# 2. gyakorlat anyaga

## Egységbe zárás: osztály, adattag, metódus, csomag

## Demo

### 1. feladat

Készítsen egy `Point` osztályt `double` típusú `x` és `y` mezőkkel. Írja meg a `move(dx,dy)` műveletet, mellyel egy pontot el lehet tolni `dx` és `dy` koordinátákkal, valamint a `mirror(cx,cy)` műveletet, mely egy `cx` és `cy` koordinátájú pontra való tükrözést valósít meg.

Készítsen `PointMain` néven Java programot, amelyben bemutatja a `Point` osztály használatát.

### 2. feladat

Szervezze a `Point` osztályt és az azt bemutató főprogramot a `point2d` csomagba.

a. A `Point` osztály ne látszódjon ki a csomagból.
b. Készítse el a `pointless.AnotherMain` osztályt. Ez a `pointless` csomag `AnotherMain` osztálya lesz. A megfelelő fájl: `pointless/AnotherMain.java`.
c. Próbálja példányosítani a `Point` osztályt az `AnotherMain` osztályból. Figyelje meg a hibaüzenetet.

## Feladatok

### 1. feladat

Módosítsa a `Point` osztályban a `mirror(p)` műveletet úgy, hogy paramétere (a tükrözési középpont) egy `Point` objektum legyen!

Írjon `distance(p)` műveletet is, mely kiszámolja az adott pont távolságát egy paraméterként kapott `p` ponttól. Használja a `Math.sqrt(...)` függvényt és a Pitagorasz tételt!

Frissítse a `PointMain` osztályt az új műveletekkel!

### 2. feladat

A `Point` osztály váljon láthatóvá más csomagokból is.

a. Az `AnotherMain` most már valóban készítsen `Point` példányokat, két módon.
   i. A teljesen minősített neve használatával.
   ii. Egy `import` segítségével.
b. Készítse el a `PointMainInDefaultPackage` osztály a névtelen csomagban.
   - Ebből használható-e a `Point`?
   - Ennek a tartalma használható-e a `Point` osztályból?

### 3. feladat

Készítsen egy `Distance` programot. Ez a parancssori paramétereket pontoknak értelmezi: a pontok szóközzel elválasztva vannak felsorolva, minden pontnál elöl az `x`, utána az `y` koordináta (ezek is szóközzel elválasztva).

Feltételezhetjük, hogy páros számú paraméter van, amelyek mind egész számok.

A program a `Point` osztály felhasználásával számítsa ki és adja össze az egymás mellett lévő pontok távolságát (pl. 3 pont esetén az 1. és a 2. pont távolságához hozzá kell adni a 2. és a 3. pont távolságát), majd az eredményt írja ki.

Példák:

```
> java Distance
0.0
> java Distance 1 2
0.0
> java Distance 0 0 3 4
5.0
> java Distance 1 2 4 6
5.0
> java Distance 1 2 4 6 7 6
8.0
```

# 4. feladat

Készítsen `plane.PublicCircle` osztályt, amellyel egy síkbeli kört reprezentálunk. Egy körnek van középpontja, `double` típusú `x`, `y` adattagja, amelyeket inicializáljunk 0-ra, illetve sugara (`radius`), amelyet inicializáljunk 1-re. Az adattagok legyenek publikusak benne.

a. Írjon `getArea()` metódust, amely ismét a kör területével térjen vissza.
   i. A `plane.but.not.flying.CircleMain.main` metódus a következőket tegye.
      - Példányosítson egy kör objektumot.
      - Írja ki a területét.
      - Állítsa be a középpontját (5, 2)-re, sugarát 10-re.
      - Írja ki újra a területét.

b. Készítsen az előzőhöz hasonló `plane.Circle` osztályt, de ebben az adattagokhoz csak megfelelően megírt getter és setter metódus férhessen hozzá.
   i. Az utóbbi váltson ki egy `IllegalArgumentException` kivételt, ha a sugara 0 vagy negatív szám lenne.
   ii. A `CircleMain.main` próbálja ki az új funkcionalitást.
      - A kivételt nem kell elkapni, azzal az anyag későbbi részén foglalkozunk.

c. A `Circle` osztályhoz készítsünk konstruktort, az adattagok beállítását ezzel végezzük.
   i. A konstruktor a setterhez hasonlóan válthasson ki kivételt.
   ii. A `CircleMain.main` próbálja ki az új funkcionalitást.

