihasználnánk menet közben: továbbá az S és az M betűhöz nem rendelhetünk nullát, azaz  $x[1] \neq 0$  és  $x[5] \neq 0$ .

```
Összes_sorrend(i,N,Db,Y,x):
  Ha i=N+1 akkor Ha Jó(x) akkor Db:=Db+1; Y[Db]:=x; Kiírás(x)
    Elágazás vége
  különben Ha i=1 vagy i=5 akkor jj:=1 különben jj:=0
    Ciklus j=jj-től 9-ig
      Ha nem Volt(i,j,x)
        akkor x[i]:=j; Összes_sorrend(i+1,N,Db,Y,x)
      Ciklus vége
    Elágazás vége
Eljárás vége.
```

Minta kódok

C++ [cpp/3\\_Szoosszeado\\_fejtoro/feladat.cpp](#)

C# [cs/3\\_Szoosszeado\\_fejtoro/feladat.cs](#)

Java [java/3\\_Szoosszeado\\_fejtoro/feladat.java](#)

Pascal [pas/3\\_Szoosszeado\\_fejtoro/feladat.pas](#)

Python [py/3\\_Szoosszeado\\_fejtoro/feladat.py](#)



Néhány hasonló fejtörő több összeadással:

THIS	SATURN	MARS	HEART
+ ISA	+ URANUS	+ VENUS	+ EARS
+ GREAT	+ NEPTUNE	+ SATURN	+ NOSE
<u>+ TIME</u>	<u>+ PLUTO</u>	<u>+ URANUS</u>	<u>+ THROAT</u>
WASTER	PLANETS	NEPTUNE	HEALTH

És egy fejtörő szorzással:

HE \* HE = SHE

#### 4. N-ből K az összes lehetséges sorrendben

Egy iskolai versenyen  $N$  tanuló vett részt. Meg szeretnénk tudni, hogy az első 3 helyen milyen lehetséges tanuló sorrendek fordulhatnak elő.

A feladat általánosítása: Válasszunk ki  $N$  elemből  $K$  különböző elemet az összes lehetséges sorrendben (ismétlés nélküli variáció)!

A megoldás ugyanolyan, mint  $N$  elem összes lehetséges sorbarendezése, csupán az így kapott sorozatoknak az első  $K$  tagját kell nézni. Ehhez nem is kell előállítani az összes sorrendet, hanem meg lehet állni akkor, amikor már az első  $K$  tagot ismerjük az adott sorozatból.

Így például 4 elemből 2 elem összes lehetséges sorrendje:

```

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

```

Összes_sorrend(i,N,K,Db,Y,x):
  Ha i>K akkor Db:=Db+1; Y[Db]:=x
  különben Ciklus j=1-től N-ig
    Ha nem Volt(i,j,x)
      akkor x[i]:=j; Összes_sorrend(i+1,N,K,Db,Y,x)
    Ciklus vége
  Elágazás vége
Eljárás vége.

```

A feladat megoldása tesztelhető az elkészült forráskód feltöltésével itt:

Weboldal	<a href="https://mester.inf.elte.hu/">https://mester.inf.elte.hu/</a>
Szint	
Téma	
Feladat	

## 5. N-ből K az összes lehetséges növekvő sorrendben

Válasszunk ki  $N$  elemből  $K$  különböző elemet az összes lehetséges növekvő sorrendben (ismétlés nélküli kombináció)!

A megoldás ugyanolyan, mint az  $N$  elemből  $K$  különböző elemet az összes lehetséges sorrendben feladaté, de csak olyan megoldásokat fogadunk el, ahol a kapott sorozat monoton növekvő.

Így például 4 elemből 2 elem összes lehetséges növekvő sorrendje:

```

1.: 1 2
2.: 1 3
3.: 1 4
4.: 2 3
5.: 2 4
6.: 3 4

```

Ha a kapott sorozat csak monoton növekvő lehet, akkor persze az  $i$ -edik tagot elég az  $i-1$ . tagnál nagyobb lehetséges értékekre kipróbálni. Ugyancsak igaz, hogy a mögötte levőknek is kell hagyni értékeket, azaz a  $K$ . érték  $N$ -ig mehet, a  $K-1$ . érték  $N-1$ -ig, a  $K-2$ . érték  $N-2$ -ig, és így tovább.

```

Összes_sorrend(i,N,K,Db,Y,x):
  Ha i>K akkor Db:=Db+1; Y[Db]:=x
  különben Ciklus j=x[i-1]+1-től N-(K-i)-ig
    Ha nem Volt(i,j,x)
      akkor x[i]:=j; Összes_sorrend(i+1,N,K,Db,Y,x)
    Ciklus vége
  Elágazás vége
Eljárás vége.

```

Megjegyzés: Azaz ez a feladat nem is visszalépéses kiválogatás, mert minden úton megoldáshoz jutunk.

Ha megengednénk az ismétlődéseket, akkor az algoritmus – a ciklus határai – egyszerűsödne.

