1. HTML-loetelu (algajad)

Tekitame tasemeloenduri algväärtusega 0. Edasi töötleme sisendit reakaupa. Kui rida on `<UL>`, siis suurendame tasemeloendurit, kui `</UL>`, siis vähendame.

Kui real on midagi muud, siis tekitab see rida väljundit ka. Kõigepealt väljastame kaks tühikut iga taseme kohta, välja arvatud viimane. Kui rea alguses on `<LI>`, siis väljastatame `*`, tühiku ja real `<LI>` järel oleva teksti, vastasel korral kaks tühikut ja terve rea sisu.

Selline lahendus on toodud failis `ullah1.pas`.

Testid

1. $N = 2$. Minimaalne test: tühi loetelu. 2 punkti.
2. $N = 5$. Ühetasemeline loetelu, kõik elemendid üherealised. 2 punkti.
3. $N = 5$. Ühetasemeline loetelu, mitmerealised elemendid. 2 punkti.
4. $N = 5$. Ühetasemeline loetelu, tühi element ja tühja rida sisaldav mitmerealine element. 2 punkti.
5. $N = 13$. Mitmetasemeline loetelu, kõik elemendid üherealised. 2 punkti.
6. $N = 28$. Mitmetasemeline loetelu, mitmerealised elemendid. 2 punkti.
7. $N = 249$. Mõõduka suurusega mitmetasemeline loetelu, mitmerealised elemendid. 2 punkti.
8. $N = 14$. Maksimumpikkusega sisendread, väljundridade pikkus ületab 255 märki. 2 punkti.
9. $N = 309$. Maksimumpikkusega sisendread, väljundridade pikkus ületab 255 märki. 2 punkti.
10. $N = 1000$. Maksimaalne test: 333 elementi, kõik üksteise sees. 2 punkti.

Kokku 20 punkti.

2. HTML-loetelu (edasijõudnud)

Tekitame tühja loendurite magasini. Edasi töötleme sisendit reakaupa. Kui rida on `<OL>`, siis lisame magasini tippu uue loenduri algväärtusega 0, kui `</OL>`, siis eemaldame magasinist tipmise loenduri.

Kui real on midagi muud, siis tekitab see rida väljundit ka. Kui rea alguses on `<LI>`, siis väljastatame kõigi magasinis olevate loendurite väärtused (põhja poolt tipu poole) ja real `<LI>` järel oleva teksti, vastasel korral vajaliku arvu tühikuid ja terve rea sisu.

Selline lahendus on toodud failis `ollah1.pas`.

Testid

1. $N = 2$. Minimaalne test: tühi loetelu. 2 punkti.
2. $N = 5$. Ühetasemeline loetelu, kõik elemendid üherealised. 2 punkti.
3. $N = 5$. Ühetasemeline loetelu, mitmerealised elemendid. 2 punkti.
4. $N = 5$. Ühetasemeline loetelu, tühi element ja tühja rida sisaldav mitmerealine element. 2 punkti.
5. $N = 13$. Mitmetasemeline loetelu, kõik elemendid üherealised. 2 punkti.
6. $N = 28$. Mitmetasemeline loetelu, mitmerealised elemendid. 2 punkti.
7. $N = 249$. Mõõduka suurusega mitmetasemeline loetelu, mitmerealised elemendid, mitmekohalised numbrid. 2 punkti.
8. $N = 14$. Maksimumpikkusega sisendread, väljundridade pikkus ületab 255 märki. 2 punkti.
9. $N = 309$. Maksimumpikkusega sisendread, väljundridade pikkus ületab 255 märki, mitmekohalised numbrid. 2 punkti.
10. $N = 1000$. Maksimaalne test: 333 elementi, kõik üksteise sees. 2 punkti.

Kokku 20 punkti.

3. Kasutajagrupid

Seda ülesannet võib muidugi püüda lahendada toore jõuga, asendades gruppide liikmete nimekirjades järjest alamgruppe nende liikmete nimekirjadega, kuni jõuame mingisse stabiilsesse seisu (ilmselt juhtub see siis, kui iga grupi liikmete nimekirjas on ainult kasutajad). See on aga üsna ebaefektiivne lahendus, sest halvimal juhul võib näiteks gruppide süsteem, kus iga grupp sisaldab üht kasutajat ja üht teist gruppi anda tulemuseks, et kõik kasutajad kuuluvad kõigisse gruppidesse. Asjaolude ebasoodsal kokkusattumisel võib selle tulemuseni jõudmiseks N grupi korral kuluda suurusjärgus N^3 operatsiooni.

