

## 1. Kolmnurgad

Üldiselt pole selles ülesandes palju paremat lahendust kui vaadata läbi kõik punktikolmikud. Seega on ülesande sisuline raskuspunkt vaadeldava kolmnurga täisnurksuse kontrollimine.

Failis `kolmlah1.cpp` olev lahendus vaatab läbi kõik punktikolmikud, sealjuures tippude erinevad järjekorrad. Sellise läbivaatuse käigus satub iga täisnurkne kolmnurk kindlasti ette ka nii, et täisnurga tipp on kolmikus esimene. Sellisel juhul on kohe teada, millised servad on kaatetid ja kolmnurga täisnurksust on lihtne kontrollida nende skalaarkorrutise põhjal (nagu on tehtud näidislahenduses) või Pythagorase teoreemi abil.

Failis `kolmlah2.pas` olev lahendus vaatab samuti läbi kõik punktikolmikud, aga iga kolmikut ainult ühes järjekorras. Seega ei saa selle lahenduse täisnurksuse kontrollimise osa täisnurga asukoha kohta midagi eeldada. Muidugi on kolmnurga kolme külje hüpotenuusiks ja kaatetiteks jagamine lihtne — hüpotenuus on pikim.

Ülesande ainuke “nõks” on asjaolu, et kuigi punktide koordinaadid on täisarvud, võivad nendest punktidest moodustatud kolmnurkade pindalad olla murdarvud.

### Testid

1.  $N = 4$ . Vastus  $\{1, 2, 4\}$ . Kõik punktid I veerandis. Üks täisnurkne kolmnurk, selle kaatetid telgedega paralleelsed. 3 punkti.
2.  $N = 6$ . Vastus  $\{1, 3, 5\}$ . Kõik punktid I veerandis. Üks täisnurkne kolmnurk, selle kaatetid telgedega mitteparalleelsed. 3 punkti.
3.  $N = 4$ . Vastus  $\{1, 3, 4\}$ . Murdarvulised pindalad, ülespoole ümardajad saavad vale vastuse. 3 punkti.
4.  $N = 5$ . Vastus  $\{1, 4, 5\}$ . Murdarvulised pindalad, allapoole ümardajad saavad vale vastuse. 3 punkti.
5.  $N = 14$ . Vastus  $\{5, 8, 12\}$ . Punktid mitmes veerandis. 3 punkti.
6.  $N = 10$ . Vastus  $\{1, 5, 7\}$ . Ühegi kolmnurga ükski külg pole telgedega paralleelne. 3 punkti.
7.  $N = 20$ ,  $|x_i| \leq 100$ ,  $|y_i| \leq 100$ . Vastus  $\{1, 10, 11\}$ . Keskmise suurusega juhuslik test. Üks täisnurkne kolmnurk. 3 punkti.
8.  $N = 30$ ,  $|x_i| \leq 10$ ,  $|y_i| \leq 10$ . Vastus  $\{1, 8, 27\}$ . Keskmise suurusega juhuslik test. Palju täisnurkseid kolmnurki. 3 punkti.
9.  $N = 80$ ,  $|x_i| \leq 300$ ,  $|y_i| \leq 300$ . Vastus  $\{19, 60, 78\}$ . Suur juhuslik test. 3 punkti.
10.  $N = 100$ ,  $|x_i| \leq 1000$ ,  $|y_i| \leq 1000$ . Vastus  $\{5, 52, 83\}$ . Maksimaalne juhuslik test. 3 punkti.

Kokku 30 punkti.

## 2. Ristkülikud

Muidugi võib seda ülesannet lahendada kõigi punktinelikute läbivaatusega. Kahjuks on see kaunis ebaefektiivne. Kui me vaatame kõiki nelikuid kõigis võimalikes järjekordades, tuleb meil  $N$  punkti analüüsimiseks läbi vaadata  $N(N-1)(N-2)(N-3)$  nelikut. Veidi parem mõte oleks vaadata iga nelikut läbi ainult üks kord, ühes järjekorras. Sel juhul oleks vaja uurida vaid  $\binom{N}{4}$  nelikut. Muidugi on sellisel juhul ühe neliku uurimine ise veidi keerulisem, sest me ei saa enam eeldada, et tipud satuvad vaatluse alla nende nelinurga serval esinemise järjekorras. Failis `ristlah0.pas` toodud lahendus vaatab iga nelikut läbi kolmes erinevas järjekorras, uurides seega kokku  $3\binom{N}{4}$  nelikut. Kuna iga neliku jaoks tuleb teha päris mitu tehet, jõuab see lahendus lubatud aja jooksul töödelda umbes 50 punkti.