# 5. feladat

Valósítsa meg a `Complex` osztályt `double` típusú valós és képzetes résszel! Írjon `abs()` metódust, amely kiszámolja a komplex szám abszolút értékét. Valósítsa meg az `add(c)`, a `sub(c)` és a `mul(c)` műveleteket oly módon, hogy az `add` adja hozzá a komplex számhoz a paraméterként kapott `c` komplex számot, a `sub` vonja ki belőle, a `mul` pedig szorozza hozzá.

```
alpha.re = 3
alpha.im = 2
beta.re = 1
beta.im = 2
alpha.add(beta)
// alpha.re == 4 && alpha.im == 4 && beta.re == 1 && beta.im == 2
```

# 6. feladat

Készítsen `string.utils.IterLetter` osztályt.

a. Ennek a konstruktora `String` paraméterű.

- Ha ez `null`, váltson ki `IllegalArgumentException` kivételt.

b. Az osztálynak legyen egy `printNext()` metódusa.
   - Ez egy új sorban írja ki a szöveg első karakterét, majd újabb hívásra a második karakterét és így tovább.
   - Ha a sztring összes karakterét kiírtuk a képernyőre, akkor a metódus egy üres sort ír ki minden további hívásra.

c. Készítse el a következő, főprogramot tartalmazó osztályokat: `Main` (a névtelen csomagban), `stringmain.Main`, `string.main.Main`, `string.utils.Main` és `string.utils.main.Main`.
   - Ezek közül egy kivételével mindegyiknek az a feladata, hogy áthívjon egy másik `main`re. Tehát egy négy hosszú hívásláncalakul ki.
     - A sorrend szabadon megválasztható.
     - Tipp: az `X` osztály `y` (osztályszintű) metódusát így lehet meghívni: `X.y(...)`.
       - Hogyan kell most leírni `X` nevét?
     - Hányfajta sorrend létezik?
     - Néhány sorrend érvénytelen. Melyek? Miért?
   - Az utoljára érintett főprogram két parancssori paramétert kap meg. A kód a `printNext` metódust annyiszor hívja meg a második paraméter szövegével, amennyi az első paraméterben kapott szám értéke.
   - Futtassa mindegyik főprogramot. Paraméterként válasszon különféle szövegeket, az üreset is (ez `""` alakban adható át).

d. Az `IterLetter` `reset()` metódusa újrakezdi a szöveg végigjárását.
   - Tehát a soron következő `printNext` hívások az első karaktertől folytatják a kiírásokat.
   - A főprogramok egyszer hívják meg ezt a metódust `m` kiírás után, és még `k` kiírást tegyenek meg. Itt `m` és `k` a harmadik és negyedik parancssori paraméter.

e. Legyen az osztályban egy `hasNext()` metódus is. Ez pontosan akkor tér vissza igaz értékkel, ha van még kiírható betű.
   - A főprogramok hívják ezt meg az újrakezdés előtt és után, valamint közvetlenül a program befejeződése előtt.

# Gyakorló feladatok - egységbe zárás

## 1. gyakorló feladat

Készítsük el a Complex osztályba a `conjugate()` műveletet, mely a komplex számot átalakítja a komplex konjugáltjára. Készítsük el a `reciprocate()` metódust, mely a komplex számot reciprokára alakítja. Definiáljuk a `div(c)` művelet, mely elosztja a komplex számot a paraméterként kapott `c` komplex számmal.

