Registrid

Modernsetel arvutiprotsessoritel on piiratud arv *registreid* — üldotstarbelisi mälupesasid, mis on märkimisväärselt kiiremad kui põhimälu. Arvutusi teostavad operatsioonid (nt. liitmine, korrutamine jne) saavad argumendid ette registritest ja väljastavad tulemuse samuti registrisse.

Selles ülesandes vaatleme *registrite hõivamise* probleemi avaldiste väärtustamisel. Kompilaatori sees esitatakse avaldis puuna. Puu lehttipud vastavad väärtustele, mida on vaja lugeda põhimälust. Vahetipud (tipud, mis pole lehttipud) vastavad tehetele, kusjuures igal vahetipul on nii palju alluvaid, kui palju on tehtel argumente. Loomulikult peavad olema operatsiooni teostamiseks kõikide tema argumentide väärtused registrites.

Kuna registreid on ainult piiratud arv, peab kompilaator otsustama, milliseid vahetulemusi hoida registrites (neid saab arvutustes kasutada kohe, kui neid vaja on) ja milliseid hoida põhimälus (need peab enne kasutamist mälust registritesse lugema). Samuti võib osutuda kasulikuks muuta mingi operatsiooni argumentide väärtustamise järjekorda (sellepärast ei garanteeri enamik kõrgkeeli kindlat järjekorda).

Sinu ülesandeks on kirjutada programm, millele antakse ette avaldisepuu ning mis leiab registrite hõivamise kava ja väärtustamise järjekorra, mille korral on ajaline koguhind vähim.

Sisend. Sisendfaili `REGS.IN` esimene rida sisaldab registrite arvu N ($1 \le N \le 100$). Teine rida sisaldab kahte täisarvu: väärtuse põhimälust registrisse lugemise hinda C_l ($1 \le C_l \le 100$) ja väärtuse registrist põhimälusse salvestamise hinda C_s ($1 \le C_s \le 100$). Ülejäänud fail kirjeldab avaldisepuud, alustades juurtipust:

- esimene rida sisaldab tipu alluvate arvu K_x ($0 \le K_x \le 10$ ja $K_x \le N$);
- kui $K_x = 0$, siis on tegemist lehttipuga ning kirjeldus on lõppenud;
- kui $K_x > 0$, siis on tegemist vahetipuga ning:
  - järgmine rida sisaldab ühte täisarvu: sellele tipule vastava operatsiooni hind C_x ($1 \le C_x \le 100$);
  - sellele järgnevad sama skeemi järgi K_x alampuu kirjeldust.

Tipud on nummerdatud arvudega $1, 2, \ldots, M$ sellises järjekorras nagu nad esinevad sisendfailis. Võib eeldada, et $M \le 10\,000$.

Väljund. Väljundfaili `REGS.OUT` esimene rida peab sisaldama avaldise arvutamise vähimat võimalikku hinda. Ülejäänud fail peab sisaldama ühte rida avaldisepuu iga vahetipu kohta. Igal real peab olema kaks täisarvu: esimene on väärtustatava tipu number, teine on 1, kui tulemus tuleb jätta registrisse, või 0, kui tulemus tuleb salvestada põhimällu (põhimällu salvestamisel lisandub väärtustamise koguhinnale veel C_s).

Operatsioonid peavad väljundfailis olema selles järjekorras nagu neid tuleb sooritada, et tagada avaldise väärtustamise koguhinna minimaalsus, kusjuures täidetud peavad olema järgmised tingimused:

- tippu saab väärtustada alles pärast seda, kui kõik tema alluvad on väärtustatud;
- kõik täidetava operatsiooni argumendid, mis pole registrites, tuleb lugeda põhimälust registritesse (hinnaga C_l argumendi kohta);

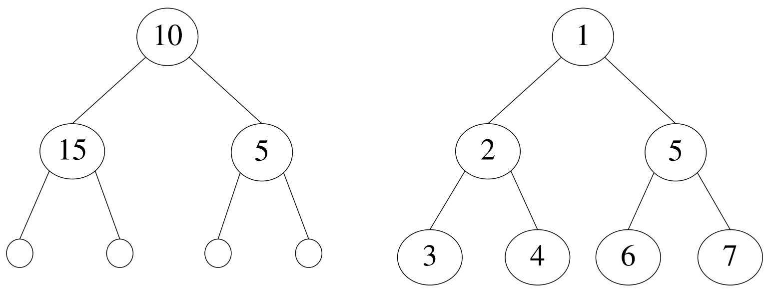
- registrid, mis sisaldavad täidetud operatsiooni argumente, saab kohe uuesti kasutusele võtta, isegi operatsiooni tulemuse säilitamiseks.

Kui on mitmeid võimalusi minimaalse hinna saavutamiseks, siis väljastada suvaline neist.

Näide.

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

Märkus. Järgnev joonis illustreerib ülaltoodud sisendfaili: mõlemad puud vastavad avaldisepuule, kusjuures vasakpoolne puu näitab vahetippude hindu ning parempoolne puu näitab tippude numbreid.



Ülaltoodud väljundfailis kirjeldatud väärtustamise plaani hind on

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

Märkus. Sulle antakse programm `REGSCHECK`, mis kontrollib faili `REGS.IN` alusel faili `REGS.OUT` korrektsust (aga mitte optimaalsust) ja annab informatiivseid veateateid.