Registrų paskirstymas

Šiuolaikiniai procesoriai turi ribotą kiekį *registrų* — bendros paskirties atmintinės, kuri yra daug spartesnė nei pagrindinė atmintinė. Skaičiavimo operacijos (pavyzdžiui, sudėtis, daugyba) savo argumentų reikšmes laiko registruose, taip pat ir rezultatus įrašo į registrus.

Šioje užduotyje nagrinėsime *registrų paskirstymą* skaičiuojant reiškinius. Kompiliatoriuje reiškiniai vaizduojami medžiu. Medžio lapai (galinės viršūnės) vaizduoja reikšmes, kurios turi būti įkeliamos iš pagrindinės atmintinės. Tarpinės (ne lapai) medžio viršūnės vaizduoja operacijas ir kiekviena jų turi tiek antrinių šakų, kiek operacija argumentų. Suprantama, visų argumentų reikšmės turi būti žinomos prieš atliekant operaciją.

Kadangi registrų yra ribotas skaičius, tai kompiliatorius turi nuspręsti, kurie tarpiniai duomenys bus laikomi registruose (jais galima pasinaudoti betarpiškai, kai tik jų prireikia), o kurie - pagrindinėje atmintinėje (jie turi būti įkeliami į registrus, kai jų prireikia). Gali būti naudinga pakeisti operacijos argumentų skaičiavimo tvarką (štai dėl ko aukšto lygio programavimo kalbos nereglamentuoja jokios bendros reiškinių skaičiavimo tvarkos).

Jums reikia parašyti programą, kuri pagal duotą reiškinio medį paskirstytų registrus ir rastų tokią skaičiavimų tvarką, kad bendros sąnaudos būtų minimalios.

Pradiniai duomenys. Pirmoje bylos `REGS.IN` eilutėje įrašytas registrų skaičius N ($1 \leq N \leq 100$). Antroje eilutėje yra du sveikieji skaičiai: reikšmės įkėlimo iš pagrindinės atmintinės į registrą sąnaudos C_l ($1 \leq C_l \leq 100$) ir reikšmės įrašymo iš registro į pagrindinę atmintinę sąnaudos C_s ($1 \leq C_s \leq 100$).

Likusios eilutės aprašo reiškinių medį pradedant šaknine viršūne:

- pirmoje eilutėje yra iš viršūnės išeinančių antrinių šakų skaičius K_x ($0 \leq K_x \leq 10$; $K_x \leq N$);
- jei $K_x = 0$, tai ši viršūnė yra lapas ir aprašymas baigtas;
- jei $K_x > 0$, tai ši viršūnė yra tarpinė ir:
  - kitoje eilutėje pateikiamas vienas sveikasis skaičius: viršūnę atitinkančios operacijos sąnaudos C_x ($1 \leq C_x \leq 100$);
  - toliau pagal tą pačią schemą aprašoma K_x pomedžių;

Laikoma, kad viršūnės sunumeruotos nuo 1 iki M ta tvarka, kuria jos pateiktos pradinėje byloje. Galioja ribojimas: $M \leq 10\,000$.

Rezultatai. Pirmoje rezultatų bylos `REGS.OUT` eilutėje turi būti įrašytas vienas skaičius - minimalios sąnaudos, reikalingos reiškiniui apskaičiuoti. Toliau kiekvienai tarpinei medžio viršūnei skiriama po vieną eilutę. Kiekvienoje eilutėje turi būti įrašoma po du sveikuosius skaičius: pirmasis skaičius nurodo numerį viršūnės, kurios reikšmė skaičiuojama, antrasis turi būti lygus 1, jei rezultatas bus laikomas registre, arba 0, jei rezultatas turi būti laikomas pagrindinėje atmintinėje (nepamirškite sąnaudų C_s prisumuoti prie bendrų reiškinio skaičiavimo sąnaudų).

Operacijos išvardijamos ta tvarka, kuria jos turi būti vykdomos. Bendros reiškinio skaičiavimo sąnaudos būtų mažiausios, būtina laikytis šių sąlygų:

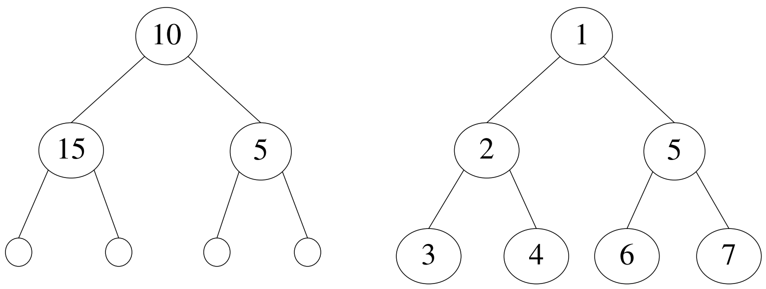
- viršūnės reikšmę galima skaičiuoti tik tada, kai jau suskaičiuotos visų jos antrinių viršūnių reikšmės,
- visi vykdomos operacijos argumentai, kurių nėra registruose, turi būti įkelti į registrus iš pagrindinės atmintinės (vieno argumento įkėlimo sąnaudos lygios C_l);
- registrai, kuriuose buvo saugomi operacijos argumentai atlikus operaciją gali būti iš karto vėl panaudoti (ypač tos pačios operacijos rezultatui išsaugoti).

Jeigu yra keletas sprendinių su minimaliomis sąnaudomis, pateikite vieną.

Pavyzdys.

```
REGS.IN     REGS.OUT
2           47
3 2         2 0
2           5 1
10          1 1
2
15
0
0
2
5
0
0
```

Pastaba. Paveiksle iliustruojamas aukščiau pateiktas pavyzdys: abu medžiai vaizduoja reiškinių medį. Kairysis medis pateikia tarpinių viršūnių (t.y. jas atitinkančių operacijų) sąnaudas, o dešinysis - viršūnių numeravimą.



Rezultatuose pateikto reiškinio apskaičiavimo plano sąnaudos lygios

$$(C_l + C_l + 15 + C_s) + (C_l + C_l + 5) + (C_l + 10) = 47.$$

Pastaba. Jums pateikta programa `REGSCHECK`, kuri tikrina jūsų pateikto rezultato korektiškumą (bet ne optimalumą) ir pateikia informatyvius pranešimus apie klaidas.