**Feladat**

Valósítsa meg az egész számokat tartalmazó zsák típust! A zsákot dinamikusan lefoglalt tömb segítségével ábrázolja! Implementálja a szokásos műveleteket (elem betevése, kivétele, üres-e a halmaz, egy elem hányszor van a zsákban), valamint két zsák metszetét (a közös elemek a kisebb előfordulási számmal maradnak meg), továbbá egy zsák kiírását, és végül a másoló konstruktort és az értékadás operátort! Törekedjen a metszetképzés műveletigényének minimalizálására, a dokumentációban mutasson rá a saját megoldásának műveletigényére!

**Zsák típus**

A feladat lényege egy felhasználói típusnak a zsák típusnak a megvalósítása.

**Típusérték-halmaz**

Olyan számokat (ebben az esetben egész számokat: ℤ) tartalmazó zsákkal akarunk dolgozni, amelyekben az különböző értékű elemek számát tartjuk nyilván. Az elemek sorrendje lényegtelen. E zsákoknak a mérete (n𝜖N) lényeges paraméter a megvalósítás szempontjából, de lényegtelen a felhasználó számára.

**Típus-műveletek**

*1. Érték betétele*

Adott érték betétele a zsákba.  
Ha a tomb még üres, az elemet beillesztjük a tömb első helyére, majd számlálóját 1-re váltjuk, ezután a méretet megnöveljük egyel.  
Ha az adott elem már létezik a zsákban, akkor az elem számlálójához adunk csak egyet, ha nem létezik még, akkora dinamikus tomb méretét megnöveljük egyel, majd az új üres helyre illesztjük az elemet, számlálóját pedig 1-re állítjuk.

*2. Érték kivétele*

Adott érték kivétele a zsákban.  
Ha az elem benne van a zsákban, és a számlálója nagyobb mint 1, akkor cask kivonunk a számlálójából 1-et. Ha benne van a zsákban, de a számlálója 1, akkor a tömbben lévő utolsó elemet az aktuális elemre másoljuk, majd a tomb méretét csökkentjük egyel.  
Ha az elem nem szerepel a zsákban, akkor békénhagyjuk.

*3. Üres-e a tomb*

Ha a méret = 0, akkor igazat ad, egyébként hamisat.

*4. Érték darabszáma*

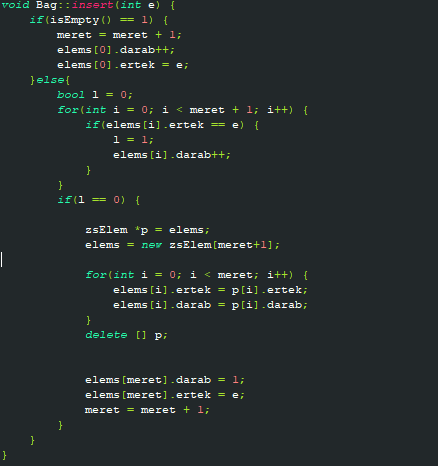
A zsákban az adott érték előfordulása.  
Ha a zsákban szerepel az adott érték, kiírja a darabszámát, egyébként kiírja hogy “Nincs ilyen elem”. Szükség esetén visszatérési értékként az elemszámot visszaadja.

*5. Metszet*

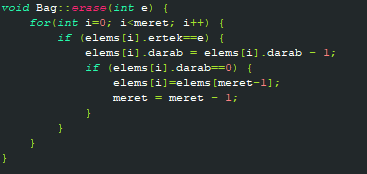
Megadja két zsák metszetét.  
Elindít egy ciklust az egyik zsákra, ebbe ágyazva pedig a másikra. Ha a két zsák valamely eleme megegyezik, beilleszti az adott zsákba ezt az elemet, majd a kisebb darabszámot be.

**Implementáció**

*1. Érték betétele*



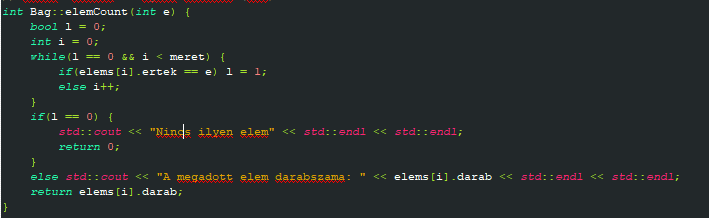
*2. Elem kivétele*



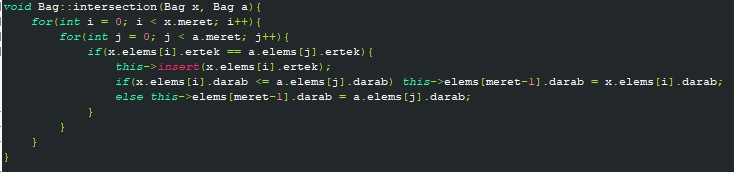
*3. Üres-e a tömb*



*4. Érték darabszáma*



*5. Metszet*

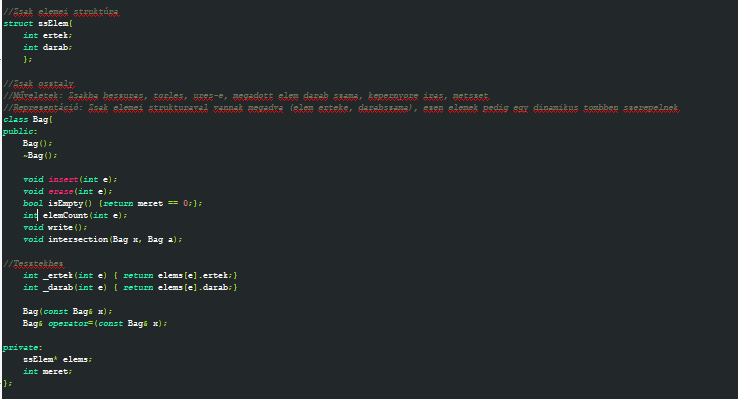


*A metszet műveletigénye*

Legjobb esetben nincs közös érték, ekkor a műveletigény csak a két ciklus, azaz *x.meret \* a.meret*. Legrosszabb esetben az összes érték metszet, ekkor a műveletigény (*x.meret \* a.meret \* 2összehasonlítás) + (x.meret \* a.meret \* 2értékadás).*

**Osztály**

A zsák egy osztály és egy struktúra segítségével jön létre:



Ez a definíció a *Bag.h*-ban van.

**Tesztelési terv**

1. Érték betétele  
 - inp.txt fájlból való betétel : 9 0 3 3 0 0

2. Érték kivétele

- Érték ami többször van bent  
 - Érték ami egyszer van bent  
 - Érték ami nincs bent

3. Üres-e

- Lekérdezés üres zsákon  
 - Lekérdezés nem üres zsákon

4. Érték darabszáma

- Érték ami többször benne van  
 - Érték ami egyszer van benne  
 - Érték ami nincs benne

(ez a 3 egy tesztesetben benne van)

5. Két zsák metszete

- Nincs közös érték  
 - Van közös érték  
 - Egy üres zsák, egy nem üres zsák  
 - Két üres zsák

6. Értékadás  
 - a = b