# 2+2

## Definīcija

Viena no visvieglākajām problēmām starp sporta programmēšanas problēmām ir divu skaitļu saskaitīšana. Ievaddatos tiek doti divi skaitļi, uzraksti programmu, kas saskaita tos.

## Ievaddatu raksturojums

Vienā rindā tiek doti 2 naturāli skaitļi intervālā [1, 100].

## Izvaddatu raksturojums

Izvadīt šo divu skaitļu summu.

#include <iostream>

using namespace std;

int main() {

int x, y;

cin >> x >> y;

cout << x+y;

}

#### ****Paraugdati****

##### **Stdin**

2 2

##### **Stdout**

4

# Reizināšana

## Definīcija

Šis ir ļoti vienkāršs uzdevums. Jūsu programmai ir jānolasa skaitlis N un jāizvada šis skaitlis pareizināts ar 2.

## Ievaddatu raksturojums

Ievaddati satur 1 skaitli N (1 <= N <= 32000)

## Izvaddatu raksturojums

Jaizvada skaitlis 2 \* N

#include <iostream>

using namespace std;

int main() {

int N;

cin >> N;

cout << N\*2;

}

#### ****Paraugdati****

##### **Stdin**

5

##### **Stdout**

10

# CipVirkne

## Definīcija

Ciparu virknē a1,a2,a3,.... katrs loceklis sākot no trešā ir vienāds ar iepriekšējo divu virknes locekļu summas pēdējo ciparu.  
Uzrakstiet programmu, kas dotiem a1,a2 un n atrod an.

## Ievaddatu raksturojums

Ievaddatu vienīgajā rindā dotas trīs veselu skaitļu a1,a2 un n vērtības.  
Zināms, ka 0<=a1<=9; 0<=a2<=9; 0<n<32768.  
Starp katriem diviem blakus skaitļiem ir viens tukšumsimbols.

## Izvaddatu raksturojums

*Izvaddatu* vienīgajā rindā jāizvada vesels skaitlis - an vērtība.

#include <iostream>

using namespace std;

int main() {

long a1, a2, an, pag;

cin >> a1 >> a2 >> an;

long arr[an];

arr[0] = a1;

arr[1] = a2;

for (int i=2;i<an;i++){

pag=(arr[i-2] + arr[i-1]);

arr[i] = pag%10;

}

cout << arr[an-1] << endl;

}

#### ****Paraugdati****

##### **Stdin**

1 7 12

##### **Stdout**

8

# Tomāti

## Definīcija

Erna nodarbojas ar tomātu audzēšanu un ir salikusi uz palodzes rindā n jaunās ražas tomātus. Viens no tomātiem ir sarkans, bet pārējie - zaļi. Ir novērots, ka katrs zaļais tomāts, kas stāvējis blakus sarkanajam, nākošajā dienā kļūst sarkans. Citi tomāti savu krāsu nemaina. Piemēram, ja pavisam ir deviņi tomāti un ceturtais no kreisās puses ir sarkans, tad pēc divām dienām būs pieci sarkani tomāti.

Uzrakstiet programmu, kas dotam tomātu skaitam, sarkanā tomāta novietojumam un dienu skaitam nosaka, cik sarkano tomātu būs pēc norādīto dienu skaita! Tomātus nogatavošanās laikā pārkārtot nedrīkst.

## Ievaddatu raksturojums

Ievaddatos dotas trīs naturālu skaitļu n, s un d vērtības, kas atdalītas ar tukšumsimboliem. n ir kopējais tomātu skaits, s apzīmē sarkanā tomāta atrašanās vietu - kurš tas ir pēc kārtas, skaitot no kreisās puses, d apzīmē dienu skaitu, pēc kāda nepieciešams uzzināt sarkano tomātu skaitu. Zināms, ka n≤109,s≤n,d≤109.

## Izvaddatu raksturojums

Jāizvada viens naturāls skaitlis - sarkano tomātu skaits pēc d dienām.

#include <iostream>

using namespace std;

int main() {

int n, s, d, x=1;

cin >> n >> s >> d;

if (n-s >=d) x=x+d;

else x=x+(n-s);

if (s-1>=d) x=x+d;

else x = x+(s-1);

cout << x;

}

#### ****Paraugdati****

##### **Stdin**

9 4 2

##### **Stdout**

5

# Virsotne

## Definīcija

Veselu skaitļu piramīdu veido sekojoši. Vispirms rindā uzraksta divus vai vairāk veselus skaitļus. Šo skaitļu rindu sauksim par piramīdas *pamatu*. Pēc tam aprēķina katru divu blakus stāvošu skaitļu summu un uzraksta šo summu vienu rindu augstāk tā, ka tā atrodas starp šo summu veidojošajiem skaitļiem. Šo darbību turpina tik ilgi, kamēr iegūts viens, visaugstāk esošais skaitlis. Teiksim, ka šis skaitlis atrodas piramīdas *virsotnē*. Uzrakstiet programmu, kas dotiem skaitļiem piramīdas pamatā nosaka, kāds skaitlis atrodas piramīdas virsotnē!

## Ievaddatu raksturojums

Ievaddatu pirmajā rindā dots naturāls skaitlis N(1<N<100) - piramīdas pamatā esošo skaitļu skaits. Faila otrajā rindā doti N veseli skaitļi, kas atdalīti ar tukšumsimboliem. Zināms, ka neviena skaitļa vērtība piramīdā pēc moduļa nepārsniedz 32767.

## Izvaddatu raksturojums

Jāizvada viens vesels skaitlis - piramīdas virsotnē esošais skaitlis.

#include <iostream>

using namespace std;

int main() {

int baseAmount; cin >> baseAmount; int num = baseAmount;

for(int i=1;i<baseAmount;i++) num = num + baseAmount-i;

int base[num];

for (int i=0;i<baseAmount;i++) cin >> base[i];

int layerSK = baseAmount, baseAmount2 = baseAmount, x=1, j=0;

for (int i=0;i<num-2;i++) if (i==layerSK-2){ i++;

layerSK = layerSK + baseAmount-x; x++;

baseAmount2--; base[baseAmount2+i] = base[j] + base[j+1];

j=j+2;

}

else{ base[baseAmount2+i] = base[j] + base[j+1]; j++; }

int lastElement = base[num-1];

cout << lastElement; }

#### ****Paraugdati****

##### **Stdin**

4

-3 2 8 -7

##### **Stdout**

20

# Burtu klucīši

**Definīcija**

Rotai ir klucīši ar burtu attēliem. Uz katra klucīša ir uzzīmēts viens latīņu alfabēta lielais burts. Uz vairākiem klucīšiem var būt uzzīmēts viens un tas pats burts. Rota no saviem klucīšiem, saliekot tos rindā vienu aiz otra, ir izveidojusi kādu vārdu. Rotas draugs Zigmārs arī vēlētos salikt kādu vārdu, diemžēl viņam nav šādu klucīšu. Rota ir ar mieru aizdot tikai tos klucīšus, no kuriem ir salikts viņas iedomātais vārds, un nevienu vairāk.  
Uzrakstiet programmu, kas dotiem Rotas saliktā un Zigmāra iedomātajiem vārdiem nosaka, vai Zigmārs no dotajiem klucīšiem varēs salikt savu iedomāto vārdu.