Nagu sageli mingite objektide vahelisi suhteid vaatlevad ülesanded, on seegi edukalt sõnastatav graafiülesandena. Kuna grupikuuluvus ei ole üldiselt sümmeetriline seos (näiteks kasutaja kuulub gruppi, kuid mitte vastupidi), siis tuleb ilmselt kasutada suunatud graafi. Sisendandmeid vaadates võiks esimene mõte olla koostada graaf nii, et gruppi tähistava tipu juurest suunduvad kaared tema liikmeid tähistavatesse tippudesse, aga tegelikult on kasulikum suunata kaared vastupidi. Siis on antud kasutajat sisaldavate gruppide leidmiseks vaja leida kõik tipud, mis on saavutatavad seda kasutajat tähistavast tipust kaari mööda liikudes.

Suunatud graafis saavutatavate tippude leidmine on väga sarnane suunamata graafi sidususkomponentide leidmise ülesandega ja ka seda ülesannet võib lahendada nii sügavuti läbimisega (nagu on tehtud failis `grplah1.pas` toodud lahenduses) kui ka laiuti läbimisega (nagu on tehtud failis `grplah2.cpp` toodud lahenduses, mis kasutab järjekordade ja tipuhulkade realiseerimiseks C++ standardteegis leiduvaid andmestruktuure).

Testid

1. $N = 1$, $\max(N_i) = 3$. Üks grupp, uuritav kasutaja sellesse ei kuulu. 2 punkti.
2. $N = 3$, $\max(N_i) = 5$. Mõned grupid, liikmed ainult kasutajad. 2 punkti.
3. $N = 4$, $\max(N_i) = 5$. Kahetasemeline gruppide hierarhia. 4 punkti.
4. $N = 6$, $\max(N_i) = 5$. Kolmetasemeline gruppide hierarhia. 4 punkti.
5. $N = 6$, $\max(N_i) = 5$. Üks grupp sisaldab iseennast. 4 punkti.
6. $N = 8$, $\max(N_i) = 5$. Kaks gruppi sisaldavad üksteist vastastikku. 4 punkti.
7. $N = 10$, $\max(N_i) = 5$. Pikem tsükkel gruppide sisalduvuses. 4 punkti.
8. $N = 100$, $\max(N_i) = 10$. Mõõdukas juhuslik test. 4 punkti.
9. $N = 200$, $\max(N_i) = 20$. Keskmine juhuslik test. 4 punkti.
10. $N = 500$, $\max(N_i) = 50$. Suur juhuslik test. 4 punkti.
11. $N = 1000$, $\max(N_i) = 100$. Maksimaalne juhuslik test. 4 punkti.

Kokku 40 punkti.

4. Rõngassärk

Nagu ülesande lähemal uurimisel selgub, on tegu graafiülesandega. Nimelt saame vaadelda iga rõngast tipuna ja iga seost kahe rõnga vahel servana. Seega on meil vaja leida vähim selline tippude hulk, mis kataks ära kõik servad, st et iga serva vähemalt üks otspunkt oleks selles tippude hulgas, sest nende tippude eemaldamisel jääks kõik servad ühe otsata. Kuna serv on aga seos kahe rõnga vahel, tähendaks see selle seose kadumist. Seega on vaja mingil viisil leida selline tippude hulk.

Tegu on suhteliselt standardse ülesandega, mida tuntakse tipukatte (ingl *vertex cover*) ülesande nime all. Kahjuks ei teata selle lahendamiseks ühtegi polünomiaalse ajaga algoritmi ja seega tuleb see ülesanne lahendada variantide läbivaatuse meetodil. (Mõte, et alati suurima seoste arvuga rõnga eemaldamine viib parima lahenduseni, on paraku vale. Selline lahendus saaks umbes 14 punkti 40 võimalikust). Läbivaatuseks on mitu meetodit:

- Proovida läbi kõik tippude kombinatsioonide valikud: idee pole paha, kuid paraku jääb selline lähenemine suhteliselt aeglaseks, kuna variante on palju. Kui seda realiseerida järjestikuse süvenemise (ingl *iterative deepening*) abil, peaks lahendus saama umbes 24 punkti. Punkte annab juurde otsingupuu pügamine tippudest lähtuvate veel alles olevate seoste arvu järgi. Samuti on võimalik lisada graafi sidusateks komponentideks jagamine ja siis iga komponendi eraldi vaatamine. Täispunkte aga selline lahendus paraku ikkagi ei anna.
- Vaadelda graafi servhaaval: Iga serva vähemalt üks otstipp peab olema valitud. Seega võime graafi servhaaval läbi käia ja iga graafis veel alles oleva serva (st serva, mille kumbki otspunkt veel valitud pole) juures mõlemat otsippu kordamööda (kõigepealt üht, siis teist) eemaldada proovida. Üllataval kombel osutub selline lahendus eelnevast oluliselt kiiremaks, andes korraliku realisatsiooni korral koguni täispunktid.
- Vaadelda graafi tipphaaval, lähtudes siiski sellest, et iga serv tuleb lõpuks kustutada: Et iga serva vähemalt üks otspunkt peab olema valitud, peame me peale otsust mingit tippu mitte valida kindlasti valima lõpuks kõik temaga serva abil seotud olevad naabrid. Seega peame iga tipu juures tegema otsuse, kas valida see tipp, või valida kõik ta naabrid. Naabrite kohe maha tõmbamisega pärast tipu mittevalimise otsustamist säästame märkimisväärselt aega ja lisatööd. Selline läbivaatuslahendus on eelnevast kahest oluliselt kiirem ja annab seega samuti täispunktid.

Testid

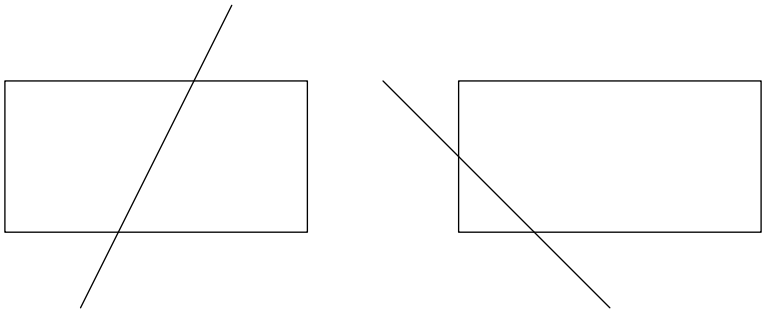
1. 0 tippu, 0 serva. Vastus 0. 2 punkti.
2. 5 tippu, 4 serva. 5 lüliga ahel. Tippude numbrid juhuslikud $1 \ldots 100$. Vastus 2. 2 punkti.
3. 20 tippu, 190 serva. 20 tipuga täisgraaf. Vastus 19. 4 punkti.
4. 48 tippu, 75 serva. 8×6 jupp tavalist rõngaskudet. Vastus 24. 4 punkti.
5. 9 tippu, 12 serva. Lihtne ahne algoritmi kontranäide. Vastus 4. 4 punkti.
6. 16 tippu, 16 serva. 16 tipuga tsükkel. Servad on antud järjekorras, mis peaks ahne algoritmi eduvõimalused minimeerima. Vastus 8. 4 punkti.
7. 20 tippu, 144 serva. Juhuslikult genereeritud sidus graaf. Vastus 16. 4 punkti.
8. 30 tippu, 124 serva. Kaks 15-tipulist juhuslikult genereeritud komponenti. Vastus 22. 4 punkti.
9. 42 tippu, 123 serva. Kaks juhuslikult genereeritud komponenti + graaf testist 5. Vastus 23. 4 punkti.
10. 43 tippu, 77 serva. 7×3 jupp rõngaskoest toru + tipurõngas ahne algoritmi eksitamiseks. Vastus 21. 4 punkti.
11. 49 tippu, 90 serva. 6×4 jupp rõngaskoest toru + tipurõngas ahne algoritmi eksitamiseks. Vastus 24. 4 punkti.

Kokku 40 punkti.

