IsoInt

Tietokoneiden muisti koostuu yksittäisistä muistisanoista, jotka nykyaikaisissa koneissa ovat 64 bitin pituisia. Muistisanan koko asettaa teknisen rajoituksen sille, kuinka suuria lukuja tietokone kykenee konekäskyillään käsittelemään. Tämä rajoitus näkyy suoraan useimmissa ohjelmointikielissä, esimerkiksi `C` ja `C++` -kielien "pitkä kokonaisluku" (`long int`) on nykyisillä koneilla yleensä 64-bittinen kokonaisluku.

Vaikka 64-bittiset luvut riittävät mainiosti useimpiin tavanomaisiin laskutehtäviin, joissakin sovelluksissa (esim. salakirjoitus) tarvitaan vielä huomattavasti suurempia lukuja, eikä lukujen suuruus ole välttämättä etukäteen tiedossa. Tällaisten sovellusten toteuttamiseen tarvitaan ns. mielivaltaisen tarkkuuden kokonaislukuja. Tehtävänäsi on suunnitella ja toteuttaa mielivaltaisen tarkkuuden kokonaisluku (talletusrakenne, perusoperaatiot). Käytämme jatkossa nimeä `IsoInt` tällaisille luvuille. Voit olettaa, että luvut eivät ole negatiivisia ts. luvun etumerkki voidaan jättää huomiotta.

Toteutettavia perusoperaatioita ovat:

- Kahden `IsoInt`:in yhteenlasku; tuloksena `IsoInt`; ja
- Kahden `IsoInt`:in kertolasku; tuloksena `IsoInt`.

Yllä kuvailtujen operaatioiden avulla on mahdollista toteuttaa erilaisia monimutkaisempia matemaattisia funktioita. Yksi tällainen on ns. *binomikerroin*, jonka arvot ovat tuttuja mm. Pascalin kolmion riveiltä. Binomikerroin on myös tärkeä käsite todennäköisyyslaskennassa; se kertoo, kuinka monta k:n kokoista (erilaista) osajoukkoa voidaan muodostaa joukosta, jossa on n alkiota. Matematiikassa binomikerrointa merkitään yleisesti $\binom{n}{k}$.

Perinteinen matemaattinen määritelmä binomikertoimen laskuun vaatii ns. kertoma-operaation ja jakolaskun toteuttamista (jakolaskun tulos on kuitenkin aina kokonaisluku):

$$\binom{n}{k} = \frac{n!}{k!(n-k)!}$$

Binomikerroin voidaan laskea pelkästään yhteenlaskun avulla käyttäen allaolevia ns. *palautuskaavoja*. Kaavoissa binomikerrointa $\binom{n}{k}$ on merkitty lähempänä ohjelmointikieliä olevalla notaatiolla $B(n,k)$.

$$\begin{aligned} B(n,0) &= 1 \\ B(0,k) &= 1 \\ B(n,k) &= B(n-1,k-1) + B(n-1,k) \end{aligned}$$

Tehtävänäsi on toteuttaa operaatio, joka laskee kahden annetun `IsoInt`:in binomikertoimen. Voit toteuttaa binomikertoimen laskun millä tavalla haluat.

Koska tehtävät tullaan tarkastamaan koneellisesti, operaatioiden käyttämiseen pitää toteuttaa komentotulkki, joka lukee laskutoimituksia oletussyötteestä (esim. `C`:ssä `stdin`) ja tulostaa laskutoimitusten tulokset oletustulokseen (`C`:ssä `stdout`). Syötteessä voi olla useita tällaisia kolmikoita peräkkäin. Kolmikoiden lukumäärä kerrotaan syötteen ensimmäisellä rivillä. Operaatiot tulee nimetä seuraavasti: yhteenlasku `Y`, kertolasku `K` ja binomikerroin `B`.

(jatkuu seuraavalla sivulla)

Esimerkki.