**Ievaddatu raksturojums**

Ievaddatu pirmajā rindā ir dota simbolu virkne - Rotas saliktais vārds. Faila otrajā rindā ir dota otra simbolu virkne - Zigmāra iedomātais vārds. Katrā virknē ir tikai latīņu alfabēta lielie burti. Katrā virknē ir vismaz viens un ne vairāk kā 250 simboli.

**Izvaddatu raksturojums**

Izvaddatu pirmajā rindā izvadiet vārdu VAR, ja Zigmārs no dotajiem klucīšiem var salikt savu iedomāto vārdu vai NEVAR, ja to izdarīt nav iespējams.

#include <algorithm>

#include <iostream>

#include <vector>

using namespace std;

int main() {

string v1, v2; cin >> v1 >> v2;

bool itCan = true;

vector<char> V1, V2;

for (int i = 0; i < v1.length(); i++) V1.push\_back(v1.at(i));

for (int i = 0; i < v2.length(); i++) V2.push\_back(v2.at(i));

for (int i = 0; i < V2.size(); i++) { if (std::find(V1.begin(), V1.end(), V2.at(i)) == V1.end()) {

itCan = false;

} else { std::vector<char>::iterator position = std::find(V1.begin(), V1.end(), V2.at(i));

if (position != V1.end()) V1.erase(position); } }

if (itCan == true) cout << " VAR";

else cout << " NEVAR"; }

#### ****Paraugdati****

##### **Stdin**

ALEKSANDRS

SANDRA

##### **Stdout**

VAR

**Trīs skaitļi**

#### ****Definīcija****

Uzrakstīt programmu, kas nosaka, vai starp ievadītajiem trijiem veseliem skaitļiem var izvēlēties divus tā, lai to summa būtu pozitīvs/negatīvs skaitlis vai nulle.

#### ****Ievaddatu raksturojums****

Ievaddatu vienīgajā rindā dotas trīs veselu skaitļu vērtības. Zināms, ka visi skaitļi ir robežās no -2147483648 līdz 2147483647.

#### ****Izvaddatu raksturojums****

Izvaddatiem jāsatur tieši trīs rindas: pirmajā rindā jāizvada vārds "VAR", ja no trim ievadītajiem skaitļiem iespējams izvēlēties divus tā, lai to summa būtu pozitīvs skaitlis un "NEVAR", ja tas nav iespējams. Faila otrajā rindā jāizvada vārds "VAR", ja no trim ievadītajiem skaitļiem iespējams izvēlēties divus tā, lai to summa būtu nulle un "NEVAR", ja tas nav iespējams. Faila trešajā rindā jāizvada vārds "VAR", ja no trim ievadītajiem skaitļiem iespējams izvēlēties divus tā, lai to summa būtu negatīvs skaitlis un "NEVAR", ja tas nav iespējams.

#include <iostream>

using namespace std;

int main() {

long x, y, z; cin >> x >> y >> z;

if(x+y > 0) cout << "VAR" << endl;

else if(x+z > 0) cout << "VAR" << endl;

else if(y+z > 0) cout << "VAR" << endl;

else cout << "NEVAR" << endl;

if(x+y == 0) cout << "VAR" << endl;

else if(x+z == 0) cout << "VAR" << endl;

else if(y+z == 0) cout << "VAR" << endl;

else cout << "NEVAR" << endl;

if(x+y < 0) cout << "VAR" << endl;

else if(x+z < 0) cout << "VAR" << endl;

else if(y+z < 0) cout << "VAR" << endl;

else cout << "NEVAR" << endl;

}

#### ****Paraugdati****

##### **Stdin**

7 -3 -4

##### **Stdout**

VAR

NEVAR

VAR

# **Nauda**

#### ****Definīcija****

Kādā valstī līdz nesenam laikam apgrozībā esošās naudas vienības bija mārciņas, šiliņi un pensi. Šīs naudas vienības savā starpā saistīja sekojošas sakarības: 1 mārciņa = 20 šiliņi; 1 šiliņš = 12 pensi

Tā kā šāda naudas sistēma bija grūti saskaņojama ar citu valstu naudas sistēmām, tad tika nolemts pāriet uz jaunu sistēmu, kurā būtu tikai mārciņas un pensi (sauksim tos par jaunajām mārciņām un jaunajiem pensiem), pie kam būtu spēkā sakarība 1 jaunā mārciņa = 100 jaunie pensi. Tika nolemts, ka pensam sava vērtība jāsaglabā (1 jaunais penss = 1 penss).

Uzrakstiet programmu, kas dotai naudas summai vecajā sistēmā aprēķina atbilstošu naudas summu jaunajā sistēmā.

#### ****Ievaddatu raksturojums****

Pirmajā rindā dotas trīs nenegatīvu veselu skaitļu **n1**, **n2** un **n3** vērtības, kas attiecīgi apzīmē kādas naudas summas mārciņu, šiliņu un pensu skaitu vecajā naudas skaitīšanas sistēmā. Zināms, ka dotās naudas daudzums nepārsniedza 100 "vecās" mārciņas. Starp katriem diviem blakus skaitļiem ir viens tukšumsimbols.

#### ****Izvaddatu raksturojums****

Pirmajā rindā jāizvada ievadītā naudas summa jaunajā naudas skaitīšanas sistēmā kā divi veseli skaitļi - jauno mārciņu un jauno pensu skaits. Skaitļiem jābūt atdalītiem ar tukšumsimbolu. Summa jāpārveido tā, lai jauno pensu skaits būtu robežās no 0 līdz 99.

#include <iostrea>

using namespace std;

int main() {

int m, s, p, M, P; cin >> m >> s >> p;

s = s + (m\*20);

P = p + (s\*12);

M = P/100;

P = P%100;

cout << M << " " << P;

}

#### ****Paraugdati****

##### **Stdin**

3 100 1000

##### **Stdout**

29 20

# Liekāki par videjo

**Definīcija**

Ievadītiem N naturāliem skaitļiem noteikt cik no šiem skaitļiem ir lielāki par visu ievadīto skaitļu aritmētisko vidējo. Piemēram, ja ievadīti skaitļi 3, 5 un 4, tad tikai viens no šiem skaitļiem (5) ir lielāks par visu skaitļu aritmētisko vidējo (4).

**Ievaddatu raksturojums**

Pirmajā rindā dota naturāla skaitļa N(1≤N≤20) vērtība. Katrā no nākošajām N faila rindām dots pa vienam naturālam skaitlim. Neviena skaitļa vērtība nepārsniedz 250.

