Programiranje 2 — izpit 2
21. junij 2024

Vsa števila, ki nastopajo v besedilu in testnih primerih, so cela.

Razdelitev točk po nalogah: 30, 35, 35.

Oddajte datoteke `naloga1.c`, `naloga2.c` in `naloga3.c`.

(1) V prvi vrstici vhodne datoteke sta podani števili $n \in [3, 9]$ in $p \in [1, 100]$, nato pa sledi p vrstic, od katerih vsaka vsebuje zaporedje najmanj dveh in največ 100 nizov dolžine 2, ločenih s po enim presledkom. Vsak niz predstavlja eno od polj na šahovnici velikosti $n \times n$. Prvi znak niza predstavlja oznako stolpca (`a` je prvi stolpec, `b` drugi itd.), drugi znak pa zaporedno številko vrstice (ta se začne z 1).

Če skakač skoči s polja (v, s) (vrstica, stolpec) na polje (v', s'), bomo rekli, da je pripadajoči *vektor premika* enak $(v' - v, s' - s)$. Kot vemo, je $(\Delta v, \Delta s)$ veljaven vektor premika natanko tedaj, ko je $|\Delta v| \cdot |\Delta s| = 2$.

Napišite program, ki v podano izhodno datoteko izpiše p vrstic. Če i-to zaporedje nizov v vhodni datoteki predstavlja veljaven skakačev sprehod po šahovnici in če vsak od osmih možnih vektorjev premika v tem sprehodu nastopa najmanj po enkrat, naj bo v i-ti vrstici izhodne datoteke zapisano število 1, sicer pa število 0. Imeni vhodne in izhodne datoteke sta podani kot argumenta ukazne vrstice.

Sledeči primer prikazuje vsebino datoteke `vhod01.txt` in datoteke `rezultat01.txt`, ki naj bi jo program ustvaril po izvedbi naslednjih ukazov:

```
gcc -o naloga1 naloga1.c
./naloga1 vhod01.txt rezultat01.txt
```

`vhod01.txt`:

```
5 4
c3 e4 d2 b1 a3 b5 d4 e2 c3
a1 c2 a3 b5 c3 b1 a3 c2 d4 b5
c3 e4 c3 e4 c3 e4 d2 b1 a3 b5 a3 b5 d4 b5 d4 b5 d4 e2 c3 b1
c3 e4 d2 b1 a3 b5 d4 e2 c3 b4 c2
```

`rezultat01.txt`:

```
1
0
1
0
```

V drugem zaporedju nastopajo vsi vektorji premika razen $(-2, -1)$. Četrto zaporedje ne predstavlja veljavnega skakačevega sprehoda.

V 50% testnih primerov je število nizov v vsaki vrstici vhodne datoteke enako natanko 42.

(2) V datoteki `naloga2.h` je podana definicija strukture `Vozlisce`, ki predstavlja vozlišče povezanega seznama. V datoteki `naloga2.c` dopolnite funkcijo

`void obdelaj(Vozlisce* zacetek)`,

tako da bo za vsako skupino najmanj dveh zaporednih vozlišč z isto vrednostjo komponente `podatek` storila dvoje: (1) vrednost komponente `podatek` v prvem vozlišču iz skupine naj spremeni v vsoto vrednosti komponent `podatek` vseh vozlišč iz skupine; (2) odstrani naj vsa vozlišča iz skupine razen prvega.

Prvo vozlišče povezanega seznama se nahaja na naslovu `zacetek`.

Na primer, če funkcija prejme seznam s podatki 3, 3, 3, 3, 5, 2, 2, 2, 7, 7, 9, 8, 8, 8 (`test01.*`), naj enkrat podčrtanim vozliščem spremeni komponento `podatek` v 12 ($= 3+3+3+3$), 6, 14 in 24 (v tem vrstnem redu), dvakrat podčrtana vozlišča pa naj odstrani. Ostala vozlišča naj pusti pri miru.

Število vozlišč seznama (n) pripada intervalu $[1, 2000]$. V 20% testnih primerov si komponente `podatek` v seznamu sledijo v zaporedju a_1, a_1, a_2, a_2, a_3, a_3, ..., kjer je $a_i \neq a_{i+1}$ za vsak i. V nadaljnjih 20% testnih primerov se nikoli ne zgodi, da bi več kot dve zaporedni vozlišči imeli isto vrednost komponente `podatek`.

(3) Napišite program, ki iz dvojiške datoteke prebere opis dvojiškega drevesa z n vozlišči, označenimi z zaporednimi številkami $1, 2, \ldots, n$, in po vrsti (od leve proti desni) izpiše zaporedne številke vozlišč na nivoju k. Vozlišče 1 je koren drevesa (edino vozlišče na nivoju 0). Datoteka vsebuje $2n$ bajtov, pri čemer bajta z indeksoma $2i-2$ in $2i-1$ (za $i = 1, \ldots, n$) podajata zaporedni številki otrok vozlišča i. Če kateri od otrok manjka, je pripadajoči bajt enak 0.

Na standardnem vhodu je podano ime datoteke, dolgo največ 20 znakov, ter števili $n \in [1, 255]$ in $k \in [0, h]$ (h je višina drevesa). Program naj iskane zaporedne številke izpiše na standardni izhod (vsako v svojo vrstico).

Sledi primer vsebine standardnega vhoda, dvojiške datoteke in standardnega izhoda, na desni pa je prikazano pripadajoče drevo.

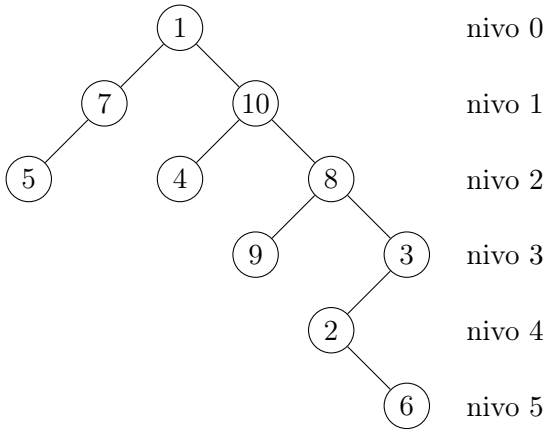
`test01.in`:

```
vhod01.bin 10 2
```

`vhod01.bin`:

0	1	2	3	4	5	6	7	8	9
7	10	0	6	2	0	0	0	0	0

10	11	12	13	14	15	16	17	18	19
0	0	5	0	9	3	0	0	4	8



`test01.out`:

```
5
4
8
```

V 20% testnih primerov velja $k = 1$.