Oikea vastaus syötteeseen

```
3
Y
22
11
K
22
11
B
22
11
```

on

```
33
242
705432
```

Pisteytys

- Oikein toimiva yhteenlasku: 20p.
- Oikein toimiva kertolasku: 20p.
- Oikein toimiva binomikerroin: 20p
- Binomikertoimen laskennan nopeus 0–40p.

Laskennan nopeudesta saa pisteitä siten, että nopein toimitettu kilpailuvastaus saa 40p, hitain 1p ja muut vastaukset nopeutensa mukaan tältä väliltä. Laskentaan saa käyttää aikaa enintään 5 minuuttia (2.67 GHz PC) ja toteutuksessa saa käyttää muistia enintään 1 gigatavun verran. Voit olettaa, että syötteessä annetut luvut ja laskutoimitusten lopputulokset ovat enintään 10.000.000:n numeron pituisia (kymmenjärjestelmässä).

Ryhmittely

Opettaja haluaa muodostaa oppilaistaan työryhmiä. Tarkoituksena on, että ryhmän jäsenten koulumenestys (todistuksen arvosanat) olisi mahdollisimman samankaltainen. Tämän samankaltaisuuden mittaamiseen hän kehittää seuraavan *etäisyysmitan*:

- Olkoot opiskelijoiden todistuksessa esiintyvät aineet numeroitu $1..n$. Yksinkertaisuuden vuoksi oletamme, että kaikki oppilaat lukevat samoja aineita ja että ne esiintyvät todistuksissa samassa järjestyksessä. Eli aine numero i tarkoittaa samaa ainetta kaikilla oppilailla.
- Merkitään oppilaan x arvosanaa aineessa i merkinnällä $x(i)$.
- Oppilaiden x ja y *aine-etäisyys* aineessa i, $D(x, y, i)$, on arvosanojen erotuksen itseisarvo. Jos $x(i) \geq y(i)$, niin $D(x, y, i) = x(i) - y(i)$; jos taas $y(i) > x(i)$, niin $D(x, y, i) = y(i) - x(i)$.
- Oppilaiden x ja y *etäisyys*, $D(x, y)$, on x:n ja y:n kaikkien aine-etäisyyksien summa. Luku $D(x, y)$ saadaan siis selville laskemalla yhteen arvot $D(x, y, i)$ kaikille $i = 1, ..., n$.

Ryhmittely voidaan toteuttaa monilla eri tavoilla. Esimerkiksi, voidaan päättää ensin, että ryhmiä on tietty määrä (esim. 5), jonka jälkeen oppilaat sijoitetaan jossakin järjestyksessä näihin ryhmiin (esim. 1. sijoitetaan ryhmään 1, 2. ryhmään 2, ..., 5. ryhmään 5, 6. ryhmään 1 jne.). Koska tavoitteena on kuitenkin muodostaa "hyviä ryhmiä", olisi suotavaa, että edellä kuvailtu $D(x, y)$ olisi mahdollisimman pieni ryhmän jäsenten kesken. Tähän tarkoitukseen tarvitaan jokin ryhmien laatua kuvaava mittari.

Olkoon G ryhmä (joukko) oppilaita ja ryhmän oppilaat $x_1, \ldots, x_m$. Ryhmän G *laatu* $L(G)$ on sen jäsenten parittaisten etäisyyksien summa. Tämä luku voidaan laskea määrittämällä ryhmän oppilaiden x_i ja x_j väliset etäisyydet kaikille indeksipareille i, j joissa $i < j$ ja laskemalla saadut etäisyydet yhteen.

Esimerkiksi seuraava `C`-kielinen ohjelmanpätkä laskee ryhmän `G` laadun muuttujaan `sum` (oletettu, että `G` on toteutettu taulukkona opiskelijoita ja että ryhmän koko `m` on tiedossa):

```
int i, j;
int sum = 0;
for (i = 0; i < m - 1; i++)
    for (j = i+1; j < m; j++)
        sum += D(G[i], G[j]);
```

Kun osataan määrätä yksittäisen ryhmän G laatu, sen avulla voidaan määrätä koko ryhmittelyn laatu ryhmien laadun summana. *Ryhmittely* on mikä tahansa oppilaiden sijoittelu ryhmiin siten, että

- jokaisessa ryhmässä on vähintään yksi oppilas;
- jokainen oppilas on sijoitettu johonkin ryhmään; ja
- mikään oppilas ei kuulu useampaan kuin yhteen ryhmään.

