PSA2: DN4

Evrovizijske tekme je konec, sledi pa ji seveda velika gostija, ki jo bodo priredili v čast zmagovalcu. Na njej se bo veliko jedlo in pilo, nas pa v skrbi za dobrobit udeležencev zanima, koliko katerega vitamina in minerala bodo udeleženci gostije zaužili.

To bomo izračunali iz treh tabel:

1. Na (i, j)-tem mestu v prvi tabeli je zapisano, koliko vitamina/minerala i se nahaja v živilu j.
2. Na (i, j)-tem mestu v drugi tabeli je zapisano, koliko živila i je potrebnega za obrok j.
3. Na (i, j)-tem mestu v tretji tabeli je zapisano, koliko porcij obroka i je na zabavi naročila država j.

Označimo tabele s T_1, T_2 in T_3. Vhodni podatki za dano podnalogo se bodo naložili v `Disk`, s katerim boste operirali. **Pozor: število dostopov do diska (branj in pisanj) je omejeno**, saj se diski radi kvarijo, prav tako pa je **omejen dodaten prostor, ki ga lahko porabimo ob računanju**.

Naloga A (5 točk)

Naredi tabelo, v kateri bomo na (i, j)-tem mestu našli količino vitamina/minerala i v obroku j. Pri tej nalogi potrebujemo torej le prvi dve tabeli.

Naloga B (5 točk)

Naredi tabelo, v kateri bomo na (i, j)-tem mestu našli količino vitamina/minerala i, ki si ga je s svojimi naročili zagotovila država j. Pri tej nalogi potrebujemo vse tri tabele.

Vhodni podatki

Vsaki od podnalog pripada vhodna datoteka (`nalogaA.txt` in `nalogaB.txt`). V prvi vrstici vhodne datoteke se nahaja število tabel $t \in \{2, 3\}$. Sledi t skupin vrstic, od katerih vsaka opisuje tabelo T_i na sledeč način.

V prvi vrstici skupine se nahajata s presledkom ločeni števili m_i in n_i, ki podajata razsežnosti tabele T_i. Velja $T_i \in \mathbb{N}^{m_i \times n_i}$. Sledi $m_i n_i$ vrstic, v katerih podamo tabelo po vrsticah. Vsaka od njih vsebuje en element tabele.

Izhodni podatki

Izpišite iskano tabelo (po vrsticah). To **morate** storiti z uporabo metode `zapisiVDatoteko(String datoteka, int i0, int vrstice, int stolpci)`, kot je to nakazano v predlogi za rešitve.

Omejitve

Tokrat bomo omejili

- količino spomina, ki ga lahko porabimo, in
- število dostopov do diska,

saj je naloga časovno precej trivialna. Spomin je omejen takole:

- Na objekt `disk` lahko spravimo 30 MB. To je približno 1000-krat več spomina, kot ga zasedejo vhodni podatki.
- Da se izognemo zapletom, bomo delovni pomnilnik simulirali kar s spremenljivkami v samem programu. Hkrati je dovoljeno hraniti do $13b^2$ podatkov z diska, kjer je $b = 19$ velikost bloka na disku.[1].

Število dostopov je omejeno na 100 (primer), 10^5 (naloga A) in 10^4 (naloga B), kar je približno dvakrat več, kot jih potrebuje uradna rešitev.

Preverjanje rešitev

Definirajte, kolikor razredov želite, nato pa poženite svojo kodo v priloženem razredu `Main`. Zaradi lažjega preverjanja ne spreminjajte njegove vsebine. Če menite, da potrebujete dve različni rešitvi (npr. `resiA` in `resiB`), ju kličite iz splošne metode `resi`.

Število dostopov do diska in pravilnost odgovora se bosta avtomatsko preverila, porabo delovnega pomnilnika pa bomo izvajalci preverili ročno. Za preverjanje rešitev potrebujete svojo izhodno datoteko in uradne rešitve v datotekah `nalogaA.sol` in `nalogaB.sol`. Končno verzijo kode oddajte na učilnici (samo izvorno kodo).

[1] Uradna rešitev jih hrani do $12b^2 + 3b$.

Primer

Vhod:

```
2
4 3
15
4
1
3
19
2
3
4
12
8
1
1
3 2
1
12
20
16
20
15
```

Izhod:

```
1
4 2
115
259
423
370
323
280
48
127
```

Primer opisuje podatke

$$T_1 = \begin{bmatrix} 15 & 4 & 1 \\ 3 & 19 & 2 \\ 3 & 4 & 12 \\ 8 & 1 & 1 \end{bmatrix}, \quad T_2 = \begin{bmatrix} 1 & 12 \\ 20 & 16 \\ 20 & 15 \end{bmatrix} \quad \text{in izhod} \quad \begin{bmatrix} 115 & 259 \\ 423 & 370 \\ 323 & 280 \\ 48 & 127 \end{bmatrix}.$$