1. Kaubavedu

1 sekund 20 punkti

Auto veab kaupa laost mitmesse poodi. On teada, kui palju kulub autol aega laos kauba pealelaadimiseks ja igas poes selle poe kauba mahalaadimiseks. Samuti on teada, kui palju kulub autol aega ühest kohast teise sõitmiseks.

Lisaks on nii lao kui ka iga poe kohta teada ajavahemik, mille jooksul auto seal kaupa laadida saab. Kui auto jõuab lao või poe juurde enne selle ajavahemiku algust, peab ta kauba laadimisega ootama. Kui auto jõuab mõne poe juurde nii hilja, et ei jõua enne lubatud aja lõppu kaupa maha laadida, siis ta ei saagi sinna poodi kaupa viia.

Kirjutada programm, mis kontrollib, kas auto saab kõigisse poodidesse kauba kohale ja enne lao sulgemist aruande sinna tagasi viia ning leiab selleks autojuhile vähima ajakuluga võimaluse.

Sisend. Tekstifaili `kaup.sis` esimesel real on üks täisarv, poodide arv N ($1 \leqslant N \leqslant 100$). Faili teisel real on lao ja järgmisel N real poodide andmed nende külastamise järjekorras.

Igaühel neist $N + 1$ reast on neli tühikutega eraldatud andmevälja, lao või poe avamise ja sulgemise kellaaeg, kauba laadimiseks ja järgmisse kohta (lao korral esimesse poodi, viimase poe korral lattu tagasi, muidu järgmisse poodi) sõitmiseks kuluv aeg. Kõik kellaajad on antud kujul $HH:MM$ ($00 \leqslant HH \leqslant 23$, $00 \leqslant MM \leqslant 59$), kõik ajakulud minutites.

Väljund. Tekstifaili `kaup.val` esimesele reale väljastada sõna `JAH` või `EI` vastavalt sellele, kas autol on võimalik kaup kõigisse poodidesse kohale viia või mitte. Kui see on võimalik, väljastada faili teisele reale vähim võimalik ajakulu alates kauba laadimiseks lao juurde jõudmisest kuni kõigi poodide külastamise järel lao juurde tagasi jõudmiseni ja kolmandale reale varaseim võimalik lao juurde saabumise kellaeg, mille korral saab optimaalse plaani järgi sõita.

Näide.

`kaup.sis`
```
2
06:00 23:00 20 10
07:00 08:00 10 20
08:00 09:00 10 20
```

`kaup.val`
```
JAH
90
07:00
```

Optimaalne sõiduplaan on selline: auto saabub lao juurde kell 07:00, kulutab 20 minutit kauba laadimiseks ja 10 minutit esimese poe juurde sõitmiseks; jõuab esimese poe juurde kell 07:30, kulutab 10 minutit kauba laadimiseks ja 20 minutit teise poe juurde sõitmiseks; jõuab teise poe juurde 08:00, kulutab 10 minutit kauba laadimiseks ja 20 minutit lao juurde tagasi sõitmiseks; jõuab lao juurde tagasi kell 08:30, ehk 90 minutit pärast esimest saabumist.

Näide.

`kaup.sis`
```
1
07:00 08:00 20 20
07:00 08:00 20 20
```

`kaup.val`
```
EI
```

Kuna kahe laadimise ja kahe sõitmise peale kokku kulub 80 minutit, ei ole kuidagi võimalik, et auto jõuaks lao juurde tagasi enne kui see kinni pannakse.

2. Sularaha

3 sekundit 40 punkti

Teatavasti on Eestis viimaseid nädalaid käibel 1-, 2-, 5-, 10-, 25-, 50-, 100- ja 500-kroonised paberrahad.

Kirjutada nende mälestuseks programm, mis leiab, mitmel erineval viisil on nendega võimalik maksta antud summa.

Kaks maksmisviisi lugeda erinevateks, kui vähemalt üht nominaali on kasutusel erinev arv. Samade rahatähtede teises järjekorras kasutamist eraldi maksmisviisiks mitte lugeda.

