# 20. Véletlen 1.

## Fej vagy írás

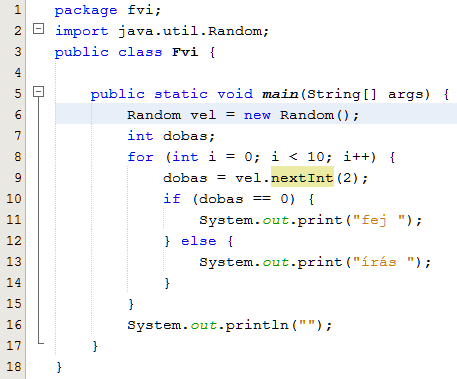
Készítsünk programot, amely egy érme feldobását szimulálja, majd kiírja az eredményt (fej vagy írás)! A program tíz dobást szimuláljon!

A feladat megoldásához meg kell ismerkedned a véletlen számokkal. Ezek a program által, egy algoritmus alapján előállított számok. Megadhatod, milyen értékek lehetnek (pl. 1-től 6-ig), és ebben a tartományban nagyjából egyenletesen oszlanak el.

A véletlen számok előállításához a java.util.Random osztályt fogjuk használni. Hogy ne kelljen kiírni a csomag nevét, a program elején importáljuk az osztályt (2. sor).

A Random osztály használata hasonlít a Scanner osztályéra. Használat előtt létre kell hozni belőle egy példányt (6. sor). Egy véletlen egész szám előállítása pedig a nextInt() metódussal történik, amelynek megadjuk, hogy melyik az a szám, amelynél kisebbnek kell lenni az eredménynek. Az előállított véletlen szám 0 és a megadott határ-1 között lesz. Ha 2-t adunk meg határnak, akkor 0 vagy 1 lesz az eredmény (9.sor).

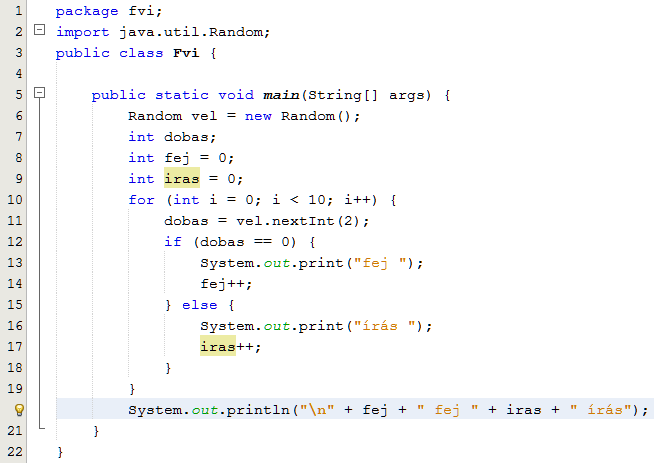
A szimulációt úgy végezzük, hogy a 0 a fejet, az 1 az írást jelenti. Nézzük a program kódját:



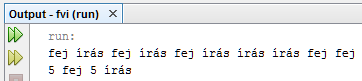
A fej és az írás után csak egy szóközt írunk ki, a 16-os sorban van egy soremelés. Készítsd el a programot (*fvi*), és próbáld ki!

Módosítsuk a programot úgy, hogy számolja a fejek és az írások számát! Szükségünk van két változóra a számláláshoz. Ezeket a program elején nullázzuk, majd az elágazás megfelelő ágában növeljük eggyel. A program végén kiírjuk az eredményt.

Végezd el a módosítást:



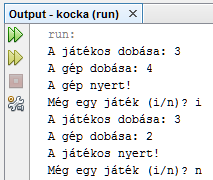
Próbáld ki ezt is!



Mennyire egyenletes az eloszlás? Próbáld meg több dobással (pl. 100) is!

## Kockadobás

Készítsünk programot a következő egyszerű játékhoz! A játékos és a gép is dob egyet-egyet a kockával. Aki nagyobbat dobott, az nyer, egyenlő esetén döntetlen. Utána kérdezzen rá a program az újabb dobásra! Minta:

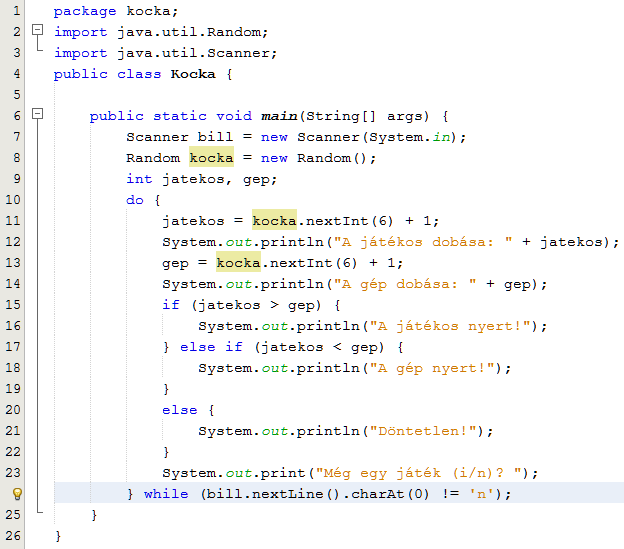


A programban két változónk lesz, a jatekos tárolja a játékos dobását, a gep pedig a gépét. Mindkettő egész típusú.

Egy játék úgy folyik, hogy a program sorsol egy dobást a játékosnak és egyet a gépnek is, majd kiírja az eredményt.

Legalább egy játékot kell játszani, utána választ a játékos, hogy szeretné-e folytatni. Ezért hátul tesztelő ciklust alkalmazunk. A feltételben a beolvasott sor első karakterét hasonlítjuk össze az 'n'-nel. A ciklus addig fut, amíg ezek nem egyenlőek. (Nagy N-nel nem fog leállni.)

Nézzük a programot:



A kockadobásnál 1 és 6 közötti véletlen számot kell előállítani. A kocka.nextInt(6) metódushívás azonban egy 0 és 5 közötti egész számot állít elő. Ezért az eredményhez hozzáadunk 1-et.

Készítsd el és próbáld ki a programot!

Általánosságban, ha nem 0 és egy határ közötti számot kell előállítani, akkor a nextInt() metódus paraméterében azt kell megadni, hogy hányféle számot kell előállítani, és az eredményhez hozzá kell adni a tartomány alsó határát.

int n = vel.nextInt(hányféle) + alsó\_határ

Például ha 10 és 20 közötti számokat szeretnénk generálni, a két határt is beleértve, akkor az utasítás:

int n = vel.nextInt(11) + 10;

Azért 11 féle számot kell előállítani, mert a 10 és a 20 is benne van a tartományban.

Az utolsó sorban a charAt metódus a beolvasott sor első karakterét adja meg.

## Feladat

Írd ide azt az utasítást, amely 20 és 99 közötti véletlen számokat állít elő, a határokat is beleértve!