**Izvaddatu raksturojums**

Vienīgajā rindā jāizvada viens vesels nenegatīvs skaitlis - to ievadīto skaitļu skaits, kas ir lielāki par visu ievadīto skaitļu aritmētisko vidējo.

#include <iostream>

#include <algorithm>

using namespace std;

int main() {

long n, vid=0, sk=0; cin >> n;

long arr[n];

for (int i=0;i<n;i++){ cin >> arr[i]; vid=vid+arr[i]; }

vid = vid/n;

for (int i=0;i<n;i++){ if (arr[i]>vid) sk++; }

cout << sk;

}

#### ****Paraugdati****

##### **Stdin**

3

5

4

3

##### **Stdout**

1

# Sakarto 3

**Definīcija**

Uzrakstiet programmu, kas dotos trīs veselos skaitļus izvada nedilstošā secībā!

**Ievaddatu raksturojums**

Vienā rindā ir doti trīs veseli skaitļi. Katra skaitļa vērtība ir robežās no -2147483648 līdz 2147483647 (ieskaitot). Starp katriem diviem blakus skaitļiem ievaddatos ir viena tukšumzīme.

**Izvaddatu raksturojums**

Jāizvada trīs veseli skaitļi - ievadītie skaitļi sakārtoti nedilstošā secībā. Starp katriem diviem blakus skaitļiem izvaddatos jābūt vienai tukšumzīmei.

#include <algorithm>

#include <iostream>

using namespace std;

int main() {

int arr[3];

for (int i=0;i<3;i++) cin >> arr[i];

int i, j;

for (i = 0; i < 3 - 1; i++)

for (j = 0; j < 3 - i - 1; j++)

if ((arr[j]) > arr[j + 1])

swap(arr[j], arr[j + 1]);

for (int i=0;i<3;i++) cout << arr[i] << ' ';

}

#### ****Paraugdati****

##### **Stdin**

1 2 1

##### **Stdout**

1 1 2

# Komats

**Definīcija**

Bieži datorā, ievadot reālu skaitli kā decimālo atdalītāju, var izmantot vai nu tikai punktu, vai tikai komatu.  
Jūsu uzdevums: dots reāls skaitlis n, kuram kā decimālais atdalītājs izmantots vai nu punkts, vai komats. Izvadiet šo pašu skaitli, kā decimālo atdalītāju izmantojot komatu. Citādi skaitlim jābūt tieši tādā pašā formātā, kā tas dots ievaddatos.

**Ievaddatu raksturojums**

Vienīgajā rindā dots reāls skaitlis n, kura pierakstā nav vairāk par 100 cipariem.

**Izvaddatu raksturojums**

Izvadīt n, kā decimālo atdalītāju izmantojot komatu.

#include <iostream>

#include <algorithm>

using namespace std;

int main() {

string n;

cin >> n;

replace( n.begin(), n.end(), '.', ',');

cout << n;

}

#### ****Paraugdati****

##### **Stdin**

3.14

##### **Stdout**

3,14

# Atrodi mazāko skaitli

**Definīcija**

Atrodi mazāko skaitli no dotajiem.

**Ievaddatu raksturojums**

Ievaddatu pirmajā rindā dots skaitļu skaits 0>n<50 . Atlikušajās n rindās doti veseli skaitļi no -100 līdz 100.

**Izvaddatu raksturojums**

Izvaddatu vienīgajā rindā jāizvada mazākais no dotajiem skaitļiem.

#include <iostream>

using namespace std;

int main() {

int n; cin >> n;

int masivs[n];

for (int i=0;i<n;i++) cin >> masivs[i];

for (int i = 0; i < n - 1; i++)

for (int j = 0; j < n - i - 1; j++)

if (masivs[j] > masivs[j + 1])

swap(masivs[j], masivs[j + 1]);

cout << masivs[0];

}

#### ****Paraugdati****

##### **Stdin**

3

1

2

3

##### **Stdout**

1

# Saskaiti pāra skaitļus

**Definīcija**

Programmai jāatrod pāra skaitļu skaits dotajā veselu skaitļu sarakstā.

**Ievaddatu raksturojums**

Ievaddatu pirmajā rindā dots skaitļu skaits 0<n<50. Visās atlikušajās rindās doti veseli skaitļi no 1 līdz 100.

**Izvaddatu raksturojums**

Vienīgajā rindā jāizvada pāra skaitļu skaits.

#include <iostream>

using namespace std;

int main() {

int n, sk=0; cin >> n;

int cip[n];

for(int i=0;i<n;i++){

cin >> cip[i];

if(cip[i]%2 == 0)

sk++;

}

cout << sk;

}

#### ****Paraugdati****

##### **Stdin**

4

3

2

3

4

##### **Stdout**

2

# Pozitivo skaitļu summa

**Definīcija**

Programmai jāatrod pozitīvo skaitļu summa no skaitļu saraksta.

**Ievaddatu raksturojums**

Pirmajā ievaddatu rindā dots skaitļu skaits 0<n<50. Atlikušajās rindās doti veseli skaitļi no -100 līdz 100.

**Izvaddatu raksturojums**

Vienīgajā rindā jāizvada visu pozitīvo skaitļu summa.

#include <iostream>

#include <algorithm>

using namespace std;

int main() {

long n, vid=0; cin >> n;

long arr[n];

for (int i=0;i<n;i++){

cin >> arr[i];

if(arr[i]>0)

vid = vid+arr[i];

}

cout << vid;

}

#### ****Paraugdati****

##### **Stdin**

3

-3

1

2

##### **Stdout**

3

# Cikls1 iesacejiem

**Definīcija**

Jāizdrukā vārds CIKLS noteiktu reižu skaitu!

**Ievaddatu raksturojums**

Ievaddatu pirmajā rindā ierakstīts vesels skaitlis a ( 0<a<32000).

**Izvaddatu raksturojums**

Stabiņā viens zem otra jāizvada vārds CIKLS a reizes.

#include <iostream>

using namespace std;

int main() {

int a; cin >> a;

for(int i=0;i<a;i++) cout << "CIKLS\n";

}

#### ****Paraugdati****

##### **Stdin**

3

##### **Stdout**

CIKLS

CIKLS

CIKLS

# Cikls 2 iesacējiem

**Definīcija**

Ielasi skaitli un vārdu. Izvadi nolasīto vārdu tik reizes, cik liels ir nolasītais skaitlis.

**Ievaddatu raksturojums**

Ievaddatu pirmajā rindā ierakstīts vesels skaitlis a (1<a<32000), bet otrajā rindā vārds.

**Izvaddatu raksturojums**

Stabiņā viens zem otra jāizvada nolasītais vārds a reizes.

