1. TEHTED HULKADEGA

30 punkti 10 sekundit

Kirjutada programm, mis sooritab tehteid hulkadega, mille elementideks on väikestest ladina tähtedest (a kuni z) moodustatud mittetühjad sõned (stringid).

Sisend: Tekstifaili `HULK.SIS` ainsal real (pikkusega kuni 100 sümbolit) on avaldis kujul $hulk_1$+$hulk_2$, $hulk_1$-$hulk_2$ või $hulk_1$*$hulk_2$, kus märgid `+`, `-` ja `*` tähistavad vastavalt hulkade ühendit, vahet ja ühisosa. Hulgad on esitatud loogelistesse sulgudesse paigutatud elementide nimekirjana. Elemendid on üksteisest eraldatud komadega ja ükski element ei esine ühes hulgas üle ühe korra. Elementide nimekiri võib olla ka tühi.

Väljund: Tekstifaili `HULK.VAL` ainsale reale väljastada sisendis kujutatud tehte tulemus sisendi kirjelduses esitatud kirjaviisis. Elementide järjekord nimekirjas ei oma tähtsust.

Märkus: Kahe hulga ühend koosneb elementidest, mis kuuluvad vähemalt ühte neist kahest hulgast. Kahe hulga vahe koosneb elementidest, mis kuuluvad esimesse, kuid mitte teise hulka. Kahe hulga ühisosa koosneb elementidest, mis kuuluvad mõlemasse hulka.

Näide1:

`HULK.SIS`
```
{a,abc,ca,aad}+{a,bb,ca}
```
`HULK.VAL`
```
{a,abc,ca,aad,bb}
```

Näide2:

`HULK.SIS`
```
{a,b}-{}
```
`HULK.VAL`
```
{a,b}
```

2. LOOSUNGID

30 punkti 10 sekundit

Loosungeid tootev väikefirma tellib loosungite valmistamiseks vajaliku materjali tehasest blokkidena `XY`, `YX`, `XXY`, `XYY`, `YXX` või `YYX`, kus `X` tähistab suvalist kaashäälikut ja `Y` suvalist täishäälikut. Tehas on igat liiki blokkidele kehtestanud kindla hinna. Et saada maksimaalset kasu, tellib väikefirma iga loosungi jaoks materjale odavaima võimaliku blokkideks tükelduse järgi, kuid esitab tellijale arve kalleima võimaliku tükelduse järgi.

Sisend: Tekstifaili `LOOSUNG.SIS` esimesel real on 6 tühikutega eraldatud täisarvu vahemikus 1 kuni 100 – blokkide `XY`, `YX`, `XXY`, `XYY`, `YXX` ja `YYX` hinnad tehases. Faili teisel real (pikkusega kuni 1000 sümbolit) on väikestest ladina tähtedest ja tühikutest koosnev loosungi tekst.

Väljund: Tekstifaili `LOOSUNG.VAL` ainsale reale väljastada üks täisarv – kalleima võimaliku tükelduse ja odavaima võimaliku tükelduse hindade vahe. Kui antud loosungi teksti ei ole üldse võimalik antud blokkideks tükeldada, väljastada `-1`.

Märkus: Ladina tähtedest loetakse täishäälikuteks `a`, `e`, `i`, `o` ja `u` (kuid mitte `y`).

Näide:

`LOOSUNG.SIS`
```
1 1 1 1 1 1
baabbaabbaab
```
`LOOSUNG.VAL`
```
2
```

3. **PROTSESSITABEL** 40 punkti 10 sekundit

Mitmekasutajaoperatsioonisüsteemides (näiteks UNIX) võib korraga käia paljude erinevate kasutajate programme. Iga programmi käivitajaks on mingi kasutaja, kellele antud volitused määravad ära tema poolt käivitatud protsessi õigused mitmesuguste ressursside kasutamiseks. Protsess võib käivitada uue protsessi, siis nimetatakse teda uue protsessi vanemaks.

Arvuti käivitamisel tekib automaatselt protsess, mille kasutajatunnus on 0 (superkasutaja tunnus) ja protsessinumber 1. Kõik muud protsessid põlvnevad mõnest juba olemasolevast protsessist. Üldjuhul pärib uus protsess oma vanema kasutajatunnuse (st uus protsess saab samad õigused arvuti ressursside kasutamiseks), kuid superkasutaja protsessidel on lubatud tekitada muudele kasutajatele kuuluvaid protsesse. Kui tavakasutaja protsess tekitab uue protsessi, mis töötab teise kasutaja õigustes, seab see ohtu teise kasutaja andmete turvalisuse, sellepärast on arvuti administraatoril vaja programmi selliste rikkumiste leidmiseks.

Sisend: Tekstifaili `PROTS.SIS` esimesel real on arvutis töötavate protsesside koguarv `N` ($1 \le N \le 1000$) ja järgmisel `N` real igaühel informatsioon ühe protsessi kohta. Protsessi info on antud kujul `PID PPID UID`, kus `PID` on protsessi ja `PPID` tema vanema täisarvulised numbrid ($1 \le PPID \le PID \le 10000$) ja `UID` on protsessi omaniku kasutajatunnus ($0 \le UID \le 10000$). Protsessid on esitatud nende numbrite (`PID`) kasvamise järjekorras. Juurprotsessi (`PID=1`) vanemaks loetakse kokkuleppeliselt teda ennast.

Väljund: Tekstifaili `PROTS.VAL` esimesele reale väljastada eelpool toodud turvanõuet rikkuvate kasutajate arv `X` ja järgmisele `X` reale igaühele ühe rikkuja informatsioon kujul $UID\ UID_1\ UID_2\ \ldots$, kus `UID` on selle kasutaja tunnus, kelle protsessidest põlvneb teiste kasutajate protsesse ning UID_1, UID_2 jne on nende kasutajate tunnused, kelle protsessid põlvnevad (otseselt või kaudselt) `UID` protsessidest. Rikkujaks loetakse ainult rikkumiste ahela initsiaatorit (st kui ühe kasutaja protsess tekitab teise kasutaja protsessi ja see omakorda kolmanda kasutaja protsessi, siis on teine kasutaja esimese ohver ja teda ei pea süüdlasena välja tooma). Väljundi read järjestada `UID` kasvamise järjekorras ning UID_1, UID_2 jne igal real järjestada kasvavas järjekorras.

Näide:

```
PROTS.SIS
10
1 1 0
2 1 1
4 1 4
5 4 0
6 1 7
8 2 2
9 6 7
10 8 3
11 5 5
12 6 7
```

```
PROTS.VAL
2
1 2 3
4 0 5
```