Efektiivsem lahendus on panna tähele, et ristküliku mistahes kolm tippu moodustavad täisnurkse kolmnurga ja neljanda tipu koordinaadid on ülejäänud kolme järgi üsna lihtsalt arvutatavad. Seega võib nelikute asemel vaadata läbi kolmikuid ja puuduvat neljandat tippu otsida ainult täisnurkse kolmiku korral. Failis `ristlah1.cpp` toodud lahendus uurib iga punktikolmikut kõigis võimalikes järjekordades, failis `ristlah2.pas` toodud lahendus aga vaatab iga kolmikut ainult ühes järjekorras.

Kolmnurga täisnurksuse kontrolliks võib kasutada kas skalaarkorrutist (nagu `ristlah1.cpp`) või Pythagorase teoreemi (nagu `ristlah2.pas`). Neljanda tipu leidmiseks kasutavad mõlemad lahendused lineaarset otsingut, mille efektiivsus on selle ülesande väikeste andmemahutude korral veel talutav. Muidugi saaks neljanda punkti otsimist kiirendada, kui punktid enne töötlemist koordinaatide järgi indekseerida, aga kasu sellest oleks üsna tühine, sest suurem osa kolmnurki pole täisnurksed ja nende puhul neljanda punkti otsimiseni ei jõutagi.

Lahendustes `ristlah0.pas` ja `ristlah2.pas` võiks kohe programmi töö alguses arvutada eraldi abimassiivi välja kõigi punktipaaride vahelised kaugused, sest neid läheb iga punktipaari jaoks vaja korduvalt. Selle lihtsa nipiga võidakse `ristlah0.pas` juurde kaks testi.

### Testid

1.  $N = 16$ . Vastus  $\{2, 3, 4, 5\}$ . Kõik punktid I veerandis. Vähima ristküliku küljed telgedega paralleelsed. 3 punkti.
2.  $N = 20$ . Vastus  $\{2, 5, 9, 13\}$ . Vähima ristküliku küljed telgedega mitteparalleelsed. 3 punkti.
3.  $N = 20$ . Vastus  $\{1, 4, 7, 10\}$ . Vähima ristküliku tipud mitmes veerandis. 3 punkti.
4.  $N = 26$ . Vastus  $\{1, 7, 23, 26\}$ . Palju täisnurkseid kolmnurki. 3 punkti.
5.  $N = 30$ . Vastus  $\{3, 11, 14, 16\}$ . Ainult üks ristkülik. 3 punkti.
6.  $N = 30$ . Vastus  $\{1, 7, 23, 26\}, \{2, 27, 28, 30\}, \{8, 11, 18, 19\}, \{12, 14, 15, 25\}$ . Vastus pole ühene. 3 punkti.
7.  $N = 80$ . Vastus  $\{11, 27, 40, 71\}$ . Palju punkte, vähe ristkülikuid. 3 punkti.
8.  $N = 68$ . Vastus  $\{20, 21, 41, 58\}$ . Palju täisnurkseid kolmnurki, kaks ristkülikut. 3 punkti.
9.  $N = 100$ . Vastus  $\{45, 64, 80, 99\}$ . Palju ristkülikuid. 3 punkti.
10.  $N = 100$ . Vastus  $\{14, 29, 46, 69\}$ . Palju täisnurkseid kolmnurki, palju ristkülikuid. 3 punkti.

Kokku 30 punkti.

### 3. Korduv väärtus

Selle ülesande esimene pähetulev lahendus on võrrelda kõiki võimalikke elemendipaare ja korduva väärtuse leidmisel väljastada see vastusena. Selline lahendus on failis `kordlah0.pas`. See kulutab suurusjärku  $N^2$  võrdlustehet, mis 10 000-elementise või pikema jada korral jääb juba liiga aeglaseks.