#include <iostream>

using namespace std;

int main() {

int sk;

string vards;

cin >> sk;

cin >> vards;

for(int i=0;i<sk;i++){

cout << vards + "\n";

}

}

#### ****Paraugdati****

##### **Stdin**

4

desa

##### **Stdout**

desa

desa

desa

desa

# Cikls3 iesācējiem

**Definīcija**

Atrodi cik reižu meklējamais skaitlis sakrīt ar kādu no dotajiem skaitļiem!

**Ievaddatu raksturojums**

Ievaddatu pirmajā rindā ierakstīts vesels skaitlis a (1<a<32000), kas norāda skaitļu skaitu un meklējamais skaitlis b (1<b<32000). Sākot ar otro rindiņu, doti vēl a skaitļi.

**Izvaddatu raksturojums**

Jāizvada cik reižu meklējamais skaitlis b atrodas skaitļos no otrās rindiņas līdz beigām.

#include <iostream>

#include <set>

using namespace std;

int main() {

int b, c, d=0;

cin >> c >> b;

int a[c];

for (int i=0;i<c;i++){

cin >> a[i];

if (a[i]==b)

d++;

}

cout << d;

}

#### ****Paraugdati****

##### **Stdin**

5 2

3

4

2

2

2

##### **Stdout**

3

# Sazarojums1

**Definīcija**

Atrodi lielāko no 2 skaitļiem.

**Ievaddatu raksturojums**

Ievaddatu pirmajā rindā ierakstīti 2 veseli skaitļi a un b ( -100000<a<100000, -100000<b<100000).

**Izvaddatu raksturojums**

 Jāizvada lielākais skaitlis. Vienādu skaitļu gadījumā jāizvada jebkurš no skaitļiem.

#include <iostream>

#include <set>

using namespace std;

int main() {

int a, b;

cin >> a >> b;

if (a>b)

cout << a;

if (a<b)

cout << b;

if (a==b)

cout << a;

}

#### ****Paraugdati****

##### **Stdin**

5 -4

##### **Stdout**

5

# Sazarojums2

**Definīcija**

Atrodi lielāko no 3 skaitļiem.

**Ievaddatu raksturojums**

Ievaddatu pirmajā rindā ierakstīti 3 veseli skaitļi a, b un c ( -100000<a<100000, -100000<b<100000, -100000<c<100000).

**Izvaddatu raksturojums**

Jāizvada lielākais no 3 skaitļiem. Vienādu lielāko skaitļu gadījumā jāizvada jebkurš no lielākajiem skaitļiem.

#include <iostream>

using namespace std;

int main() {

int a, b, c; cin >> a >> b >> c;

if (a>=b)

if(a>=c)

cout << a;

else

cout << c;

else{

if (a<=b)

if(b>=c)

cout << b;

else

cout << c;

}

#### ****Paraugdati****

##### **Stdin**

3 -4 6

##### **Stdout**

6

# Spēļu skaits

**Definīcija**

Jebkurās sacensībās, kur sportisti sacenšas katrs ar katru var izveidot tabulu rezultātu pierakstīšanai. Aprēķināt turnīra spēļu skaitu, ja dots dalībnieku skaits un dalībnieki sacenšas katrs ar katru vienu reizi.

**Ievaddatu raksturojums**

Izvaddatu pirmajā rindā ierakstīts dalībnieku skaits a (2<=a<=1000).

**Izvaddatu raksturojums**

Jāizvada spēļu skaits.

#include <algorithm>

#include <iostream>

using namespace std;

int main() {

int s, speles=0; cin >> s;

for(int i=1;i<s;i++) speles = speles+I;

cout << speles;

}

#### ****Paraugdati****

##### **Stdin**

3

##### **Stdout**

3

# Mazais faktorials

**Definīcija**

Atrodi dotā skaitļa faktoriālu. Skaitļa n faktoriāls ir visu naturālo skaitļu reizinājums no 1 līdz n. Piemēram, skaitļa 4 faktoriāls ir 24 (1\*2\*3\*4). Faktoriālu matemātikā parasti apzīmē pieliekot skaitlim galā izsaukuma zīmi. Piemēram, 4!

**Ievaddatu raksturojums**

Faila pirmajā rindā ierakstīts naturāls skaitlis n ( 1<=n<=15).

**Izvaddatu raksturojums**

Jāizvada skaitļa n faktoriāls.

#include <iostream>

#include <set>

using namespace std;

int main() {

int x, z=1;

cin >> x;

for (int i=2;i<=x;i++){

z = z\*i;

}

cout << z;

}

#### ****Paraugdati****

##### **Stdin**

3

##### **Stdout**

6

# Ingusa dzimšanasdiena

**Definīcija**

Ingusam ir dzimšanas diena. Viņam paliek N gadi. Pēc tradīcijas uz viņa kūkas jābūt tik svecītēm, cik viņam ir gadu.

**Ievaddatu raksturojums**

Dots 1 <= N <= 100.

**Izvaddatu raksturojums**

Kūkas formāts ir sekojošs.

Svecīte:

$  
|

Pamatne sastāv no 3 biskvītu slāņiem.

-----  
~~~~~  
-----

Starp svecītēm ir atstarpe.

Zem katras svecītes un atstarpes ir jābūt biskvītam.

Praktisks piemērs ir redzams izvaddatos.

#include <iostream>

using namespace std;

int main() {

int n; cin >> n;

for (int i=0;i<n;i++) cout << "$ ";

cout << endl;

for (int i=0;i<n;i++) cout << "| ";

cout << endl; cout << "-";

for (int i=0;i<n-1;i++) cout << "--";

cout << endl; cout << "~";

for (int i=0;i<n-1;i++) cout << "~~";

cout << endl; cout << "-";

for (int i=0;i<n-1;i++) cout << "--";

}

##### **Stdin**

3

##### **Stdout**

$ $ $

| | |

-----

~~~~~

-----

# Fibonači virknes elementi

**Definīcija**

Matemātikā par Fibonači skaitļiem sauc virknes 1, 1, 2, 3, 5, 8, 13, 21, 34, 55 … elementus. Tās pirmie divi locekļi ir vienādi ar 1, bet katru nākamo locekli iegūst saskaitot divus iepriekšējos. Uzraksti programmu, kas atrod šīs virknes n-to elementu.

**Ievaddatu raksturojums**

Ievadatu pirmajā rindā ir dots virknes elementa kārtas numurs n (1<=n<=45).

**Izvaddatu raksturojums**

Izvaddatu pirmajā rindā jāizvada atbilstošais Fibonači virknes skaitlis.

#include <algorithm>

#include <iostream>

using namespace std;

int main() {

int n; cin >> n;

long c1 = 1, c2 = 1, pag = 0;

if(n<3) switch(n){

case 1: pag = c1; break;

case 2: pag = c2; break;

}

else for (int i = 0; i < n - 2; i++) {

pag = c1 + c2;

c1 = c2;

c2 = pag;

}

cout << pag;

}

#### ****Paraugdati****

##### **Stdin**

10

##### **Stdout**

55

# Iniciāļu izdruka

**Definīcija**

Doti vārds un uzvārds, kas ierakstīti vienā rindiņā un atdalīti ar pauzi. Izdrukāt vārda un uzvārda pirmos burtus, liekot aiz katra burta punktu.