## 2. gyakorló feladat

Készítse el a `Line` osztályt, mellyel egy adott sík egyeneseit reprezentálhatjuk. Egy egyenest az `(ax + by = c)` összefüggés ír le, ahol `a`, `b` és `c` számok `double` típusúak. (Ezek lesznek az osztály adattagjai.)

Írjon az osztályba egy `contains(p)` műveletet, mely eldönti, hogy egy `p` pont rajta van-e az egyenesen!

Írjon egy `isParallelWith(l)` és egy `isOrthogonalTo(l)` metódust, melyek eldöntik, hogy az egyenes párhuzamos-e a paraméterként kapott `l` egyenessel, illetve merőleges-e rá!

## 3. gyakorló feladat

Készítse el a `Segment` osztályt, mely egy szakaszt reprezentál. A szakasz objektumok ábrázolásához a két végpont koordinátáit tároljuk el. Az adattagok `x1`, `y1`, `x2`, `y2` legyenek, mind `double` típusú.

Írjon az osztályba egy `line()` metódust, mely visszaad egy olyan `Line` objektumot, amely a szakaszra illeszkedő egyenest reprezentál.

Írjon az osztályba egy `contains(p)` műveletet, mely eldönti, hogy egy `p` pont rajta van-e a szakaszon!

Készítsen `orientation(p)` metódust a `Segment` osztályba, mely eldönti, hogy a szakasz kezdőpontjából a végpontjába mutató vektor, valamint a szakasz végpontjából a paraméterként kapott `p` pontba mutató vektor milyen orientációjú. A metódus adjon vissza 0-t, ha a `p` rajta van a szakasz által meghatározott egyenesen, adjon vissza pozitív értéket, ha a két vektor az óramutató járásával megegyező irányban van egymással, illetve negatív értéket, ha az óramutató járásával ellenkező irányú. Ez elég egyszerű: ha a `p` pont koordinátáit $x_3$ és $y_3$ jelöli, akkor a metódus az alábbi kifejezést adja vissza.

$(y_2 - y_1)(x_3 - x_2) - (y_3 - y_2)(x_2 - x_1)$

Készítsen egy `intersects(s)` metódust, mely visszaadja, hogy a szakasznak van-e közös pontja a paraméterként kapott `s` szakasszal! A megoldáshoz használja az alábbi segítséget!

http://www.dcs.gla.ac.uk/~pat/52233/slides/Geometry1x1.pdf <Geometry1x1.pdf>

## 4. gyakorló feladat

a

Készítsük el a `Rectangle` osztályt, mely a koordinátatengelyekkel párhuzamos oldalú téglalapok ábrázolására alkalmas! A `Rectangle` objektumukban tároljuk el valamelyik csúcspontjuk `x` és `y` koordinátáját, valamint a téglalap szélességét és magasságát. Ez négy adattagot jelent: `x`, `y`, `width` és `height`. Legyen mindegyik típusa dupla-pontosságú lebegőpontos típus.

A szélesség és a magasság felvehet negatív értéket is. Legyen például az `r` négy adattagja rendre 1, 5, 6, -2. Ekkor az `r` bal alsó csúcsának koordinátái 1 és 3 lesznek.

Definiáljuk a `Rectangle` objektumokon a `topLeft()`, a `topRight()`, a `bottomLeft()` és a `bottomRight()` metódusokat, melyek mindegyike egy `Point` objektumot ad vissza, értelemszerűen a téglalap megfelelő csúcsának a koordinátáit.

