# 8. gyakorlat

## Konstruktor-inicializálási lista

Konstruktor törzse előtt adattag beállítása, (ős konstruktor hívása, stb)

Az adattagok mindig az osztályban való deklarálás sorrendjében inicializálódnak.

## Túlterhelés (Overloading)

C++-ban van. (C-ben még nem volt.)

Ugyan abból a nevű függvényből ugyan abban a névtérben lehet több, ha a szignatúrájuk különbözik. (Visszatérési érték nincs figyelembe véve.) Különböző paraméterszám, sorrend más, típusok mások, stb.

## Default paraméter érték

"Hátulról előre." Csak az utolsó $n$ paraméternek lehet default értéke.

## Paramétereket $const int$ referencia formában vesszük át.

Nem fogjuk módosítani az átvett elemet.

## Hibadobás

Mindent el lehet dobni. Pl.:

**throw** **-**1**;**

## Másolás (Copy) konstruktor

MyVector vCopy**(**v**);**

Konstruktor-szerű, copy konstruktor.

A default konstruktor adattagot adattagonként másol.

Ez nem mindig jó, mert ha az egyik vektort törlöm, a másik is törlődik. Ha a másolat hatóköre megszűnik, az eredeti is megszűnik.

Shallow copy: sekély

Deep copy: mély

Saját copy konstruktor célja: új tárterület, elemek mély másolása.

## Értékadás operátor

MyVector vCopy **=** v**;**

Operátorok: speciális alakban hívható függvények.

Privát tagok elérése nem gond. Adott típuson belüli tagfüggvény hozzáférhet reprezentációhoz.

copy-swap idiom

## Mi az a this

Az objektum, amin meghívtuk a függvényt.

v**.**push\_back**(**7**);** == push\_back**(**7**,&**v**);**

A rejtett paraméter: const MyVector**\***const this

Ekkor: \_curr**=**other**.**\_curr**;** == **this.**\_curr**=**other**.**\_curr**;**

Gyakorlat vége.