

# Raktározási probléma

A Tiszanekeresd 2000 Logisztika Kft. az Ipar 4.0 Innovációs Program keretében agilis módon kívánja a jövő technológiáit a szervezeti struktúrájába beépíteni, annak érdekében, hogy továbbra is a megszokott világszínvonalú szolgáltatásait nyújthassa. A feladat a regionális raktárak kihasználtságának fejlesztése. Egy raktárba különböző alakú és méretű raklapok kerülhetnek. A cél minden megadott áru elhelyezése a raktárépületben.

## 1. Bevezetés

Legyen adott egy  $P \in N^{L \times W}$  mátrix, amely a raktárban elhelyezett raklapok pozícióját tartalmazza. Legyen adott továbbá  $V = \{(l_i, w_i)\}_{i=1}^N$ , raklapok halmaza, ahol  $l_i$  az  $i$ . raklap hossza,  $w_i$  pedig az  $i$ . raklap szélessége, illetve egy  $O = \{(j_i, k_i)\}_{i=1}^M$  oszlopok halmaza, melyek a raktár celláinak határánál elhelyezkedő tartóoszlopokat jelölik, ahol  $j$  a hosszanti,  $k$  a szélességi koordináta a  $P$  mátrix bal-felső sarkától indexelve.

Például egy 5x7 méretű raktár és

$$V = \{(4,2), (3,2), (1,2), (2,5), (2,2), (2,1), (3,1)\},$$

$$O = \{(2,2), (3,4)\}$$

raklapok esetén például egy lehetséges allokáció:

$P =$

1	1	1	1	4	4	6
1	1	1	1	4	4	6
2	2	3	3	4	4	7
2	2	5	5	4	4	7
2	2	5	5	4	4	7

Célunk, hogy az összes  $V$ -beli csomagot elhelyezzük a  $P$  mátrixban. A raklapok elforgatása megengedett, sőt, a legtöbb esetben szükséges. Egy tartóoszlop csak a raklapok szélénél és sarkánál helyezkedhet el, nem mehet keresztül azon. Példa egy hibás megoldásra:

1	2	2
1	2	2

A problémáról egyébként belátható, hogy NP-nehéz, ám nagy méret esetén is léteznek hatékony algoritmusok, amelyek megoldást találnak.

## 2. Feladat

Valósítsa meg a raktár feltöltését Java vagy Python nyelven! Az algoritmus tetszőlegesen megválasztható, kódot viszont nem emelhet át külső forrásból.

### 2.1. Java

A megoldás tartalmazzon egy *Main* osztályt, ezen belül pedig egy *main()* függvényt. A bemenetet a standard inputon várja, a kimenetet a standard outputra írja. A program forráskódját *zip* fájlba tömörítve tölts fel a HF portálon (<https://hf.mit.bme.hu>).

### 2.2. Python

A megoldás egyetlen python fájlt tartalmazzon, amely a bemenetet a standard inputon várja, a kimenetet a standard outputra írja. A *zip* fájlba tömörített egyetlen python fájlt tölts fel a HF portálon (<https://hf.mit.bme.hu>).

### 2.3. Bemenet

A bemenet táblázatokkal tagolt szöveges adat, amely első sorában a raktár hosszát és szélességét, második sorában az oszlopok számát, harmadik sorban a raklapok számát, utána megadott számú sorban az oszlopok koordinátáit, majd a többi sorban a raklapok dimenzióit tartalmazza. A fenti esetben:

5	7
2	
7	
2	2
3	4
4	2
3	2
1	2
2	5
2	2
2	1
3	1

## 2.4. Kimenet

Kimenetként írja a **P** mátrixot a standard outputra, tabulátorokkal tagolt formátumban. (Típushiba szokott lenni, hogy a sorok végén felesleges tabulátor van, a kiértékelés során az ilyen megoldásokat nem fogadjuk el).

1	1	1	1	4	4	6
1	1	1	1	4	4	6
2	2	3	3	4	4	7
2	2	5	5	4	4	7
2	2	5	5	4	4	7

## 3. Értékelés

Az értékelés egyre nehezedő tesztesetek alapján, automatikusan történik. Az egyes tesztesetekben csak olyan megoldást fogadunk el, amely az összes raklapot megfelelően elhelyezi! A végső pontszámot a sikeresen teljesített tesztesetek száma adja. Sikertelenség esetén visszajelzést adunk arról, hogy mely esetekben bukott el a program, ekkor természetesen lehet újra próbálkozni.