A `Rectangle` osztályhoz készítsünk egy főprogramot, mely meghatározza a parancssori argumentumaként kapott téglalapok befoglaló téglalapjának csúcspontjait. A főprogram parancssori argumentumai számok legyenek (legalább 4). Minden számnégyes egy `Rectangle` objektumot határoz meg. Ezeket kell feldolgozni, és a végén kiírni a befoglaló téglalap bal alsó és jobb felső csúcsainak koordinátáit.

```
$ java RectangleMain 3 5 1 -7 2 5 8 8
Bounding rectangle: 2.0;-2.0 - 10.0;13.0
```

A fenti példa két téglalap befoglaló téglalapját határozza meg. Az első téglalap egyik csúcsa (3,5) koordinátájú, szélessége 1, magassága -7. A másik téglalap egyik csúcsa (2,5) koordinátájú, szélessége és magassága egyaránt 8. Az eredményként kapott befoglaló téglalap bal alsó csúcsa (2,-2), jobb felső csúcsa (10,13), amely egyébként a második kapott téglalap jobb felső csúcsa is.

Segítség: a befoglaló téglalap bal alsó csúcsának x-koordinátájának meghatározásához keressük meg a legkisebb értéket a kapott téglalapok bal alsó csúcsának x-koordinátái között, stb.

# Gyakorló feladatok - csomagok

## 1. gyakorló feladat

A `zoo.animal.Panda` osztály egy pandát reprezentál.

a. Az osztály a panda nevének, életkorának és tartózkodási országának eltárolására alkalmas. Ehhez értelemszerű típusú és nevű adattagokat kell használni.
b. Az osztálynak két konstruktora van.
   i. Az első egy újszülött pandát ír le: megkapja és eltárolja a másik két adatát.
   ii. A másik konstruktor a név kivételével kapja meg az adatokat. Ennek a pandának az legyen a neve, hogy `Y years old foundling from C`, ahol `Y` és `C` helyén az életkor és az ország szerepel.
c. Az osztály rendelkezik egy `happyBirthday()` metódussal is. Ez kiírja a panda nevét, országát és az eggyel megnövekedett életkorát is. A metódus egy számot is kap paraméterül (`limitYear`), amennyiben az életkor ezt meghaladja, akkor a pandát visszaköltöztetik a Kínai Népköztársaságba.
d. A `zoo.keeper.Crikey` főprogramja próbálja ki a fentieket.

# 2. gyakorló feladat

Bővítse a `string.utils` csomagot a `string.utils.IterWord` osztállyal.

a. Az osztály konstruáláskor egy szöveget kap meg.
b. Az osztály `printNext()` metódusa új sorban a képernyőre írja a sztring következő szavát.
c. Az osztálynak szintén legyen `restart()` és `hasNext()` metódusa.
d. A már létező főprogramok mellé kerüljön egy-egy `WordMain`, amely bemutatja az osztály használatát.

# 3. gyakorló feladat

Készítse el a `magic.library.Incantation` osztályt, amely egy varázslatos ráolvasást ábrázol. Ez egy `text` szöveges és egy `index` egész szám adatot tárol.

a. Az osztály egyik konstruktora átveszi és beállítja a két paraméter értékét.
   - Ha az előbbi `null`, váltson ki `IllegalArgumentException` kivételt.
b. A másik konstruktor egy `Incantation` példányt kap meg, és ennek az adattagjaiból tölti fel a sajátokat.
c. Legyen mindkét adattagnak gettere, az `index`nek settere is.
d. Az osztály `enchant()` metódusa egy `Incantation` példányt (`otherInc`) kap meg és egy `isPrepend` logikai értéket. A metódus az alábbiak szerint módosítja a hívott példány szövegét.
   i. Először megpróbáljuk felvenni a `text` szöveg `index`edik szavát.
      - Ha például `index` értéke `3`, akkor a harmadik szóról van szó.
      - Tipp: a `String` osztály `split` metódusát érdemes használni.
   ii. Ha nincsen ilyen szó, mert `index` túl magas vagy alacsony (akár negatív is lehet), a metódus hamis visszatérési értéket ad.
   iii. A felvett szót `otherInc` szövege elé írjuk, ha `isPrepend` értéke igaz, és mögé, ha hamis.
      - Egy szóköz is kerüljön a régi szöveg és az új szó közé.
   iv. Az `index` adattag eggyel megnő/lecsökken `isPrepend` értékétől függően.
   v. A metódus igazzal tér vissza, jelezve, hogy a szöveg megváltozott.
e. A `magic.Soliloquy` osztály `reciteIncantations()` metódusát a főprogram fogja hívni, és a következőket teszi.
   i. Két `Incantation` példányt kap meg (`inc1` és `inc2`), valamint egy `idx` egész és egy `startWithAppend` logikai értéket.
   ii. Háromszor hívja meg az `inc1` példányra az `enchant` metódust. Az első paraméter mindegyik esetben `inc2`.
      - A második paraméter az első hívásban `startWithAppend` ellentéte.
      - A másodikban `startWithAppend`.
      - Harmadjára `true`.
   iii. Mindegyik hívás után írja ki a kifejezés visszatérési értékét és mindkét objektum adatait.
      - Ez utóbbiakat a getterek segítségével lehet elérni.
      - A formátumhoz lásd a lenti példát.
      - A kiírás kódja a `printStatus` segédfüggvénybe kerüljön.