Opettaja on jo tyytyväinen hienoon keksintöönsä, kunnes huomaa, että hänen upea mittarinsa antaa parhaan laadun sellaiselle ryhmittelylle, jossa jokainen oppilas on yksin omassa ryhmässään. Tämä ei ollut tarkoitus. Estääkseen tällaisen ilmiön hän päättääkin, että jokaisessa ryhmässä pitää olla vähintään tietty määrä jäseniä (esim. 4).

Tehtävänäsi on kehittää ja toteuttaa menetelmä, joka määrää mahdollisimman laadukkaan ryhmittelyn eli sellaisen, jolle laatumittari antaa mahdollisimman pienen arvon ja jossa jokaisessa ryhmässä on vähintään k jäsentä. Huomaa, että sinun tulee päättää myös ryhmien lukumäärä.

Tehtävä annetaan tekstitiedostona siten, että tiedoston ensimmäisellä rivillä on opiskelijoiden lukumäärä, toisella rivillä aineiden lukumäärä ja kolmannella rivillä ryhmän minimikoko. Näitä lukuja seuraavat kunkin oppilaan arvosanat, jotka ovat omilla riveillään välilyönneillä eroteltuina. Ohjelmasi pitää lukea tällainen tiedosto oletussyötteestä.

Laadittu ryhmittely kirjoitetaan oletustulokseen seuraavassa muodossa

- ensimmäiselle riville tulostetaan ryhmien lukumäärä; ja
- seuraaville riveille tulostetaan ryhmien jäsenten numerot (järjestysnumerot syötteessä) pilkulla eroteltuna.

Voit olettaa, että oppilaita on enintään 100000, aineita on enintään 30 kpl, ja että arvosanat ovat kokonaislukuja 5,6,7,8,9 ja 10.

Esimerkki.

Olkoon syöttötiedosto seuraava:

```
8
10
4
5 5 6 6 7 7 8 8 9 9
6 6 7 7 8 8 9 9 5 5
7 7 8 8 9 9 5 5 6 6
8 8 9 9 5 5 6 6 7 7
9 9 5 5 6 6 7 7 8 8
5 5 5 5 5 5 5 5 5 5
8 8 8 8 8 6 6 6 6 6
9 9 9 9 9 9 9 9 9 9
```

Jatkossa oppilaisiin viitataan heidän järjestysnumeroillaan ao. tiedostossa.

Oppilaiden 1 ja 2 etäisyys eli $D(1,2)$ on

$$(6-5)+(6-5)+(7-6)+\cdots+(9-5)=16.$$

Jos oppilaat 1,2,3 ja 4 sijoitetaan samaan ryhmään, on muodostetun ryhmän laatu

$$\begin{aligned} &D(1,2)+D(1,3)+D(1,4)+D(2,3)+D(2,4)+D(3,4) &&= \\ &16+24+24+16+24+16 &&= \quad 120 \end{aligned}$$

Vastaavasti toisen ryhmän laatu (jossa oppilaat 5, 6, 7 ja 8) on $20+16+20+20+40+20 = 136$ ja koko tällaisen ryhmittelyn laatu on $120 + 136 = 256$.

Jos oppilaat ryhmitellään "parittomiin ja parillisiin" eli ryhmiin 1,3,5,7 ja 2,4,6,8, on tällaisen ryhmittelyn laatu $110 + 144 = 254$. Jälkimmäinen ryhmittely on siis näistä kahdesta parempi.

Ensimmäisen esimerkkitapauksen mukainen ryhmittely tulostettaisiin muodossa

```
2
1,2,3,4
5,6,7,8
```

Pisteytys.

