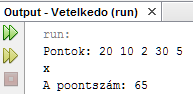
# 08. Tömbök 7.

## Vetélkedő

A mai feladatban beolvassuk egy vetélkedő egy játékosának pontszámait, kivesszük közülük a legkisebbet, majd a többit összeadjuk és kiírjuk. Egy nem szám karakter (pl. x) beírásával jelezzük, hogy nincs több szám.



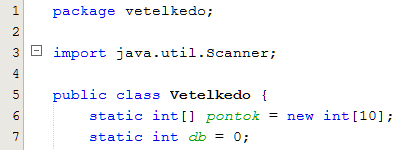
Eközben megismerkedünk az Arrays osztály néhány metódusával.

## Tömb elemeinek beolvasása

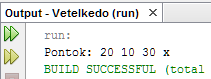
Először be kell olvasnunk a billentyűzetről egy tömb elemeit.

Nem tudjuk előre, hogy hány számot fognak beírni, csak annyit tudunk, hogy legfeljebb 10-et. Ezért egy 10 elemű tömböt deklarálunk, és számoljuk, hány elemet olvastunk be.

Kezdj egy új projektet *vetelkedo* néven! Hozz létre a Vetelkedo osztályban egy pontok nevű, 10 egész számot tartalmazó tömböt, és egy db változót, amelynek kezdőértéke legyen 0!



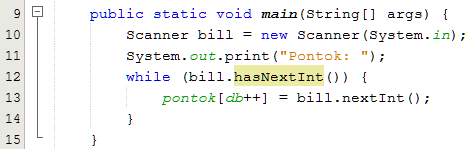
A program csak egyszer kérdezi meg a pontokat, de ezután akárhány pontszámot beírhatunk szóközökkel, tabulátorokkal és sorvégjelekkel (Enterekkel) elválasztva. A számsorozat végét egy olyan karakterrel jelezzük, amely nem értelmezhető számként vagy elválasztójelként. Az alábbi példában ez az x:



Ebben az esetben a pontok tömb 0. elemébe a 20, 1. elemébe a 10, 2. elemébe pedig a 30 kerül majd. A db változó értéke 3 lesz.

A beolvasáshoz szükségünk van a Scanner osztály egy újabb metódusára, a hasNextInt() metódusra. Ez jelzi, hogy a bemeneten a következő karakterek értelmezhetők-e egész számként vagy elválasztójelként. Az Enter megnyomása után vár a további karakterekre.

A beolvasást így valósítjuk meg:



Írd be, és próbáld ki különböző hosszúságú számsorozatokkal! A számokat külön sorokba is írhatod, a sorozat végét jelezd egy tetszőleges betűvel!

## Az Arrays osztály

Az Arrays osztály metódusai a tömbökkel kapcsolatos feladatok megoldását könnyítik meg. Nézd meg először a hivatalos online dokumentációt: <https://docs.oracle.com/javase/8/docs/api/java/util/Arrays.html>

Az alábbi táblázatban kiemeltünk néhány fontos metódust:

|  |  |
| --- | --- |
| metódus | leírás |
| copyOf(t,n) | Másolatot készít a t tömb első n eleméről egy új tömbbe. |
| toString(t) | Stringgé alakítja a t tömböt, amelyet ki lehet íratni. |
| sort(t) | Növekvő sorrendbe rendezi a t tömböt. |

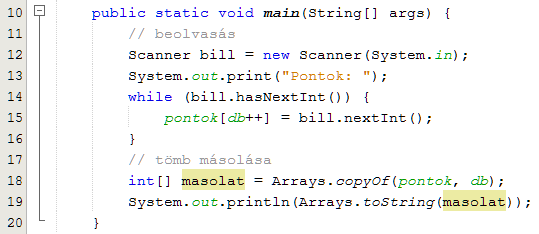
Az egyes metódusoknak többféle változata is van, hogy különböző típusú tömbökkel is lehessen használni őket.

Ezek a metódusok osztálymetódusok, vagyis híváskor az osztály nevét (Arrays) kell eléjük írni.

## Tömb másolása, kiíratása

Programunkban először másolatot készítünk a pontok tömbről, majd ezt kiíratjuk. A masolat tömb mérete akkora lesz, hogy éppen elférjenek benne a beírt pontok. Ezt úgy érjük el, hogy db számú elemet másolunk.

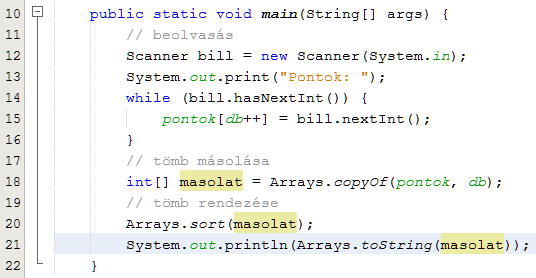
Egészítsd ki a main metódust az alábbiak szerint:



Próbáld ki! Figyeld meg a kiírás formátumát!

## Tömb rendezése

A következő lépés a masolat tömb rendezése. Egészítsd ki a main metódust így, majd próbáld ki!