Sisend. Tekstifaili `raha.sis` ainsal real on täisarv S ($0 \leqslant S \leqslant 1\,000\,000$), makstav summa.

Väljund. Tekstifaili `raha.val` ainsale reale väljastada erinevate võimalike maksmisviiside arv.

Näide.

`raha.sis`	`raha.val`
`6`	`5`

6-kroonise summa maksmiseks on järgmised võimalused:
$6 = 1 + 1 + 1 + 1 + 1 + 1$,
$6 = 2 + 1 + 1 + 1 + 1$,
$6 = 2 + 2 + 1 + 1$,
$6 = 2 + 2 + 2$,
$6 = 5 + 1$.

3. Moodulite optimeerimine

1 sekund 40 punkti

Suuremad programmid jagatakse parema hallatavuse huvides sageli mooduliteks. Iga moodul sisaldab hulga funktsioone või klasse ning kui mõnes teises moodulis läheb neid funktsioone või klasse vaja, võib teine moodul esimese importida.

Näiteks selleks, et moodul `Game`, mis vastutab mänguloogika eest, saaks kasutada funktsioone moodulitest `Math` ja `IO`, võib mooduli `Game` koodi kirjutada käsu "`import Math, IO`". Moodul `IO` võib omakorda importida moodulid `Screen` ja `File` käsuga "`import Screen, File`".

Kui üks moodul impordib teise, saab ta automaatselt juurdepääsu ka kõigile neile moodulitele, mida impordib teine moodul, ning lisaks ka kõigile neile, mida impordivad teise mooduli poolt imporditud moodulid, j.n.e.

Ütleme, et moodul X kasutab moodulit Y, kui moodul X impordib mooduli Y või leidub moodulite jada A_1, A_2, ..., A_n nii, et X impordib A_1, A_1 impordib A_2, ..., A_n impordib Y.

Kirjutada programm, mis optimeerib antud moodulite importimiskäsud nii, et ükski moodul ei impordiks ühtki teist moodulit, mille funktsioonidele ja klassidele tal niikuinii juurdepääs olemas on.

Sisend. Tekstifaili `mood.sis` esimesel real on täisarv N ($1 \leqslant N \leqslant 1000$), teisi mooduleid importivate moodulite arv programmis.

Järgmisel N real on igaühel ühe mooduli nimi ja sellest kooloniga eraldatult tema imporditavate moodulite loetelu, kus moodulite nimed on eraldatud komadega.

Iga mooduli nimi koosneb 1 ... 20 suurest ja väikesest ladina tähest ja nimed on tõstutundlikud. Sisendfaili ühegi rea pikkus ei ületa 250 märki ja moodulite koguarv süsteemis ei ületa 2000. Võib eeldada, et ükski moodul ei kasuta iseennast.

Väljund. Tekstifaili `mood.val` väljastada täpselt N rida. Igale reale väljastada ühe mooduli ja tema imporditavate moodulite nimekiri pärast optimeerimist. Ridade järjekord failis ja moodulite järjekord real pole olulised.

Näide.

`mood.sis`

```
3
Game:Math,Screen,IO
IO:Screen,File
File:System
```

`mood.val`

```
Game:Math,IO
IO:Screen,File
File:System
```

Moodul `Game` saab juurdepääsu mooduli `Screen` sisule mooduli `IO` kaudu.

1. Eelmine arv

1 sekund 20 punkti

Vaatleme järgmist eeskirja neljakohalisest arvust uue neljakohalise arvu saamiseks:

$$2345 \to 23 + 2 + 3 = 28,\ 45 + 4 + 5 = 54 \to 2854.$$

Eeskirja saab rakendada ainult juhul, kui mõlema liitmise tulemuseks on kahekohalised arvud.

Kirjutada programm, mis leiab eeltoodud reegli järgi moodustatud jadas antud arvule eelneva.

