

#include <cctype>

#include <iostream>

#include <string>

using namespace std;

int main() {

int a, b, step = 1;

cin >> a >> b;

if (a > b) step = -1;

for (int i = a - 1; i != b - 1 + step; i += step) {

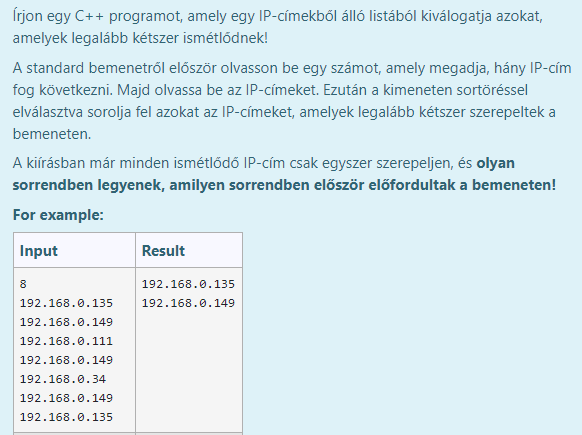
cout << char('A' + i);

}

cout << endl;

return 0;

}



#include <iostream>

#include <string>

using namespace std;

int main() {

int n;

cin >> n;

string\* ips = new string[n];

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

cin >> ips[i];

}

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

bool found = false;

for (int j = 0; j < i && !found; ++j) {

if (ips[j] == ips[i]) found = true;

}

if (!found) {

for (int j = i + 1; j < n && !found; ++j) {

if (ips[j] == ips[i]) {

found = true;

cout << ips[i] << endl;

}

}

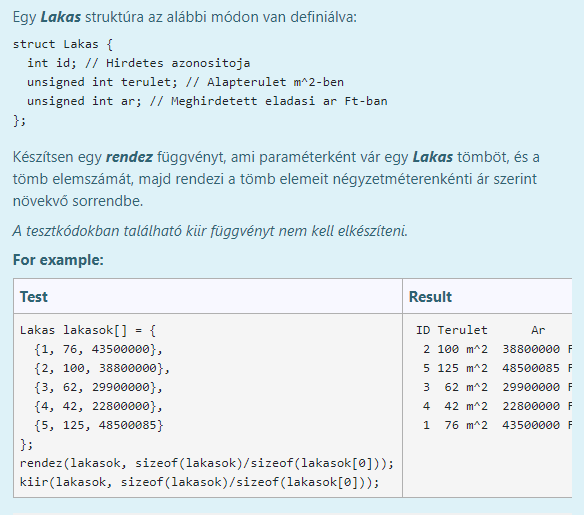
}

}

delete[] ips;

return 0;

}



void rendez(Lakas tomb[], int db) {

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

int mini = i;

double minval = double(tomb[i].ar) / tomb[i].terulet;

for (int j = i + 1; j < db; ++j) {

if (double(tomb[j].ar) / tomb[j].terulet < minval) {

mini = j;

minval = tomb[j].ar / tomb[j].terulet;

}

}

if (mini != i) {

Lakas csere = tomb[i];

tomb[i] = tomb[mini];

tomb[mini] = csere;

}

}

}



Versenyzo\* eloz(Versenyzo\* horgony, string nev) {

if (!horgony || !horgony->kov) return horgony;

Versenyzo\* akt = horgony;

while (akt->kov && akt->kov->nev != nev) akt = akt->kov;

if (akt->kov) {

string tmp = akt->nev;

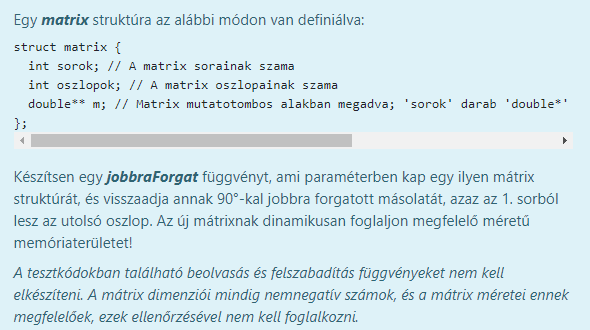
akt->nev = akt->kov->nev;

akt->kov->nev = tmp;

}

return horgony;

}



matrix jobbraForgat(const matrix& mtx) {

matrix uj = {mtx.oszlopok, mtx.sorok, nullptr};

uj.m = new double\*[uj.sorok];

for (int i = 0; i < uj.sorok; ++i) {

uj.m[i] = new double[uj.oszlopok];

for (int j = 0; j < uj.oszlopok; ++j) {

uj.m[i][j] = mtx.m[mtx.sorok - j - 1][i];

}

}

return uj;

}