iv. Az első kiírás után `inc1` `index` adattagjának értéke álljon `idx`re.

f. A `magic.Soliloquy` osztályban levő főprogram hat parancssori paramétert vár (`argN` lentebb) és a következőket teszi.
   i. Feltehető, hogy a paraméterek száma és tartalma megfelelő, nem kell ellenőrizni.
   ii. A főprogram elkészíti az `inc1`, `inc2` és `inc3` `Incantation` példányokat. Az első kettő tartalmát az első négy parancssori paraméter értékei alapján inicializálja, a harmadikat pedig a másodikból.
      - Az első és harmadik parancssori paramétert idézőjellel (`"`) kell majd körbevenni, mert szóközöket tartalmaznak.
   iii. Ezután a következőket teszi.
      - Meghívja a `reciteIncantations` segédfüggvényt az `inc1, inc2, arg5, true` paraméterekkel.
      - Beállítja `inc1` `index`ét `arg6` értékre.
      - Meghívja a `reciteIncantations` segédfüggvényt az `inc1, inc3, arg5, false` paraméterekkel.

g. Kipróbálás: a főprogram kapja meg a következő paramétereket: `programming is a creative activity`, `4`, `to be or not to be`, `-123`, `1`. Ennek hatására a kiírások legyenek az alábbiak.

```
true;7;programming is a fun and creative activity;-123;to be or not to be creative
true;1;programming is a fun and creative activity;-123;to be or not to be creative
true;0;programming is a fun and creative activity;-123;programming to be or not to be creative
false;0;programming is a fun and creative activity;-123;programming to be or not to be creative

true;3;programming is a fun and creative activity;-123;fun to be or not to be
true;1;programming is a fun and creative activity;-123;fun to be or not to be
true;2;programming is a fun and creative activity;-123;fun to be or not to be programming
true;1;programming is a fun and creative activity;-123;is fun to be or not to be programming
```

# 4. gyakorló feladat

Készítsen egy `game.utils.Vehicle` osztályt, amellyel egy játék járművét reprezentáljuk. Egy járműnek van modelid-je (`int`), rendszáma (`String`), és két színállapota (`color1`, `color2` `int` típusú adatok). A rendszámhoz készítsen setter és getter metódusokat.

Készítsen `game.Player` osztályt, amellyel egy játékost reprezentálunk. Egy játékosnak van neve (`String`), IP-címe (`String`), egészségi állapota (`int`) és lehet járműve (`Vehicle`) (ha nincsen, akkor tároljunk `null` értéket).

A játékos osztályhoz készítsen `print()` metódust, amellyel ki lehet íratni egy játékos legfontosabb információit: nevét, IP-címét, egészségi állapotát, illetve járművének rendszámát (ha van).

Készítsen `game.Main` főprogramot, amelyben példányosít legalább 3 járművet és legalább 2 játékost. Az egyik játékoshoz tartozzon jármű. A főprogram írja ki a képernyőre a játékosok adatait.