Järgmine võimalus on jada elemendid ära sortida (ükskõik, kas kasvavalt või kahanevalt). Siis satuvad korduvad väärtused kõrvuti ja nende leidmiseks ei pea enam võrdlema kõiki paare, vaid ainult iga elementi talle järgnevaga (või talle eelnevaga). Võrdlusi kulub sel juhul suurusjärku  $N$  ja lahenduse tööajas domineerib sortimisele kulutatud aeg. Kui kasutada mõnd naiivset sortimis-algoritmi, kulub selleks suurusjärku  $N^2$  operatsiooni, mis jääb alates 10 000-elementisest jadast jälle liiga aeglaseks. Failis `kordlah1.pas` toodud lahendus kasutab kuhjameetodit, mille tööaeg on suurusjärku  $N \log N$ , ja mis mahub vabalt ajalimiidi sisse ka pikima lubatud jada töötlemisel.

Veel üks variant on loendada kõigi võimalike väärtuste esinemissagedused. Võimalikke väärtusi on täpselt  $N$ , seega pole igäihe jaoks eraldi loenduri tekitamine mingi probleem. Sel juhul kulub  $N$  sammu loendurite algväärtustamiseks, siis umbes  $N$  sammu jada töötlemiseks ja lõpuks veel korra  $N$  sammu korduva elemendi leidmiseks. Kokku seega suurusjärku  $N$  operatsiooni. Selline lahendus on failis `kordlah2.pas`.

Ja lõpuks on võimalik ka “matemaatiline” lahendus: on teada, et arvude  $1 \dots N$  summa on  $N(N+1)/2$ , seega saame otsitava väärtuse, kui liidame kõik jada elemendid ja lahutame tulemusest  $N(N+1)/2$ . Selline lahendus, mis on toodud failis `kordlah3.pas`, kulutab samuti  $N$  sammu, kuid vajab oma tööks  $N$ -elementise massiivi asemel vaid paari abimuutujat. Tuleb vaid olla ettevaatlik, et mitte tekitada summadega opereerimisel ületäitumist: kuigi jada elemendid mahuvad kõik 16-bitise täisarvu sisse, on nende summa hoidmiseks vaja juba 32-bitist muutujat.

#### Testid

1.  $N = 1$ . Vastus 1. Minimaalne test. 3 punkti.
2.  $N = 5$ . Vastus 1. Kordub minimaalne element. 3 punkti.
3.  $N = 5$ . Vastus 5. Kordub maksimaalne element. 3 punkti.
4.  $N = 7$ . Vastus 6. Korduv väärtus on sisendfailis esimene. 3 punkti.
5.  $N = 7$ . Vastus 4. Korduv väärtus on sisendfailis viimane. 3 punkti.
6.  $N = 100$ . Vastus 68. Keskmise suurusega juhuslik test. 3 punkti.
7.  $N = 1\,000$ . Vastus 701. Keskmise suurusega juhuslik test. Esimene, kus elementide summa ei mahu 16-bitise täisarvu sisse. 3 punkti.
8.  $N = 10\,000$ . Vastus 130. Suur juhuslik test. Esimene, kus ruutkeerukusega lahendus jääb ajahätta. 3 punkti.
9.  $N = 20\,000$ . Vastus 8 099. Suur juhuslik test. 3 punkti.
10.  $N = 30\,000$ . Vastus 17 962. Maksimaalne test. 3 punkti.

Kokku 30 punkti.

#### 4. Sortimine

Muidugi võib seda ülesannet lahendada otse tekstist lähtuvalt: igal sammul leiame vastavalt minimaalse või maksimaalse elemendi ning väljastame ja eemaldame selle. Järjestamata massiivis minimaalse või maksimaalse leidmiseks kulub  $O(N)$  operatsiooni. Kui me allesjäävate elementide järjekorra säilitamisest ei hooli, on efektiivne viis leitud elemendi eemaldamiseks see lihtsalt massiivi viimase elemendiga asendada. Otsimise keerukust see muidugi ei muuda ja nii on failis `sortlah0.pas` toodud lahenduse keerukus ikkagi  $O(N^2)$ .