**Ievaddatu raksturojums**

Ievaddatu pirmajā rindā ierakstīts vārds un uzvārds. Vārda un uzvārda garums nepārsniedz 20 simbolus. Tie sākas ar lielo burtu un starp tiem ir viena atstarpe.

**Izvaddatu raksturojums**

Izvaddatu pirmajā rindā jāizvada vārda un uzvārda pirmie burti, liekot aiz katra burta punktu.

#include <iostream>

#include <set>

using namespace std;

int main() {

string a, b, c;

cin >> a >> b;

char a1, b1;

a1 = a.at(0);

b1 = b.at(0);

cout << (char) toupper(a1);

cout << '.';

cout << (char) toupper(b1);

cout << '.';

}

#### ****Paraugdati****

##### **Stdin**

Aina Ziema

##### **Stdout**

A.Z.

# Vārds no otrās puses

**Definīcija**

Uzdevums uzrakstīt doto vārdu no otras puses.

**Ievaddatu raksturojums**

Ievaddatu vienīgajā rindā dots vārds.

**Izvaddatu raksturojums**

Izvaddatu vienīgajā rindā jāizvada dotais vārds, uzrakstīts no otras puses.

#include <iostream>

#include <set>

using namespace std;

int main() {

string x, z;

cin >> x;

int sk = x.length();

for (int i=sk-1;i>=0;i--){

z = z + x.at(i);

}

cout << z;

}

#### ****Paraugdati****

##### **Stdin**

kartupelis

##### **Stdout**

sileputrak

# A burtu skaits vārdā

**Definīcija**

Uzdevums sameklēt burtu 'a' skaitu dotajā vārdā.

**Ievaddatu raksturojums**

Ievaddatu vienīgajā rindā dots vārds. Vārda maksimālais burtu skaits ir 100.

**Izvaddatu raksturojums**

Izvaddatu vienīgajā rindā jāizvada viens skaitlis, kas norāda cik reižu šajā vārdā ir atkārtojies burts 'a'?

#include <iostream>

#include <set>

using namespace std;

int main() {

string vards, a;

int sk=0;

cin >> vards;

for (int i=0;i<vards.length();i++){

a = vards.at(i);

if (a.compare("a") == 0)

{sk++;}

}

cout << sk;

}

#### ****Paraugdati****

##### **Stdin**

abra

##### **Stdout**

2

# Virkne10

#### ****Definīcija****

Dota desmit skaitļu virkne. Virknes elementi ir veseli skaitļi robežās no 0 līdz 100. Uzdevums atrast:

* pirmā un pēdējā elementa summu;
* elementu skaitu, kas lielāki par pirmo elementu;
* elementu skaitu, kas mazāki par pēdējo elementu.

#### ****Ievaddatu raksturojums****

Ievaddatu vienīgajā rindā ir desmit veseli skaitļi robežās no 0 līdz 100, kas atdalīti viens no otra ar tukšumzīmi.

#### ****Izvaddatu raksturojums****

Izvaddatu vienīgajā rindā ir trīs uzdevumā prasītie skaitļi, kas atdalīti viens no otra ar tukšumzīmi.

#include <iostream>

#include <algorithm>

using namespace std;

int main() {

long arr[10];

for (int i=0;i<10;i++) cin >> arr[i];

cout << (arr[0]+arr[9]) << " ";

int sk=0;

for (int i=1;i<10;i++) if(arr[i]>arr[0]) sk++;

cout << sk << " ";

sk = 0;

for (int i=0;i<9;i++) if(arr[i]<arr[9]) sk++;

cout << sk << " ";

}

#### ****Paraugdati****

##### **Stdin**

1 1 1 1 1 1 1 1 1 2

##### **Stdout**

3 1 9

# Taisnstūra paralēlskaldņa tilpums

**Definīcija**

Taisnstūra paralēlskaldnis ir daudzskaldnis ar sešām skaldnēm, kas visas ir taisnstūri. Ja taisnstūra paralēlskaldņa šķautņu garumi ir *a*, *b* un *c* (attēlā), tad tilpums V=a\*b\*c.

Aprēķināt taisnstūra paralēlskaldņa tilpumu, ja doti tā tīs škautņu garumi.

**Ievaddatu raksturojums**

Ievaddatu vienīgajā rindā doti taisnstūra paralēskaldņa trīs šķautņu garumi a, b un c (0<a,b,c<100).

**Izvaddatu raksturojums**

Izvaddatu vienīgajā rindā jāizvada taisnstūra paralēlskaldņa tilpums.

#include <iostream>

#include <set>

using namespace std;

int main() {

int a, b, c;

cin >> a >> b >> c ;

cout << (a\*b\*c);

}

#### ****Paraugdati****

##### **Stdin**

1 2 3

##### **Stdout**

6

# Cikls 4 iesācējiem

**Definīcija**

Izdrukāt visus skaitļus no 1 līdz dotajam skaitlim.

**Ievaddatu raksturojums**

Ievaddatu vienīgajā rindā dots vesels skaitlis a (1<a<32000).

**Izvaddatu raksturojums**

Izvaddati satur a rindas un tajās atrodas pēc kārtas visi veselie skaitļi no 1 līdz a.

#include <iostream>

#include <set>

using namespace std;

int main() {

int a;

cin >> a;

for (int i=1;i<=a;i++){

cout << i;

cout << "\n";

}

}

#### ****Paraugdati****

##### **Stdin**

3

##### **Stdout**

1

2

3

# Sazarojums3

**Definīcija**

Pārbaudīt vai skaitlis atrodas starp diviem dotajiem skaitļiem!

**Ievaddatu raksturojums**

Ievaddatu pirmajā rindā doti 2 skaitļi a un b (-100<a<b<100), bet otrajā rindā skaitlis c.

**Izvaddatu raksturojums**

Izvaddatu vienīgajā rindā jāizvada teksts "ir", ja skaitlis c atrodas starp skaitļiem a un b un teksts "nav" pretējā gadījumā. Ja skaitlis c sakrīt ar kādu no skaitļiem a vai b, tad tas neatrodas starp skaitļiem a un b.

#include <iostream>

using namespace std;

int main() {

int a, b, c;

cin >> a >> b;

cin >> c;

if (c > a && c < b)

{

cout << "ir";

} else {

cout << "nav";

}

}

#### ****Paraugdati****

##### **Stdin**

3 8

5

##### **Stdout**

ir

# Cikls5 iesācējiem

**Definīcija**

Jāatrod visu pēc kārtas ņemtu veselo skaitļu summa no viena skaitļa līdz otram.

