# 25. Metódusok 4.

## Lottó



Tegyük fel, hogy elmúltál 18 éves, és lottózni szeretnél. Melyik fajta lottón van a legnagyobb esélyed a telitalálatra egy szelvény kitöltésével?

Az esély annál nagyobb, minél kevesebb módon lehet kitölteni a lottószelvényt. Olyan programot fogunk készíteni, amely meghatározza a lehetséges kitöltések számát lottótípusonként. Ez alapján már könnyen el lehet dönteni az esélyeket.

A lottónál n szám közül kell k darabot kiválasztani úgy, hogy a sorrend nem számít. A matematikusok meghatározták, hogy ekkor a lehetséges kitöltések száma:



Az alábbi táblázat n és k értékét mutatja a különböző lottók esetén. Az 5-ös lottóra ki van töltve. Töltsd ki a többit is!

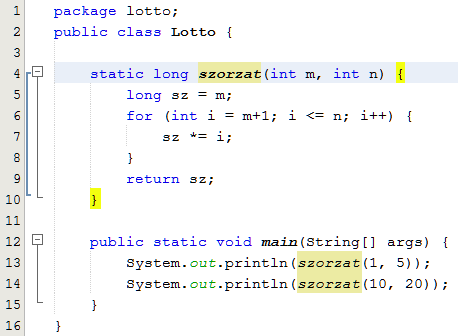
|  |  |  |  |
| --- | --- | --- | --- |
|  | 5-ös lottó | 6-os lottó | Skandináv |
| n | 90 |  |  |
| k | 5 |  |  |

A képletben a számlálóban és a nevezőben is számokat kell összeszorozni. A számlálóban n-k+1 -től n-ig, a nevezőben 1-től k-ig.

Először egy olyan metódust készítünk, amely kap két egész számot (m, n), és előállítja a számok szorzatát m-től n-ig.

Kezdj egy új projektet *lotto* néven!

Először elkészítjük a szorzat metódust, és hozzá egy main metódust a teszteléshez:



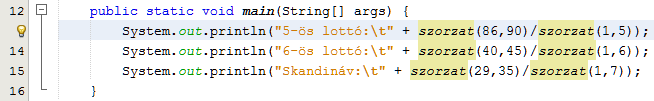
Gondoljuk végig a metódus működését! A szorzat (sz) először m-mel egyenlő. Utána sorban megszorozzuk a számokkal egészen n-ig. Végül a metódus visszaadja a szorzat értékét.

Feltételezzük, hogy m kisebb, mint n.

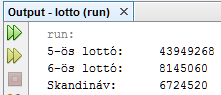
A szorzat egészen nagy szám is lehet, ezért a visszaadott érték long típusú.

Próbáld ki a programot! Láthatod, hogy a második esetben nagyon nagy szám lesz az eredmény. Ha nem 20-ig, hanem 30-ig szorzod össze a számokat, akkor pedig meglepő módon negatív lesz a szorzat. Ennek az az oka, hogy az eredmény már nem fér el egy long típusú változóban. Próbáld ki!

Ezután módosítsd a main metódust a feladatnak megfelelően:



Futtasd a programot! Melyik lottón lenne a legnagyobb esélyed?



Megjegyzés: A skandináv lottónál növeli az esélyt, hogy kétszer húznak. Az esélyek azonban így is nagyon kicsik!

## Rekurzió

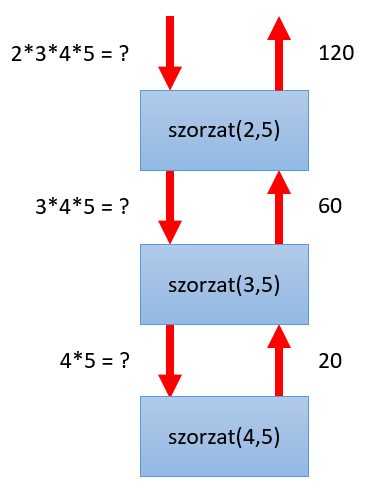
A számok összeszorzását m-től n-ig másképp is elvégezhetjük. Tegyük fel, hogy 2-től 5-ig szeretnénk összeszorozni őket (m=2, n=5)!

A számok szorzata 2-től 5-ig egyenlő 2 \* a számok szorzata 3-tól 5-ig,  
a számok szorzata 3-tól 5-ig egyenlő 3 \* a számok szorzata 4-tól 5-ig,  
A számok szorzata 4-től 5-ig egyenlő 4 \* 5.

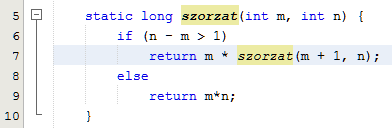
Rövidebben:  
szorzat(2,5) = 2 \* szorzat(3,5) = 2 \* (3 \* szorzat(4,5))

Tehát a szorzat kiszámítását visszavezethetjük egy másik szorzat kiszámítására, azt megint egy másikra és így tovább. Csak arra kell figyelnünk, hogy ha m és n különbsége már csak egy, akkor végezzük el a szorzást.

A számítás során szorzat metódus saját magát hívja meg többször (csak más határokkal), ahogyan a következő ábrán látható: (Ezt nevezik rekurziónak.)



Módosítsuk a szorzat metódust rekurzívra a programban így:



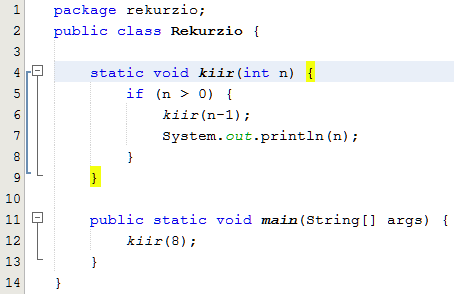
Fontos! A rekurzív metódusnak mindig van legalább egy paramétere, és van benne legalább egy if utasítás. if nélkül a végtelenségig hívná saját magát.

A program többi része nem változik. Próbáld ki a működését! Ellenőrizd, hogy ugyanazokat az eredményeket kapod-e, mint az előző változattal!

## Még több rekurzió

A következő programban kiíratjuk a számokat 1-től 8-ig. Ezt egy ciklussal is könnyen meg tudnánk oldani, de most ciklus nélkül, rekurzióval fogjuk.

Nézzük a programot!



Mit csinál ez a program?

A main metódus csak meghívja a kiir metódust, paraméterként pedig 8-at ad át.

A kiir metódus megnézi, hogy a kapott paraméter nagyobb-e mint nulla. Ha igen, meghívja saját magát 1-gyel kisebb paraméterrel (7, 6, 5, ... ,1, 0), majd kiírja a paraméterként kapott számot. Egyébként nem csinál semmit.

Készítsd el és próbáld ki a programot!

Mi történik, ha felcseréled a 6-os és a 7-es sort (Alt+Shift+nyilak)? Írd ide:

Megjegyzés: Minden feladatot meg lehet oldani rekurzió nélkül (ciklusokkal), de néha egyszerűbb a program a rekurzió alkalmazásával.