```

Összes_sorrend(i,N,K,Db,Y,x):
  Ha i>K akkor Db:=Db+1; Y[Db]:=x
  különben Ciklus j=x[i-1]-től N-ig
    Ha nem Volt(i,j,x)
      akkor x[i]:=j; Összes_sorrend(i+1,N,K,Db,Y,x)
  Ciklus vége
Elágazás vége
Eljárás vége.

```

A feladat megoldása tesztelhető az elkészült forráskód feltöltésével itt:

Weboldal	<a href="https://mester.inf.elte.hu/">https://mester.inf.elte.hu/</a>
Szint	
Téma	
Feladat	

## 6. Feladatok a sakktáblán

### N bástya elhelyezése egy NxN-es sakktáblán

Helyezzünk el egy NxN-es sakktáblán az összes lehetséges módon N bástyát úgy, hogy ne üssék egymást!

A bástyákról azt kell tudnunk, hogy a sorukban és az oszlopukban álló bábukat üthetik. Tehát úgy kell elhelyezni a bástyákat, hogy minden sorban és minden oszlopban is pontosan 1 bástya legyen!

Ebből következik, hogy a megoldásokban elég azt megmondani, hogy az 1. oszlopban levő bástya melyik sorban áll, a második oszlopban levő bástya melyik sorban áll, ... és így tovább.

Mivel mindegyiknek különböző sorokban kell állnia, ezért a feladat ugyanaz, mint az N elem összes lehetséges sorrendjének előállítás.

Egy lehetséges megoldás N=8-ra:

				B			
			B				
					B		
		B					
						B	
	B						
							B



B							
---	--	--	--	--	--	--	--

```

Összes_bástya(i,N,Db,Y,x):
  Ha i>N akkor Db:=Db+1; Y[Db]:=x
  különben Ciklus j=1-től N-ig
    Ha nem Volt(i,j,x)
      akkor x[i]:=j; Összes_bástya(i+1,N,Db,Y,x)
    Ciklus vége
  Elágazás vége
Eljárás vége.

Volt(i,j,x):
  k:=1
  Ciklus amíg k<i és x[k]≠j
    k:=k+1
  Ciklus vége
  Volt:=(k<i)
Függvény vége.

```

A feladat megoldása tesztelhető az elkészült forráskód feltöltésével itt:

Weboldal	<a href="https://mester.inf.elte.hu/">https://mester.inf.elte.hu/</a>
Szint	
Téma	
Feladat	

## N vezér elhelyezése egy NxN-es sakktáblán

Helyezzünk el egy NxN-es sakktáblán az összes lehetséges módon N vezért úgy, hogy ne üssék egymást!

A vezérekről azt kell tudnunk, hogy a sorukban, az oszlopukban és az átlójukban álló bábukat üthetik. Tehát úgy kell elhelyezni a vezéreket, hogy minden sorban és minden oszlopban is pontosan 1 vezér legyen, és minden átlóban legfeljebb 1 vezér legyen!

Ebből következik, hogy a megoldásokban elég azt megmondani, hogy az 1. oszlopban levő bástya melyik sorban áll, a második oszlopban levő bástya melyik sorban áll, ... és így tovább.

Egy lehetséges megoldás N=5-re és N=4-re:

		V					
						V	
	V						
			V				
V							

	V			
				V
V				
		V		

Itt tehát nem elég azt megnézni egy új vezér elhelyezésekor, hogy volt-e már a  $j$  . sorban vezér, hanem azt is kell, hogy volt-e már az  $(i, j)$  mezővel egy átlóban vezér.

```

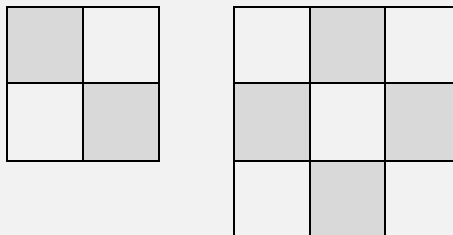
Összes_vezér(i, N, Db, Y, x) :
  Ha  $i > N$  akkor  $Db := Db + 1$ ;  $Y[Db] := x$ 
  különben Ciklus  $j = 1$ -től  $N$ -ig
    Ha nem Volt( $i, j, x$ )
      akkor  $x[i] := j$ ; Összes_vezér( $i + 1, N, Db, Y, x$ )
    Ciklus vége
  Elágazás vége
Eljárás vége.

Volt(i, j, x) :
  k := 1
  Ciklus amíg  $k < i$  és  $x[k] \neq j$  és  $i - k \neq \text{abs}(j - x[k])$ 
    k := k + 1
  Ciklus vége
  Volt := (k < i)
Függvény vége.

```

Megjegyzés: Amíg a bástyás feladatnak mindig van (még hozzá nagyon sok) megoldása, addig a vezéres feladatnak  $N=2$  és  $N=3$  esetén biztosan nincs egy megoldása sem, más  $N$ -ek esetén pedig vannak megoldások, de sokkal kevesebb, mint a bástyás feladatnál.

Példa: az alábbi sakktáblákra nem lehet elhelyezni 2, illetve 3 vezért úgy, hogy ne üssék egymást:



A feladat megoldása tesztelhető az elkészült forráskód feltöltésével itt:

Weboldal	<a href="https://mester.inf.elte.hu/">https://mester.inf.elte.hu/</a>
Szint	
Téma	
Feladat	