Sisend. Tekstifaili `eelm.sis` ainsal real on neljakohaline positiivne täisarv N.

Väljund. Tekstifaili `eelm.val` ainsale reale väljastada neljakohaline positiivne täisarv M, millest eelpool kirjeldatud reegli rakendamisega saab sisendis antud arvu N. Kui võimalikke vastuseid on mitu, väljastada ükskõik milline neist. Kui pole ühtki võimalust, väljastada sõna `POLE`.

Näide.

`eelm.sis`	`eelm.val`
`2854`	`2345`

Näide.

`eelm.sis`	`eelm.val`
`1010`	`POLE`

Hindamine. Selles ülesandes saavad `POLE`-vastusega testide eest punkte ainult need programmid, mis lahendavad õigesti vähemalt ühe testi, kus eelmine arv leidub.

2. Kaubavedu

1 sekund 40 punkti

Auto veab kaupa laost mitmesse poodi. On teada, kui palju kulub autol aega laos kauba pealelaadimiseks ja igas poes selle poe kauba mahalaadimiseks. Samuti on teada, kui palju kulub autol aega ühest kohast teise sõitmiseks.

Lisaks on nii lao kui ka iga poe kohta teada ajavahemik, mille jooksul auto seal kaupa laadida saab. Kui auto jõuab lao või poe juurde enne selle ajavahemiku algust, peab ta kauba laadimisega ootama. Kui auto jõuab mõne poe juurde nii hilja, et ei jõua enne lubatud aja lõppu kaupa maha laadida, siis ta ei saagi sinna poodi kaupa viia.

Kirjutada programm, mis kontrollib, kas auto saab kõigisse poodidesse kauba kohale ja enne lao sulgemist aruande sinna tagasi viia ning leiab selleks autojuhile vähima ajakuluga võimaluse.

Sisend. Tekstifaili `kaup.sis` esimesel real on üks täisarv, poodide arv N ($1 \leqslant N \leqslant 100$). Faili teisel real on lao ja järgmisel N real poodide andmed nende külastamise järjekorras.

Igaühel neist $N + 1$ reast on neli tühikutega eraldatud andmevälja, lao või poe avamise ja sulgemise kellaaeg, kauba laadimiseks ja järgmisse kohta (lao korral esimesse poodi, viimase poe korral lattu tagasi, muidu järgmisse poodi) sõitmiseks kuluv aeg. Kõik kellaajad on antud kujul $HH:MM$ ($00 \leqslant HH \leqslant 23$, $00 \leqslant MM \leqslant 59$), kõik ajakulud minutites.

Väljund. Tekstifaili `kaup.val` esimesele reale väljastada sõna `JAH` või `EI` vastavalt sellele, kas autol on võimalik kaup kõigisse poodidesse kohale viia või mitte. Kui see on võimalik, väljastada faili teisele reale vähim võimalik ajakulu alates kauba laadimiseks lao juurde jõudmisest kuni kõigi poodide külastamise järel lao juurde tagasi jõudmiseni ja kolmandale reale varaseim võimalik lao juurde saabumise kellaeg, mille korral saab optimaalse plaani järgi sõita.

Näide.

`kaup.sis`
```
2
06:00 23:00 20 10
07:00 08:00 10 20
08:00 09:00 10 20
```

`kaup.val`
```
JAH
90
07:00
```

Optimaalne sõiduplaan on selline: auto saabub lao juurde kell 07:00, kulutab 20 minutit kauba laadimiseks ja 10 minutit esimese poe juurde sõitmiseks; jõuab esimese poe juurde kell 07:30, kulutab 10 minutit kauba laadimiseks ja 20 minutit teise poe juurde sõitmiseks; jõuab teise poe juurde 08:00, kulutab 10 minutit kauba laadimiseks ja 20 minutit lao juurde tagasi sõitmiseks; jõuab lao juurde tagasi kell 08:30, ehk 90 minutit pärast esimest saabumist.

Näide.