- Toimiva ohjelma 20p.
- Ryhmittelyn laatu 80p.

Ryhmittelyn laatu pisteytetään siten, että paras toimitettu kilpailuvastaus saa 80p, heikoin 1p ja muut vastaukset ryhmittelyn laadun mukaan tältä väliltä.

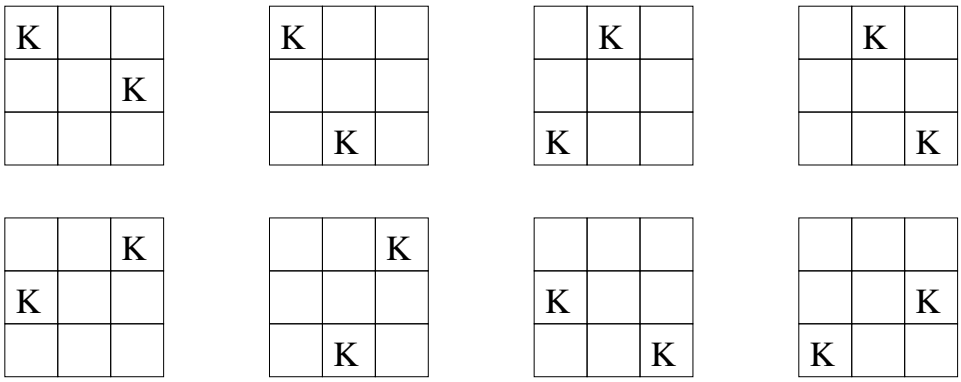
Laskentaan saa käyttää aikaa enintään yhden minuutin (2.67 GHz PC) ja toteutuksessa saa käyttää muistia enintään 1 gigatavun verran. Aikarajan seuraamiseksi ajan kulumista kannattaa tarkkailla ratkaisujen etsimisen yhteydessä ja ajan loppuessa palauttaa paras löydetty tulos.

Huomautus. Etenkin suurille tehtäville ei ole mahdollista löytää parasta mahdollista (optimaalista) ryhmittelyä annetussa ajassa. Tämä johtuu puhtaasti matemaattisista syistä; asiasta innostuneet voivat miettia erilaisten ryhmittelyjen lukumäärää. Tärkeintä onkin etsiä niin hyvä ratkaisu kuin annetussa ajassa on mahdollista käyttämällä mitä hyvänsä mieleen tulevia keinoja hyvien ratkaisujen etsimiseen.

Kuningattaret

Shakissa kuningatar voi liikkua vaaka-, pysty- tai viistosuuntaisesti. Tehtävänäsi on laskea, kuinka monella tavalla kaksi kuningatarta voidaan sijoittaa $n \times n$ -shakkilaudalle niin, että ne eivät uhkaa toisiaan.

Esimerkiksi tapauksessa $n = 3$ mahdollisuudet ovat:



Tapauksessa $n = 10$ mahdollisuuksia on jo 3480.

Ohjelmasi tulee lukea standardisyötteestä yksi kokonaisluku: shakkilaudan koko n. Tämän jälkeen ohjelmasi tulee kirjoittaa standarditulostukseen yksi kokonaisluku: kuningatarten sijoitustapojen määrä.

Esimerkkisyöte 1:

```
3
```

Esimerkkitulostus 1:

```
8
```

Esimerkkisyöte 2:

```
10
```

Esimerkkitulostus 2:

```
3480
```

Arvostelussa testataan sekä ohjelman toimivuutta että tehokkuutta. Ohjelman toimivuutta testataan 60 pisteen arvoisilla testeillä, joissa n on välillä 1–50. Ohjelman tehokkuutta testataan 40 pisteen arvoisilla testeillä, joissa n on välillä 1–100000. Yhteensä ohjelmasi voi saada siis 100 pistettä. Ohjelmasi käytössä on 1 sekunti suoritusaikaa (2.67 GHz PC) ja 1 gigatavu muistia.