Programovanie (1) v C/C++ 2024/25
Tréningový príklad na skúšku 1a
Úradník

Toto je cvičný príklad pre prvý príklad na skúške. Predtým ako začnete programovať, si poriadne rozmyslite, aké dátové štruktúry (polia, matice, struct-y a pod.) chcete v programe použiť.

Úradník má v kancelárii dlhý stôl, na ktorý sa vedľa seba zmestí P kôpok úradných spisov. Stôl si teda predstavíme rozdelený na políčka očíslované zľava doprava 0 až $P-1$. Na každom políčku môže byť kôpka niekoľkých spisov, alebo môže byť prázdne. Celkovo je na stole S spisov očíslovaných 0 až $S-1$. Ráno má úradník všetky spisy na políčku 0 usporiadané podľa čísla tak, že spis 0 je na vrchu kopy a spis $S-1$ na spodku.

Ako úradník pracuje so spismi, môže zobrať horných k spisov z niektorého políčka a presunúť ich na iné políčko. Ak na tomto políčku už nejaké spisy boli, nové spisy položí na ne. Vašou úlohou je naprogramovať simuláciu tohto procesu.

Na vstupe sú celé kladné čísla P (počet políčok) a S (počet spisov). Každý z ďalších riadkov vstupu začína jedným z písmen `p`, `v` alebo `k`. Písmeno `p` znamená presun spisov medzi políčkami. Za písmenom nasledujú celé čísla a, b, c. pričom a je číslo políčka, z ktorého sa budú presúvať spisy, b je číslo políčka, na ktoré sa budú presúvať spisy a c je počet presunutých spisov. Môžete predpokladať, že a a b sú navzájom rôzne čísla z rozsahu 0 až $P-1$ a c je kladné číslo, ktoré neprevyšuje počet spisov aktuálne umiestnených na políčku a. Po vykonaní presunu spisov vypíšte, koľko je na stole kôpok, teda políčok s nenulovým počtom spisov.

Riadok začínajúci písmenom `v` znamená výpis. Na riadku je zadané aj poradové číslo políčka, pričom na konzolu treba vypísať spisy ležiace na tomto políčku v poradí od spodného po vrchný. Ak na políčku nie je žiaden spis, vypíšte namiesto zoznamu spisov pomlčku. Napokon riadok začínajúci písmenom `k` znamená koniec vstupu. Pri výpise dodržte formát z príkladu nižšie. Na konci riadkov nevypisujte medzery.

V programe je zakázané používať polia konštantných veľkostí. Polia alokujte dynamicky (príkazom `new`) alebo použite štruktúry, ktoré menia veľkosť podľa potreby (`vector`, `string` a podobne). Dynamicky alokovanú pamäť odalokujte.

Príklad vstupu:

```
5 6
v 0
p 0 3 3
v 0
v 1
v 3
p 0 3 1
v 0
v 3
p 3 1 4
p 1 2 2
v 3
v 1
v 2
p 2 0 2
p 1 0 2
v 0
v 1
v 2
v 3
v 4
k
```

Príklad výstupu:

```
policko 0: 5 4 3 2 1 0
pocet kopok: 2
policko 0: 5 4 3
policko 1: -
policko 3: 2 1 0
pocet kopok: 2
policko 0: 5 4
policko 3: 2 1 0 3
pocet kopok: 2
pocet kopok: 3
policko 3: -
policko 1: 2 1
policko 2: 0 3
pocet kopok: 2
pocet kopok: 1
policko 0: 5 4 0 3 2 1
policko 1: -
policko 2: -
policko 3: -
policko 4: -
```