`kaup.sis`
```
1
07:00 08:00 20 20
07:00 08:00 20 20
```

`kaup.val`
```
EI
```

Kuna kahe laadimise ja kahe sõitmise peale kokku kulub 80 minutit, ei ole kuidagi võimalik, et auto jõuaks lao juurde tagasi enne kui see kinni pannakse.

3. Surnud ring

1 sekund 40 punkti

Suuremad programmid jagatakse parema hallatavuse huvides sageli mooduliteks. Iga moodul sisaldab hulga funktsioone või klasse ning kui mõnes teises moodulis läheb neid funktsioone või klasse vaja, võib teine moodul esimese importida.

Näiteks selleks, et moodul `Game`, mis vastutab mänguloogika eest, saaks kasutada funktsioone moodulitest `Math` ja `IO`, võib mooduli `Game` koodi kirjutada käsu "`import Math, IO`". Moodul `IO` võib omakorda importida moodulid `Screen` ja `File` käsuga "`import Screen, File`".

Kui üks moodul impordib teise, saab ta automaatselt juurdepääsu ka kõigile neile moodulitele, mida impordib teine moodul, ning lisaks ka kõigile neile, mida impordivad teise mooduli poolt imporditud moodulid, j.n.e.

Ütleme, et moodul X kasutab moodulit Y, kui moodul X impordib mooduli Y või leidub moodulite jada $A_1, A_2, \ldots, A_n$ nii, et X impordib A_1, A_1 impordib $A_2, \ldots, A_n$ impordib Y.

Kirjutada programm, mis kontrollib, kas moodulite vahel on "surnud ring", kus moodul X kasutab moodulit Y ja moodul Y kasutab omakorda moodulit X.

Sisend. Tekstifaili `ring.sis` esimesel real on täisarv N ($1 \leqslant N \leqslant 1000$), moodulite arv programmis. Moodulid on tähistatud täisarvudega $1 \ldots N$.

Faili järgmisel N real on igaühel ühe mooduli kirjeldus. Real $1 + i$ on kõigepealt mooduli i imporditavate moodulite arv K_i ($0 \leqslant K_i \leqslant 20$) ja selle järel tühikutega eraldatult nende moodulite tähised $A_{i,1}, A_{i,2}, \ldots, A_{i,K_i}$.

Väljund. Kui moodulite kasutuses ei ole ühtki "surnud ringi", väljastada tekstifaili `ring.val` ainsale reale sõna `EI`. Vastasel juhul väljastada faili esimesele reale sõna `JAH` ja teisele reale tühikuga eraldatult kahe üksteist vastastikku kasutava mooduli tähised. Kui "surnud ringe" on mitu, väljastada ükskõik milline neist.

Näide.

`ring.sis`
```
4
2 2 3
2 3 4
0
0
```

`ring.val`
```
EI
```

Moodul 1 impordib kaks moodulit, 2 ja 3. Moodul 2 impordib samuti kaks moodulit, 3 ja 4. Moodulid 3 ja 4 ei impordi midagi.

Näide.

`ring.sis`
```
4
2 4 2
1 3
1 2
0
```

`ring.val`
```
JAH
2 3
```

Hindamine. Selles ülesandes saavad `EI`-vastusega testide eest punkte ainult need programmid, mis lahendavad õigesti vähemalt ühe `JAH`-vastusega testi.

1. Arvude jagumine

1 sekund 20 punkti

Kirjutada programm, mis leiab kolme antud täisarvu hulgas kõik erinevad üksteisega jaguvate arvude paarid.

Arve x ja y nimetame üksteisega jaguvate arvude paariks arvude A_1, A_2 ja A_3 hulgas, kui arv y jagub arvuga x ning leiduvad i ja j, mille korral $x = A_i$, $y = A_j$ ja $i \neq j$. Paare x_1 ja y_1 ning x_2 ja y_2 nimetame erinevateks, kui $x_1 \neq x_2$ või $y_1 \neq y_2$ (või mõlemat korraga).