**Ievaddatu raksturojums**

Ievaddatu vienīgajā rindā doti divi veseli skaitļi a un b (0<a<b<100).

**Izvaddatu raksturojums**

Izvaddatu vienīgajā rindā jāizvada visu pēc kārtas ņemtu veselu skaitļu summa no a līdz b, ieskaitot skaitļus a un b.

#include <iostream>

#include <set>

using namespace std;

int main() {

int a, b, sum=0;

cin >> a >> b;

for (int i=a;i<=b;i++){

sum = sum+i;

}

cout << sum;

}

#### ****Paraugdati****

##### **Stdin**

2 6

##### **Stdout**

20

# Sazarojums4

**Definīcija**

Pārbaudīt vai skaitlis ir pāra skaitlis.

**Ievaddatu raksturojums**

Ievaddatu vienīgajā rindā ir ievadīts vesels skaitlis a (0<a<100000).

**Izvaddatu raksturojums**

Izvaddatu vienīgajā rindā jāizvada teksts "ir", ja skaitlis ir pāra skaitlis un teksts "nav" pretējā gadījumā.

#include <iostream>

#include <set>

using namespace std;

int main() {

int a;

cin >> a;

if (a%2 != 1)

{cout << "ir";}

else

{cout << "nav";}

}

#### ****Paraugdati****

##### **Stdin**

64

##### **Stdout**

ir

# Vai pietiks tapešu?

**Definīcija**

Jānītis nolēma visas savas istabas sienas aplīmēt ar tapetēm. Viņš zina savas istabas izmērus, tapetes ruļļa izmērus un ruļļu skaitu. Uzraksti programmu, kas pēc dotajiem lielumiem nosaka, vai pietiks tapešu istabas sienu aplīmēšanai?

**Ievaddatu raksturojums**

Ievaddatu pirmajā rindā doti 3 veseli skaitļi a - istabas platums, b - istabas garums un c - istabas augstums (0<a,b,c<20). Ievaddatu otrajā rindā doti 3 veseli skaitļi d - tapetes ruļļa platums, e - tapetes ruļļa garums, f - ruļļu skaits.

**Izvaddatu raksturojums**

Izvaddatu pirmajā rindā jāizvada starpība starp sienu laukumu un tapešu ruļļu noklāto laukumu, bet otrajā rindā jāizvada teksts "pietiek", ja tapetes pietiek visu sienu noklāšanai un teksts "nepietiek" pretējā gadījumā.

#include <iostream>

using namespace std;

int main() {

long x, y, z; cin >> x >> y >> z;

long p, g, s; cin >> p >> g >> s;

long s1, s2, sienasLauk, ruljuLauk;

s1 = x\*z; s2 = y\*z;

sienasLauk = s1\*2 + s2\*2;

ruljuLauk = p\*g\*s;

if(sienasLauk == ruljuLauk) cout << 0 << endl << "pietiek";

else if(sienasLauk < ruljuLauk) cout << (sienasLauk-ruljuLauk) << endl << "pietiek";

else cout << (sienasLauk-ruljuLauk) << endl << "nepietiek";

}

#### ****Paraugdati****

##### **Stdin**

3 4 3

1 3 13

##### **Stdout**

3

nepietiek

# Vārds no burtiem

**Definīcija**

Izdrukāt vārdu, kas veidojas no citu vārdu pirmajiem burtiem.

**Ievaddatu raksturojums**

Katrā no ievaddatu piecām rindām dots viens vārds.

**Izvaddatu raksturojums**

Izvaddatu vienīgajā rindā ir vārds, kas sastāv no piecu doto vārdu pirmajiem burtiem.

#include <iostream>

using namespace std;

int main() {

string v1, v2, v3, v4, v5;

cin >> v1; cin >> v2; cin >> v3; cin >> v4; cin >> v5;

cout << v1.at(0);

cout << v2.at(0);

cout << v3.at(0);

cout << v4.at(0);

cout << v5.at(0);

}

#### ****Paraugdati****

##### **Stdin**

sile

abra

upe

lietus

ezers

##### **Stdout**

saule

# Pēdējais burts vārdā

**Definīcija**

Atrast vārda pēdējo burtu.

**Ievaddatu raksturojums**

Ievaddatu vienīgajā rindā dots vārds.

**Izvaddatu raksturojums**

Izvaddatu vienīgajā rindā ir dotā vārda pēdējais burts.

#include <iostream>

using namespace std;

int main() {

string vards;

cin >> vards;

cout << vards.at(vards.length()-1);

}

#### ****Paraugdati****

##### **Stdin**

bulta

##### **Stdout**

a

# Dīvaina summa

**Definīcija**

Doti n veseli nenegatīvi skaitļi. Jāatrod šo skaitļu dīvainā summa, izmantojot tālāk doto algoritmu.

Ja skaitlis ir nepāra skaitlis, tad tas tiek vienkārši pieskaitīts summai.

Ja skaitlis ir pāra skaitlis, tad tas tiek izdalīts ar 2 un pieskaitīts summai.

**Ievaddatu raksturojums**

Ievaddatu pirmajā rindā dots skaitļu skaits n (1<=n<=100). Tālākajās n rindās doti veseli skaitļi a (1<=a<=100).

**Izvaddatu raksturojums**

Izvaddatu vienīgajā rindā ir atrastā summa.

#include <iostream>

using namespace std;

int main() {

int n; cin >> n;

int masivs[n];

for (int i=0;i<n;i++) cin >> masivs[i];

int sum=0;

for (int i=0;i<n;i++) if(masivs[i]%2 != 0) sum = sum+masivs[i];

else sum = sum + (masivs[i]/2);

cout << sum;

}

#### ****Paraugdati****

##### **Stdin**

4

2

6

3

4

##### **Stdout**

9

# Dotā burta skaits vārdā

**Definīcija**

Atrast cik reižu dotais burts atkārtojas dotajā vārdā.

**Ievaddatu raksturojums**

Ievaddatu pirmajā rindā ir dots vārds, kura garums nepārsniedz 20 simbolus, bet otrajā rindā burts.

**Izvaddatu raksturojums**

Izvaddatu vienīgajā rindā ir skaitlis, kas norāda cik reižu dotais burts atkārtojas dotajā vārdā.

#include <iostream>

using namespace std;

int main() {

string vards, a, b;

int sk=0;

cin >> vards; cin >> b;

for (int i=0;i<vards.length();i++){

a = vards.at(i);

if (a.compare(b) == 0)

{sk++;}

}

cout << sk;

}

#### ****Paraugdati****

##### **Stdin**

abra

a

##### **Stdout**

2

# Garākais vārds

**Definīcija**

Atrast garāko no 2 dotajiem vārdiem. Ja garāks ir pirmais vārds vai vārdi ir vienāda garuma, tad atbilde ir „Pirmais”, bet pretējā gadījumā atbilde ir „Otrais”.

