8. Scrieți un algoritm care rezolvă problema labirintului.

#include <iostream>

#include <stdlib.h>

using namespace std;

struct tpunct {

int x,y;

};

int a[50][50];

tpunct stiva[100];

tpunct coada[100];

int i,j,m,n,sr,sc,br,bc,s;

int vc, vs;

int sfc, k;

bool f;

int main() {

cout << "Introdu dimensiunile labirintului: \n";

cout << "m = "; cin >> m;

cout << "n = "; cin >> n;

for (i = 1; i < m; i++)

for (j = 1; j < m; j++)

a[i][j] = rand() % 2 - 1;

cout << "Matricea initiala: \n";

for (i = 1; i < m; i++) {

for (j = 1; j < n; j++)

if (a[i][j] < 0)

cout << " " << a[i][j];

else

cout << " " << a[i][j];

cout << '\n';

}

cout << "Introdu coordonatele soarecelui: ";

cout << "\nsr = "; cin >> sr;

cout << "sc = "; cin >> sc;

a[sr][sc] = 1;

i = sr;

j = sc;

cout << "Introdu coordonatele brinzei: ";

cout << "\nbr = "; cin >> br;

cout << "bc = "; cin >> bc;

vc = 1;

sfc = 1;

coada[vc].x = sr;

coada[vc].y = sc;

do {

f = false;

if ((i - 1 > 0) && (a[i - 1][j] == 0)) {

a[i - 1][j] = a[i][j] + 1;

sfc = sfc + 1;

coada[sfc].x = i - 1;

coada[sfc].y = j;

f = true;

}

if ((i + 1 <= m) && (a[i + 1][j] == 0)) {

a[i + 1][j] = a[i][j] + 1;

sfc = sfc + 1;

coada[sfc].x = i + 1;

coada[sfc].y = j;

f = true;

}

if ((j - 1 > 0) && (a[i][j - 1] == 0)) {

a[i][j - 1] = a[i][j] + 1;

sfc = sfc + 1;

coada[sfc].x = i;

coada[sfc].y = j - 1;

f = true;

}

if ((j + 1 <= n) && (a[i][j + 1] == 0)) {

a[i][j + 1] = a[i][j] + 1;

sfc = sfc + 1;

coada[sfc].x = i;

coada[sfc].y = j + 1;

f = true;

}

if (f == false) {

vc = vc + 1;

i = coada[vc].x;

j = coada[vc].y;

}

} while ((coada[vc].x != br) || (coada[vc].y != bc));

cout << "Matricea nr. 2: \n";

for (i = 1; i < m; i++) {

for (j = 1; j < n; j++)

if (a[i][j] < 0)

cout << " " << a[i][j];

else

cout << " " << a[i][j];

cout << '\n';

}

k = a[br][bc];

cout << "k =" << k << '\n';

vs = 1;

stiva[vs].x = br;

stiva[vs].y = bc;

i = br;

j = bc;

do {

f = false;

if ((i - 1 > 0) && (a[i - 1][j] == k - 1)) {

vs = vs + 1;

stiva[vs].x = i - 1;

stiva[vs].y = j;

f = true;

}

if ((i + 1 <= m) && (a[i + 1][j] == k - 1) && (f == false)) {

vs = vs + 1;

stiva[vs].x = i + 1;

stiva[vs].y = j;

f = true;

}

if ((j + 1 <= n) && (a[i][j + 1] == k - 1) && (f == false)) {

vs = vs + 1;

stiva[vs].x = i;

stiva[vs].y = j + 1;

f = true;

}

if ((j - 1 > 0) && (a[i][j - 1] == k - 1) && (f == false)) {

vs = vs + 1;

stiva[vs].x = i;

stiva[vs].y = j - 1;

f = true;

}

i = stiva[vs].x;

j = stiva[vs].y;

k = k - 1;

} while (k != 1);

for (i = vs; i > 1; i--)

cout << "\n(" << stiva[i].x << ", " << stiva[i].y << ");";

return 0;

}