Sisend. Tekstifaili `jagu.sis` ainsal real on kolm tühikutega eraldatud täisarvu A_1, A_2, A_3, kõik väärtustega $1 \ldots 10\,000$.

Väljund. Tekstifaili `jagu.val` igale reale väljastada kaks tühikuga eraldatud täisarvu x ja y, üks eeltoodud nõuetele vastav paar. Ridade järjekord failis pole oluline, kuid arvude järjekord real on. Kui ühtki jaguvate arvude paari pole, väljastada faili ainsale reale sõna `POLE`.

Näide.

`jagu.sis`
```
1 2 3
```
`jagu.val`
```
1 2
1 3
```

Näide.

`jagu.sis`
```
1 2 2
```
`jagu.val`
```
1 2
2 2
```

Näide.

`jagu.sis`
```
2 3 5
```
`jagu.val`
```
POLE
```

Hindamine. Selles ülesandes saavad `POLE`-vastusega testide eest punkte ainult need programmid, mis lahendavad õigesti vähemalt ühe testi, kus jaguvaid paare leidub.

2. Järjekord

1 sekund 40 punkti

Teeninduspunkti, kus ühe kliendi teenindamine võtab aega M minutit, saabuvad ettemääramata aegadel kliendid. Kui kliendi saabudes on teenindaja vaba, alustab ta kohe uue kliendi teenindamist. Vastasel juhul jääb uus klient järjekorda ootama ja teenindaja alustab järjekorras järgmise kliendi teenindamist kohe, kui on eelmise kliendiga lõpetanud.

Kirjutada programm, mis leiab klientide saabumise aegade põhjal viimase kliendi teenindamise lõpu aja.

Sisend. Tekstifaili `saba.sis` esimesel real on kaks tühikuga eraldatud täisarvu, klientide arv N ($1 \leqslant N \leqslant 100$) ja ühe kliendi teenindamiseks kuluv aeg M ($1 \leqslant M \leqslant 100$). Järgmisel N real on igaühel ühe kliendi saabumise aeg kujul $HH{:}MM$ ($00 \leqslant HH \leqslant 23$, $00 \leqslant MM \leqslant 59$). Ajad on antud klientide saabumise järjekorras.

Väljund. Tekstifaili `saba.val` ainsale reale väljastada viimase kliendi teenidamise lõpu aeg samuti kujul $HH{:}MM$. Võib eeldada, et vastus ei ületa 23:59.

Näide.

`saba.sis`
```
3 20
12:20
13:10
13:20
```
`saba.val`
```
13:50
```

Esimese kliendi teenindamine lõpeb kell 12:40. Teise kliendi teenindamine lõpeb kell 13:30. Seega peab kolmas klient selle ajani ootama ja tema teenindamine lõpebki kell 13:50.

3. Sularaha

3 sekundit 40 punkti

Teatavasti on Eestis viimaseid nädalaid käibel 1-, 2-, 5-, 10-, 25-, 50-, 100- ja 500-kroonised paberrahad.

Kirjutada nende mälestuseks programm, mis leiab, mitmel erineval viisil on nendega võimalik maksta antud summa.

Kaks maksmisviisi lugeda erinevateks, kui vähemalt üht nominaali on kasutusel erinev arv. Samade rahatähtede teises järjekorras kasutamist eraldi maksmisviisiks mitte lugeda.

Sisend. Tekstifaili `raha.sis` ainsal real on täisarv S ($0 \leqslant S \leqslant 250$), makstav summa.

Väljund. Tekstifaili `raha.val` ainsale reale väljastada erinevate võimalike maksmisviiside arv.

Näide.

`raha.sis`	`raha.val`
`6`	`5`

6-kroonise summa maksmiseks on järgmised võimalused:
$6 = 1 + 1 + 1 + 1 + 1 + 1$,
$6 = 2 + 1 + 1 + 1 + 1$,
$6 = 2 + 2 + 1 + 1$,
$6 = 2 + 2 + 2$,
$6 = 5 + 1$.