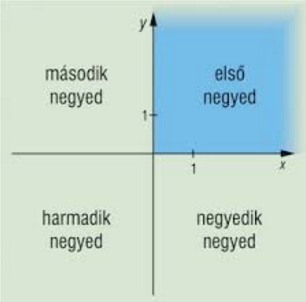
# 11. Elágazás 4.

## Síknegyedek

Az elágazás bármelyik ágában újabb elágazások lehetnek. Ezt fogjuk alkalmazni a következő programban.

A feladat az, hogy olvassuk be egy pont x és y koordinátáit, majd írjuk ki, hogy a pont melyik síknegyedben van. Feltételezzük, hogy egyik koordináta sem nulla. A síknegyedek számozását a következő ábra mutatja:

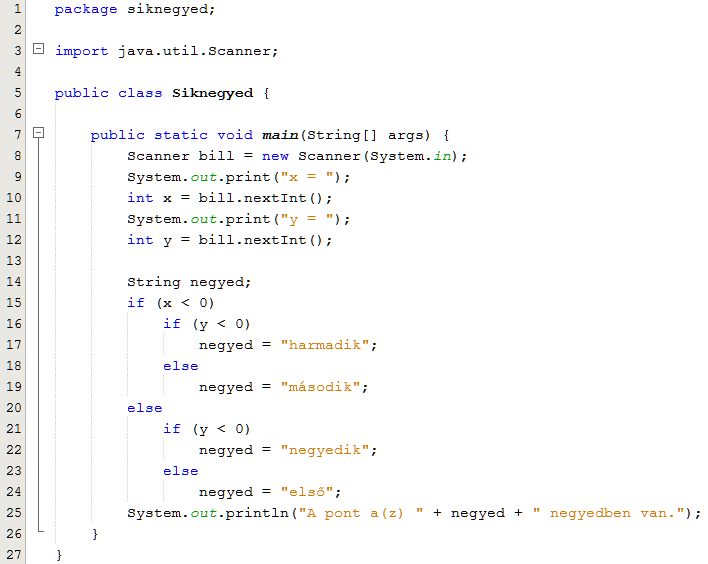


Gondoljuk először végig, hogy határozzuk meg a síknegyedet:

Ha x<0, akkor:  
 Ha y<0, akkor 3. negyed  
 egyébként 2. negyed

Egyébként   
 Ha y<0, akkor 4. negyed  
 egyébként 1. negyed

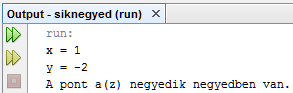
Nézzük a programot:



A beolvasás után deklarálunk egy String típusú (szöveges) változót, ebbe kerül majd az eredmény.

Utána az if utasítás mindkét ágában egy újabb if utasítás van.

Készítsd el a programot (negyedek), majd próbáld végig minden ágát!

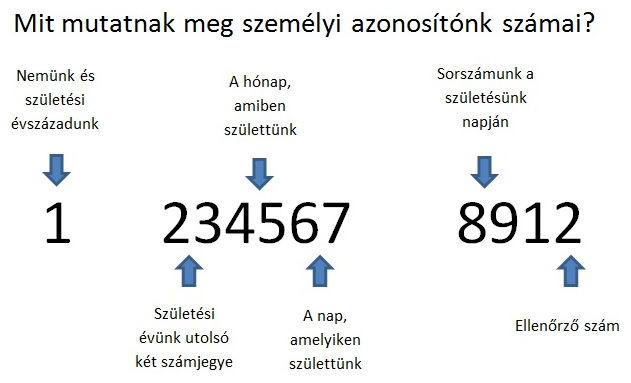


Megjegyzés: Mivel minden ágban csak egy utasítás van, nem használtunk kapcsos zárójeleket. Ha bármelyik ágban több utasítást szeretnénk elhelyezni, természetesen szükségesek a zárójelek.