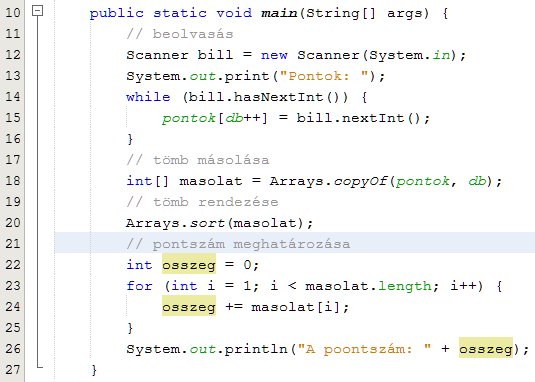


Az Arrays.sort() metódus nagyon gyors rendezési algoritmust valósít meg, ezért lehetőleg ezt használd! (Sokkal gyorsabb, mint a buborékos rendezés ☺)

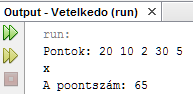
Sajnos a csökkenő sorrendbe rendezést nem lehet ilyen egyszerűen elvégezni, erről majd később lesz szó.

## Feladat befejezése

A feladat szerint a legkisebb kivételével össze kell adni a beolvasott számokat. A rendezés után ez nagyon egyszerű: a második (vagyis 1. indexű) elemtől adjuk össze a tömb elemeit:



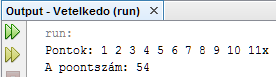
Próbáld ki!



## Önálló feladat

Egészítsd ki a programot úgy, hogy ha betelik a 10 elemű tömb, akkor már ne fogadjon további számokat! (Egyszerűen ne vegye őket figyelembe!)

Próbáld ki!



## Wrapper (csomagoló) osztályok

Már tanultuk, hogy a Java nyelvben vannak primitív és referencia típusok. A primitív típusok egy egyszerű értéket tartalmaznak (szám, karakter vagy logikai érték), a referencia típusok pedig egy objektumnak a címét (pl. string, tömb, …)

Vannak olyan esetek, amikor csak referencia típust lehet alkalmazni, de mi egyszerű adatokat szeretnénk megadni. Például a tömbök csökkenő rendezését csak akkor tudjuk beállítani az Arrays.sort() metódusban, ha a tömb elemei referencia típusúak, vagyis objektumok.

Ilyen esetekben az egyszerű adatokat betehetjük, becsomagolhatjuk egy-egy objektumba (ahogyan az ábrán a sütiket a csomagolásba).



Ha például egész számokból szeretnénk objektumokat csinálni, akkor az Integer osztályt használhatjuk. Az átalakítás mindkét irányban teljesen automatikus (auto-boxing):

Integer n = 23;  
int x = n + 1;

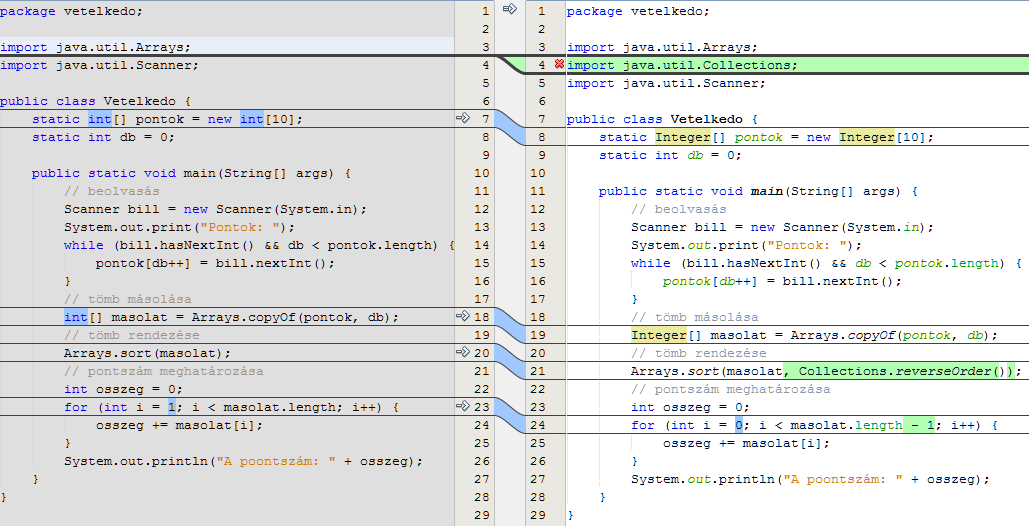
A következő táblázat mutatja, hogy milyen wrapper (csomagoló) osztályokat használhatunk:

|  |  |
| --- | --- |
| **primitív típus** | **osztály** |
| byte | Byte |
| boolean | Boolean |
| char | Character |
| double | Double |
| float | Float |
| int | Integer |
| long | Long |
| short | Short |

## A program új változata

Ez alapján elkészítettem a program egy másik változatát, amely csökkenő sorrendbe rendezi a pontokat.

Az alábbi ábrát a NetBeansben a History gombra kattintva állítottam elő. A bal oldalon az eredeti program, a jobb oldalon az új változat látható:



Figyeld meg a különbségeket:

* Az int típus helyett az Integer típust használtam a tömböknél (8., 19. sor).
* A fordított sorrendet a Collections.reverseOrder() metódussal adtam meg (21.sor), amihez importálni kellett a Collections osztályt (4. sor).
* Mivel a legkisebb pontszám most a tömb végére került, módosítani kellett a határokat a for ciklusban (24. sor).

Írd át te is a programot, és próbáld ki!