Tõhusam lahendus on kõigepealt massiivi elemendid näiteks mittekahanevalt järjestada ja hakata neid siis siksak-järjekorras väljastama: kõigepealt sortimistulemuse viimane, siis esimene, siis eelviimane, siis teine, jne.

Failis `sortlah1.pas` toodud lahendus kasutab massiivi sortimiseks kiirmeetodit. Selle põhiidee on järgmine: valime massiivi elementide hulgast ühe ja tõstame kõik sellest väiksemad massiivi alguse ja suuremad massiivi lõpu poole (seda on võimalik teha massiivi ühekordse läbimisega), seejärel järjestame kummagi poole elemendid omavahel sama algoritmi rekursiivse kasutamisega. Kiirmeetodi keskmine tööaeg  $N$ -elemendise massiivi sortimisel on  $O(N \log N)$ . Halvim võimalik tööaeg on küll  $O(N^2)$ , aga see ei realiseeru praktikas peaaegu kunagi.

Failis `sortlah2.pas` toodud lahendus kasutab massiivi sortimiseks ühildusmeetodit. Selle põhiidee on järgmine: jagame massiivi pooleks ja järjestame kummagi poole omaette sama algoritmi rekursiivse kasutamisega, seejärel aga moodustame kahest järjestatud osast ühe, poole pikema järjestatud osa (seda on võimalik teha kummagi osa ühekordse läbimisega). Ühildusmeetodi tööaeg on  $O(N \log N)$  nii keskmiselt kui ka halvimal juhul, kuid massiivi sortimisel vajab ta  $O(N)$  lisamälu.

Veel üks võimalik  $O(N \log N)$  tööajaga sortimisalgoritm on kuhjameetod, mille realisatsiooni võib leida algajate rühma jadaülesande lahendusest `kordlah1.pas`.

#### Testid

1.  $N = 1$ . Minimaalne test. 1 punkt.
2.  $N = 2$ . Arvud vales järjekorras. 2 punkti.
3.  $N = 5$ . Kõik väärtused võrdsed. 2 punkti.
4.  $N = 10$ . Ainult kaks erinevat väärtust. 2 punkti.
5.  $N = 20$ . Suure absoluutväärtusega negatiivsed arvud. 2 punkti.
6.  $N = 5\,000$ . Keskmise juhuslik test. 3 punkti.
7.  $N = 10\,000$ . Suur juhuslik test. 5 punkti.
8.  $N = 30\,000$ . Maksimaalne juhuslik test. 7 punkti.
9.  $N = 30\,000$ . Maksimaalne test: sisend juba nõuetekohaselt sorteeritud. 1 punkt.
10.  $N = 30\,000$ . Maksimaalne test: sisend sorteeritud nõutule vastupidiselt (minimaalne, siis maksimaalne, ...). 1 punkt.
11.  $N = 30\,000$ . Maksimaalne test: mittekahanevalt sorteeritud jada. 2 punkti.
12.  $N = 30\,000$ . Maksimaalne test: mittekasvavalt sorteeritud jada. 2 punkti.

Kokku 30 punkti.

## 5. Silbitamine

Selle ülesande lahendamiseks tuleb muidugi kõigepealt hoolikalt tekstis toodud reegleid lugeda. Seda tehes võib märgata, et tegelikult on meil ainult kaks reeglit: esimene ja teine reegel on sisuliselt sama reegli erijuhud (järgmisse silpi läheb täishäälikute vahel oleva kaashäälikute grupi viimane, ka siis, kui grupp koosneb ainult ühest kaashäälikust), samuti kolmas ja neljas reegel on sisuliselt üks reegel (täishäälikute jadas on eraldi silbid kaks esimest, kaks järgmist jne).

Üks võimalus on need reeglid otseselt realiseerida. Selline lahendus on failis `silbalah1.c`.

Alternatiiv on panna tähele, et nende reeglite seisukohalt on ühe sümboli ees uue silbi alustamise või mittealustamise seisukohalt olulised ainult mõned üksikud faktid sellele sümbolile eelnenud teksti kohta. Sellises olukorras on sageli tõhus ehitada lahendus üles olekumasinale ehk automaadile nagu on tehtud failis `silbalah2.pas`.