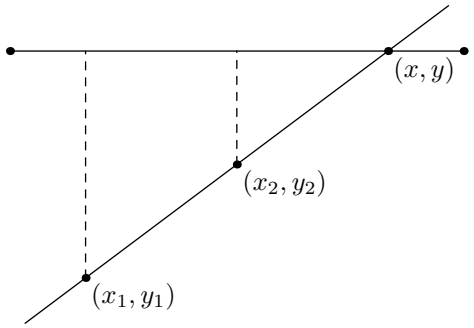
5. Ristküliku värvimine

Selle ülesande lahendamist on kasulik alustada tähelepanekust, et sirge võib ristkülikut lõigata ainult kahel põhimõtteliselt erineval viisil:

- kui sirge lõikab ristküliku vastaskülgi, jagab ta ristküliku kaheks trapetsiks (joonisel vasakul), mille pindalad on kergesti arvutatavad lõikepunktide ja ristküliku tippude omavaheliste kauguste järgi;
- kui sirge lõikab ristküliku lähiskülgi, jagab ta ristküliku täisnurkseks kolmnurgaks ja (mingiks) viisnurgaks (joonisel paremal); sarnaselt eelmise juhuga on kolmnurga pindala kergesti arvutatav lõikepunktide ja ristküliku tippude omavaheliste kauguste järgi; viisnurga pindala otse leida oleks üsna tülikas, aga selleks pole ka vajadust: viisnurga pindala on see, mis ristkülikust pärast kolmnurga "lahutamist" järele jääb.



Ei tohiks olla raske märgata, et (tänu sellele, et ristküliku küljed on paralleelsed koordinaattelgedega) piisab sirge ja ristküliku külgede lõikepunktide koordinaatide leidmiseks sarnaste täisnurksete kolmnurkade põhiomaduste tundmisest.



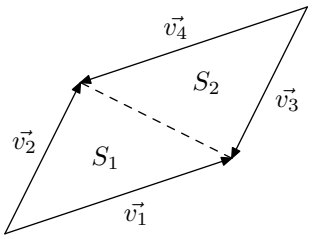
Just nendel tähelepanekutel põhinebki failis `ristlah1.pas` toodud lahendus.

Testid

20 testi, igaüks 2 punkti, kokku 40 punkti.

6. Nelinurga värvimine

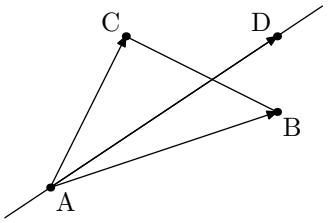
Selle ülesande lahendamist võib alustada näiteks tähelepanekust, et kumera nelinurga diagonaal (nelinurga vastastippe ühendav sirglõik; alloleval joonisel punktiiriga) jagab selle alati kaheks kolmnurgaks ja kummagi kolmnurga pindala on leitav selle külgedeks olevate nelinurga külgede vektorkorrutisena: $S = S_1 + S_2 = \frac{1}{2}|\vec{v_1} \times \vec{v_2}| + \frac{1}{2}|\vec{v_3} \times \vec{v_4}|$.



Edasi on, sarnaselt algajate rühma ülesandega, kasulik tähele panna, et

- kui sirge lõikab nelinurga vastaskülgi, jagab ta selle kaheks nelinurgaks, mille pindalad on omakorda arvutatavad eeltoodud võttega;
- kui sirge lõikab nelinurga lähiskülgi, jagab ta selle kolmnurgaks ja viisnurgaks; kolmnurga pindala on arvutatav selle kahe külje vektorkorrutise abil ja viisnurga pindala saame nelinurga pindalast kolmnurga oma lahutamisega.

Muidugi on selle kõige jaoks vaja osata kontrollida, kas antud sirge ja antud lõik lõikuvad ja lõikumise korral leida ka lõikepunkti koordinaadid. Taas tuleb appi vektorkorrutis: sirge AD ja lõik BC lõikuvad, kui lõigu otspunktid asuvad teine teisel pool sirget ja seda saab kontrollida vektorkorrutiste $\vec{AD} \times \vec{AB}$ ja $\vec{AD} \times \vec{AC}$ märkide võrdlemisega.



Lõikumise korral jagab sirge kolmnurga ABC kaheks kolmnurgaks, mille pindalade suhe on sama, mis vektorkorrutiste $\vec{AD} \times \vec{AB}$ ja $\vec{AD} \times \vec{AC}$ absoluutväärtuste oma, järelikult saab ka lõikepunkti koordinaadid leida vektorkorrutise abil!

Testid

20 testi, igaüks 2 punkti, kokku 40 punkti.