**Ievaddatu raksturojums**

Ievaddatu 2 rindās doti 2 vārdi, kuru garums nepārsniedz 20 simbolus.

**Izvaddatu raksturojums**

Izvaddatu vienīgajā rindā jāizdrukā atbilde **Pirmais** vai **Otrais.**

#include <iostream>

#include <set>

using namespace std;

int main() {

string v1, v2;

cin >> v1 >> v2;

long x = v1.length(), y = v2.length();

if (x == y)

cout << "Pirmais";

else if(x > y){

cout << "Pirmais";

}

else

cout << "Otrais";

}

#### ****Paraugdati****

##### **Stdin**

desa

kurbulis

##### **Stdout**

Otrais

# Kurš burts pirmais?

**Definīcija**

Dots vārds, kura garums nepārsniedz 20 burtus un 2 burti. Atrast un izvadīt to burtu no diviem dotajiem, kurš, lasot doto vārdu, parādās pirmais. Zināms, ka katrs no burtiem vārdā sastopams tieši vienu reizi.

**Ievaddatu raksturojums**

Ievaddatu pirmajā rindā dots vārds, bet otrajā un trešajā rindā katrā pa burtam.

**Izvaddatu raksturojums**

Izvaddatu vienīgajā rindā ir atrastais burts.

#include <iostream>

#include <set>

using namespace std;

int main() {

string vards, a, b, c;

cin >> vards;

cin >> a;

cin >> b;

for (int i=0;i<vards.length();i++){

c = vards.at(i);

if (c.compare(a) == 0)

{cout << a; break;}

else if (c.compare(b) == 0)

{cout << b; break;}

}

}

#### ****Paraugdati****

##### **Stdin**

pasaka

k

s

##### **Stdout**

s

# Vārda veidošana

**Definīcija**

Doti 2 vārdi, kuru garums nepārsniedz 20 burtus. Pārbaudīt vai no pirmā vārda burtiem var izveidot otro vārdu. Zināms, ka otrais vārds ir īsāks par pirmo un tajā neviens burts neatkārtojas. Atbildes ir **Var** un  **Nevar**.

**Ievaddatu raksturojums**

Ievaddatu pirmajās divās rindās doti 2 vārdi.

**Izvaddatu raksturojums**

Izvaddatu vienīgajā rindā ir vārds **Var** vai **Nevar**.

#include <algorithm>

#include <iostream>

#include <vector>

using namespace std;

int main() {

string v1, v2; cin >> v1 >> v2;

bool itCan = true;

vector<char> V1, V2;

for(int i=0;i<v1.length();i++) V1.push\_back(v1.at(i));

for(int i=0;i<v2.length();i++) V2.push\_back(v2.at(i));

for(int i=0;i<V2.size();i++) if (std::find(V1.begin(), V1.end(), V2.at(i)) == V1.end()){

itCan = false; }

if (itCan == true) cout << "Var";

else cout << "Nevar";

}

#### ****Paraugdati****

##### **Stdin**

saule

sula

##### **Stdout**

Var

# Massīva kārtošana

**Definīcija**

Doti n (1<n<=10) veseli skaitļi robežās no -100 līdz 100. Uzdevums sakārtot tos nedilstošā secībā un izvadīt vienu zem otra.

**Ievaddatu raksturojums**

Ievaddatu pirmajā rindā dots skaitlis n. Sākot ar 2 rindu n rindās ierakstīti veseli skaitļi.

**Izvaddatu raksturojums**

Izvadatu n rindās izvadīti n skaitļi nedilstošā secībā.

#include <iostream>

using namespace std;

int main() {

int n; cin >> n;

int masivs[n];

for (int i=0;i<n;i++) cin >> masivs[i];

for (int i = 0; i < n - 1; i++){

for (int j = 0; j < n - i - 1; j++){

if (masivs[j] > masivs[j + 1]) swap(masivs[j], masivs[j + 1]);

}

}

for (int i=0;i<n;i++){

cout << masivs[i] << endl;

}

}

#### ****Paraugdati****

##### **Stdin**

3

7

-4

1

##### **Stdout**

-4

1

7

# Sakārto vārdus alfabēta secībā

**Definīcija**

Doti n (1<n<10) skolēnu vārdi. Sakārtot tos alfabēta secībā.

**Ievaddatu raksturojums**

Ievaddatu pirmajā rindā dots skaitlis n. Tālāk n rindās doti skolēnu vārdi.

**Izvaddatu raksturojums**

Izvaddatu n rindās ir skolēnu vārdi alfabēta secībā.

#include <algorithm>

#include <iostream>

using namespace std;

int main() {

int n; cin >> n;

string arr[n];

for (int i=0;i<n;i++) cin >> arr[i];}

int i, j;

for (i = 0; i < n - 1; i++)

for (j = 0; j < n - i - 1; j++)

if ((arr[j]).at(0) > (arr[j + 1]).at(0)) swap(arr[j], arr[j + 1]);

for (int i=0;i<n;i++) cout << arr[i] << endl;

}

#### ****Paraugdati****

##### **Stdin**

3

Valdis

Anna

Inese

##### **Stdout**

Anna

Inese

Valdis

# Pēdējie cipari

**Definīcija**

Doti divi veseli skaitļi (0<a,b<10000). Atrast abu skaitļu pēdējo ciparu summu.

**Ievaddatu raksturojums**

Ievaddatu vienīgajā rindā ir divi skaitļi, atdalīti ar tukšumzīmi.

**Izvaddatu raksturojums**

Izvaddatu vienīgajā failā ir abu skaitļu pēdējo ciparu summa.

#include <algorithm>

#include <iostream>

#include <vector>

using namespace std;

int main() {

int n, m;

cin >> n >> m;

int pag = n%10 + (m%10);

cout << pag ;

}

#### ****Paraugdati****

##### **Stdin**

121

13

##### **Stdout**

4

# Aizstāj burtu

**Definīcija**

Dots vārds un 2 burti. Uzdevums sameklēt dotajā vārdā visus burtus, kas vienādi ar pirmo doto burtu un aizstāt tos ar otro doto burtu.

**Ievaddatu raksturojums**

Ievaddatu pirmajā rindā dots vārds, bet otrajā un trešajā rindā viens burts.

**Izvaddatu raksturojums**

Izvaddatu vienīgajā rindā jāizvada vārds pēc pārveidošanas.

#include <iostream>

using namespace std;

int main() {

string vards, vards2="", a, b, c;

cin >> vards; cin >> a; cin >> b;

for (int i=0;i<vards.length();i++){

c = vards.at(i);

if (c.compare(a) == 0){vards2 = vards2+b;}

else {vards2 = vards2+c;}

}

cout << vards2;

}

#### ****Paraugdati****

##### **Stdin**

lapa

a

u

##### **Stdout**

lupu

# Ciparu summa

**Definīcija**

Atrast vesela skaitļa ciparu summu!