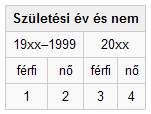
Tipp: A program áttekinthetőségét nagy mértékben javítja a behúzások megfelelő használata. Ha beírás közben egy program a behúzások miatt áttekinthetetlen, automatikusan meg tudod formázni az Alt+Shift+F megnyomásával, vagy a jobb gombos menüből a Format parancsot választva. Ha most kipróbálod a programban, automatikusan elhelyezi a kapcsos zárójeleket is.

## Személyi szám

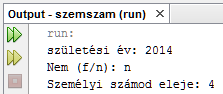
Személyi azonosítónk (személyi számunk) a lakcímkártyánk hátoldalán található.



A személyi szám első számjegye a személy születési idejétől és nemétől függ, ahogy a következő táblázat mutatja:



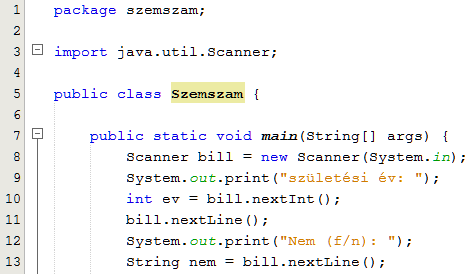
Feladatunk olyan programot készíteni, amely beolvassa egy személy születési évét és nemét, majd kiírja személyi számának első számjegyét! Minta:



Kezdj egy projektet szemszam néven! A program három részből fog állni:

adatok beolvasása  
személyi szám első jegyének meghatározása  
eredmény kiírása

Nézzük a program elejét:

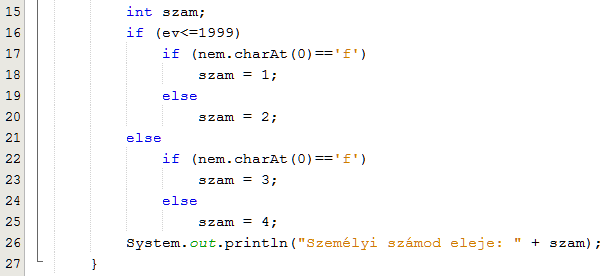


A születési év beolvasásában nincs semmi újdonság, viszont a 11-es sornak látszólag nincs sok értelme. Erre a sorra azért van szükség, mert a Scanner a születési év beolvasása után a sor további részét (az entert, vagyis a sor vége jelet) nem olvassa be. Így a következő nextLine() metódus csak ezt a sorvége jelet olvasná be.

Ezért a következő sor beolvasása előtt egy nextLine() utasítással beolvassuk az előző sorvége jelet, de azt nem tesszük változóba. Így a következő sort már jól fogja beolvasni.

A nem változóba egy Stringet olvastatunk be, de nekünk csak az első karakterére van szükségünk. Ezt így kapjuk meg: nem.charAt(0)

Ennek felhasználásával a program második fele:



Próbáld ki az összes lehetőséget!

## if – else párosítás

Ha egymásba ágyazott if utasítások vannak a programban (mint az előző példákban), akkor az a szabály, hogy minden else az előtte lévő legközelebbi if-hez tartozik, kivéve ha az egy másik blokkban van.

Ez néha problémát okoz. Nézzük a következő példát:

int x = 20;  
if ( x > 0 )  
 if ( x < 10)   
 System.out.println("10-nél kisebb");  
else  
 System.out.println("nem pozitív");

Mit fog kiírni a program? Azt, hogy nem pozitív, pedig ez nem igaz! Miért?

Azért, mert a Java-t nem érdeklik a behúzások, és az else ágat a közelebbi, második if-fel párosította össze, mintha ezt írtuk volna:

int x = 20;  
if ( x > 0 )  
 if ( x < 10)   
 System.out.println("10-nél kisebb");  
 else  
 System.out.println("nem pozitív");

Ha az else ágat az első if-hez szeretnénk párosítani, akkor kapcsos zárójeleket kell alkalmaznunk:

int x = 20;  
if ( x > 0 ) {  
 if ( x < 10)   
 System.out.println("10-nél kisebb");  
}  
else  
 System.out.println("nem pozitív");

Így már a várakozásoknak megfelelően működik a program (és nem ír ki semmit).