Automaadil põhinevad lahendused on kasulikud sellepoolest, et ei vaja kogu analüüsitava teksti korraga mällulugemist. Selle ülesande väikeste andmemahitude puhul pole see muidugi eriline võit, aga reaalsed tekstianalüüsid töötlevad sageli faile, mis on mällu mahtumiseks lootusetult liiga suured. Teksti pikkusest sõltumatu mäluvajadusega realisatsiooni näide on edasijõudnute rühma silbitamisülesande lahendus failis `silbelah2.pas`.

## Testid

1.  $N = 5$ . Esimene reegel. 4 punkti.
2.  $N = 1$ . Teine reegel. 4 punkti.
3.  $N = 2$ . Esimene ja teine reegel. 4 punkti.
4.  $N = 5$ . Kolmas ja neljas reegel. 4 punkti.
5.  $N = 4$ . Põhiliselt kolmas ja neljas reegel. 4 punkti.
6.  $N = 5$ . Mitmesugused sõnad, kõik reeglid. 4 punkti.
7.  $N = 14$ . Mitmesugused sõnad, kõik reeglid. 4 punkti.
8.  $N = 18$ . Mitmesugused sõnad, kõik reeglid. 4 punkti.
9.  $N = 80$ . Mitmesugused sõnad, kõik reeglid. 4 punkti.
10.  $N = 100$ . Maksimaalne, aga ebarealistlik test. 4 punkti.

Kokku 40 punkti.

## 6. Silbitamine

Selle ülesande lahendamiseks tuleb muidugi kõigepealt hoolikalt tekstis toodud reegleid lugeda. Seda tehes võib märgata, et tegelikult on meil ainult kaks reeglit: esimene ja teine reegel on sisuliselt sama reegli erijuhud (järgmisse silpi läheb täishäälikute vahel oleva kaashäälikute grupi viimane, ka siis, kui grupp koosneb ainult ühest kaashäälikust), samuti kolmas ja neljas reegel on sisuliselt üks reegel (täishäälikute jadas on eraldi silbid kaks esimest, kaks järgmist jne).

Üks võimalus on need reeglid otseselt realiseerida. Selline lahendus on failis `silbelah1.c`.

Alternatiiv on panna tähele, et nende reeglite seisukohalt on ühe sümboli ees uue silbi alustamise või mittealustamise seisukohalt olulised ainult mõned üksikud faktid sellele sümbolile eelnenud teksti kohta. Sellises olukorras on sageli tõhus ehitada lahendus üles olekumasinale ehk automaadile nagu on tehtud failis `silbelah2.pas`.

Automaadil põhinevad lahendused on kasulikud sellepoolest, et ei vaja kogu analüüsitava teksti korraga mällulugemist. Selle ülesande väikeste andmemahitude puhul pole see muidugi eriline võit, aga reaalsed tekstianalüüsid töötlevad sageli faile, mis on mällu mahutamiseks lootusetult liiga suured.

### Testid

1.  $N = 8$ . Esimene ja teine reegel. iga sõna eraldi real. 4 punkti.
2.  $N = 9$ . Kolmas ja neljas reegel, iga sõna eraldi real. 4 punkti.
3.  $N = 8$ . Kõik reeglid, mitu sõna real, eraldajad ainult tühikud. 4 punkti.
4.  $N = 10$ . Igasugused sõnad, lisaks segavad sümbolid. 4 punkti.
5.  $N = 5$ . Tavaline tekst, aga ridade lõpus tühikud. 4 punkti.
6.  $N = 6$ . Tavaline tekst, aga ridade alguses tühikud. 4 punkti.
7.  $N = 12$ . Tavaline tekst, lisaks tühjad read, neist üks faili lõpus. 4 punkti.
8.  $N = 9$ . Tavaline tekst, lisaks tühjad read, neist üks faili alguses. 4 punkti.
9.  $N = 51$ . Üsna suur realistlik test: tühikud, kirjavahemärgid, tühjad read. 4 punkti.
10.  $N = 99$ . Suur test. HTML-fail. 4 punkti.

Kokku 40 punkti.