**Ievaddatu raksturojums**

Ievaddatu vienīgajā rindā dots vesels skaitlis a (0<a<264).

**Izvaddatu raksturojums**

Izvaddatu vienīgajā rindā jāizvada skaitļa a ciparu summa.

#include <iostream>

using namespace std;

int main() {

long x,sum=0,z;

cin >> x;

for (long i=10;i<x;i=i\*10){

z = i;

}

for (long i=z;i>=10;i=i/10){

sum = sum + (x/i);

x = x%i;

}

sum = sum +x%10;

cout << sum << endl;

}

#### ****Paraugdati****

##### **Stdin**

123

##### **Stdout**

6

# Sporta skola

#### ****Definīcija****

Gudrinieku ciema sporta skolas klase mēdz spēlēt daudz sporta spēles. Lai to darītu, ir jāsadalās divās komandās. Skolotājs izdomāja, kā to darīt:

* Sastāda visus rindā.
* Piešķir katram skaitli pēc kārtas no N līdz M.

Pāra skolnieki ir vienā komandā, nepāra otrā.

#### ****Ievaddatu raksturojums****

Ievaddatos tiek doti divi skaitļi 0 ≤ N ≤ M ≤ 100.

#### ****Izvaddatu raksturojums****

Izvaddatos jāizvada:

1. rindā visi nepāra kārtas spēlētāju numuri.

2. rindā visi pāra kārtas spēlētāju numuri.

#include <algorithm>

#include <iostream>

using namespace std;

int main() {

int n, m; cin >> n >> m;

for (int i = n; i <= m; i++) {

if (i % 2 == 1)

cout << i <<' ';

}

cout << endl;

for (int i = n; i <= m; i++) {

if (i % 2 == 0) cout << i << ' ';

}

}

#### ****Paraugdati****

##### **Stdin**

3 6

##### **Stdout**

3 5

4 6

# Cikls6 iesācējiem

**Definīcija**

Doti divi skaitļi. Izdrukāt mazāko skaitli tik reižu, cik liels ir lielākais skaitlis.

**Ievaddatu raksturojums**

Doti 2 veseli skaitļi a un b (1<=a,b<=100).

**Izvaddatu raksturojums**

Izdrukāt mazāko skaitli vienu zem otra tik reizes, cik liels ir lielākais skaitlis.

#include <iostream>

using namespace std;

int main() {

int a, b; cin >> a >> b;

if(a>b){

for(int i=0;i<a;i++) {cout << b; cout << "\n";}

} else { for(int i=0;i<b;i++) {cout << a; cout << "\n";} }

}

#### ****Paraugdati****

##### **Stdin**

2 4

##### **Stdout**

2

2

2

2

# Vārds no pirmajiem burtiem

**Definīcija**

Dots noteikts vārdu skaits. Izveidot vārdu no doto vārdu pirmajiem burtiem.

**Ievaddatu raksturojums**

Pirmajā rindā dots vārdu skaits a (1<=a<=100). Nākamajās a rindās doti vārdi.

**Izvaddatu raksturojums**

Vienīgajā rindā izdrukāts vārds, kas veidojas no doto vārdu pirmajiem burtiem.

#include <iostream>

using namespace std;

int main() {

string finalVards="";

int a;

cin >> a;

string vards[a];

for (int i=0;i<a;i++){

cin >> vards[i];

finalVards = finalVards + vards[i].at(0);

}

cout << finalVards;

}

#### ****Paraugdati****

##### **Stdin**

4

diena

es

sams

aita

##### **Stdout**

desa

# Sakārto vārda burtus

**Definīcija**

Dots vārds. Izveido jaunu vārdu, kas veidots no dotā vārda burtiem alfabētiskā secībā.

**Ievaddatu raksturojums**

Vienīgajā rindā dots vārds, kura garums ir vismaz 2 burti un ne varāk kā 100 burti.

**Izvaddatu raksturojums**

Vienīgajā rindā izvadīti dotā vārda burti alfabētiskā secībā.

#include <iostream>

using namespace std;

int main() {

string vards, vards2="", a, b, c;

int sk=0, vardsLength;

cin >> vards;

vardsLength = vards.size();

for (int j=0;j<vardsLength;j++){

sk=0;

a = vards.at(0);

for (int i=1;i<vards.length();i++){

b = vards.at(i);

if(a>b) a = b;

}

for (int i=0;i<vards.size();i++){

c = vards.at(i);

if (c == a)

break;

else

sk++;

}

vards.erase(sk, 1);

vards2 = vards2+a;

}

cout << vards2;

}

#### ****Paraugdati****

##### **Stdin**

desa

##### **Stdout**

ades

# ZigZag dialekts

**Definīcija**

Gudrinieku ciems ir ļoti liels un ne visi tā iedzīvotāji runā vienā dialektā. Lai gan dialekti ir no vienas valodas, lai vieni iedzīvotāji saprastu citus, ir nepieciešams vārdu konvērtētājs. Tavs uzdevums ir uzrakstīt konvērtētāju, kurš jebkuru vārdu konvertēs pēc sekojoša algoritma. Algoritms uzraksta burtu ZigZag formā: vārda pirmie trīs burti tiek uzrakstīti stabiņā, nākamais viens burts tiek uzrakstīts otrā stabiņa centrā, nākamie trīs burti 3. stabiņā, nākamais burts 4. stabiņa centrā utt. Pēc tam algoritms nolasa vārdu no rindām secīgi.

Piemēram, ja dotais vārds ir **PAYPALISHIRING**, tā ZigZag forma ir šāda:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| P |  | A |  | H |  | N |
| A | P | L | S | I | I | G |
| Y |  | I |  | R |  |  |

Tālāk lasot katru rindu, veidojās vārds ZigZag dialektā – **PAHNAPLSIIGYIR**.

**Ievaddatu raksturojums**

Ievaddatos tiek dots vārds, kura garums ir starp 0 un 100 simboliem. Vārds sastāv tikai no latīņu alfabēta lielajiem burtiem.

**Izvaddatu raksturojums**

Izvaddatos jāizvada vārds ZigZag dialektā.

#include <iostream>

using namespace std;

int main() {

string vards; cin >> vards;

string r1="", r2="", r3="";

for (int i=0;i<vards.length();i=i+4) if(i <= vards.length()) r1 = r1 + vards.at(i);

for (int i=1;i<vards.length();i=i+2) if(i <= vards.length()) r2 = r2 + vards.at(i);

for (int i=2;i<vards.length();i=i+4) if(i <= vards.length()) r3 = r3 + vards.at(i);

cout << r1 << r2 << r3;

}

#### ****Paraugdati****

##### **Stdin**

PAYPALISHIRING

##### **Stdout**

PAHNAPLSIIGYIR