МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»

Лабораторна робота № 5

З дисципліни

“Математичні методи дослідження операцій”

Виконав:

Студент групи КН-314

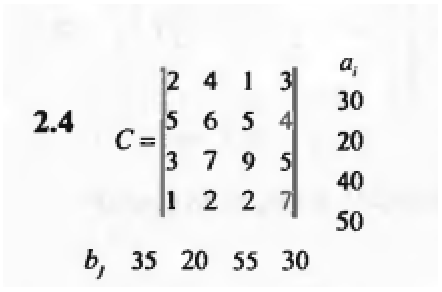
Ляшеник Остап

Прийняв

Шиманський Володимир Михайлович

Львів - *2023*

Постановка завдання



2.4

$$\begin{array}{c|cccc|c} & & & & & a_i \\ C = & 2 & 4 & 1 & 3 & 30 \\ & 5 & 6 & 5 & 4 & 20 \\ & 3 & 7 & 9 & 5 & 40 \\ & 1 & 2 & 2 & 7 & 50 \\ b_j & 35 & 20 & 55 & 30 & \end{array}$$

```
#include <iostream>

using namespace std;

const int n(4), m(4);

int** cycle_runner(bool mask[n][m], int start[2], int last[2], int current[2], int l, int** way, int& size){

int** local_way = new int*[l+1];

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

local_way[i] = new int[2];

local_way[i][0] = way[i][0];

local_way[i][1] = way[i][1];

}

local_way[l] = new int[2];

local_way[l][0] = current[0];

local_way[l][1] = current[1];

if(last[0] == current[0] || l == 0){

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

if(i == start[0] && current[1] == start[1] && l > 1) {

size = l+1;

return local_way;

}
if(mask[i][current[1]] == 1 && i != current[0]){

int next_cords[2] {i, current[1]};

int** tmp_arr = cycle_runner(mask, start, current, next_cords, l+1, local_way, size);

if(tmp_arr != nullptr)

return tmp_arr;

}

}

return nullptr;

}

if(last[1] == current[1] || l == 0){

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

if(i == start[1] && current[0] == start[0] && l > 1) {

size = l+1;

return local_way;

}

if(mask[current[0]][i] == 1 && i != current[1]){

int next_cords[2] {current[0], i};

int** tmp_arr = cycle_runner(mask, start, current, next_cords, l+1, local_way, size);

if(tmp_arr != nullptr)

return tmp_arr;

}

}

return nullptr;

}

}
```

```
int main(){

int c[n][m] = {{2,4,1,3}, {5,6,5,4}, {3,7,9,5}, {1,2,2,7}};

int a[n] = {30,20,40,50};

int b[m] = {35,20,55,30};


int summ = 0;

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

summ += a[i];

}

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

summ -= b[i];

}

if(summ) {

cout << "problem is unsolvable" << endl;

return 0;

}

int potentials[n][m]{0};

bool potentials_mask[n][m]{0};

int i_(0), j_(0);

while (true) {

if (a[i_] == 0) ++i_;

else if (b[j_] == 0) ++j_;

if (i_ == n || j_ == m)

break;

if (a[i_] >= b[j_]) {
```

```
a[i_] -= b[j_];
potentials[i_][j_] = b[j_];
b[j_] = 0;
} else {
b[j_] -= a[i_];
potentials[i_][j_] = a[i_];
a[i_] = 0;
}
potentials_mask[i_][j_] = 1;
}
int alpha[n]{0};
int betha[m]{0};
bool alpha_mask[n] {0};
bool betha_mask[n] {0};
while(true){
for(int i = 0; i < n; i++){
alpha_mask[i] = 0;
}
alpha_mask[0] = 1;
for(int i = 0; i < m; i++){
betha_mask[i] = 0;
}
while(true) {
for (int i = 0; i < n; i++) {
for (int j = 0; j < m; j++) {
```

```
if (potentials_mask[i][j] == 1) {

if (betha_mask[j] == 0 && alpha_mask[i]) {

betha_mask[j] = 1;

betha[j] = c[i][j] - alpha[i];

} else if (alpha_mask[i] == 0 && betha_mask[j]) {

alpha_mask[i] = 1;

alpha[i] = c[i][j] - betha[j];

}

}

}

}

bool braker = 1;

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

if(!alpha_mask[i])

braker = 0;

}

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

if(!betha_mask[i])

braker = 0;

}

if(braker)

break;

}

int highest_empty_potential = 0;

int h_i, h_j;
```

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

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

if(potentials_mask[i][j] == 0){

potentials[i][j] = alpha[i]+betha[j]-c[i][j];

if(potentials[i][j] > highest_empty_potential){

highest_empty_potential = potentials[i][j];

h_i = i;

h_j = j;

}

}

}

}

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

cout << alpha[i] << '\t';

}

cout << endl;

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

cout << betha[i] << '\t';

}

cout << endl;

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

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

if(potentials_mask[i][j])

cout << "[" << potentials[i][j] << "]\t";

else
```

```
cout << potentials[i][j] << '\t';

}

cout << endl;

}

cout << endl;

if(highest_empty_potential <= 0){

break;

}

int start_cords[2] {h_i, h_j};

int len;

int** cycle_cords = cycle_runner(potentials_mask, start_cords, start_cords,
start_cords, 0, nullptr, len);

if(cycle_cords == nullptr){

cout << "Unable to solve" << endl;

break;

}

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

cout << cycle_cords[i][0]+1 << " " << cycle_cords[i][1]+1 << endl;

}

cout << endl;

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

int min_potential = 1000000;

int l_i, l_j;

for(int i = 1; i < len; i+=2){

if(potentials[cycle_cords[i][0]][cycle_cords[i][1]] <= min_potential){

min_potential = potentials[cycle_cords[i][0]][cycle_cords[i][1]];
```

```
l_i = cycle_cords[i][0];
l_j = cycle_cords[i][1];
}
}
potentials_mask[h_i][h_j] = 1;
potentials_mask[l_i][l_j] = 0;
potentials[h_i][h_j] = min_potential;
for(int i = 1; i < len; i++){
if(cycle_cords[i][0] != l_i || cycle_cords[i][1] != l_j){
if(i % 2){
potentials[cycle_cords[i][0]][cycle_cords[i][1]] -= min_potential;
}else{
potentials[cycle_cords[i][0]][cycle_cords[i][1]] += min_potential;
}
}
}
}
int sum = 0;
for(int i = 0; i < n; i++){
for(int j = 0; j < m; j++){
if(potentials_mask[i][j])
sum += potentials[i][j]*c[i][j];
}
}
cout << "RESULT: " << sum;
```

```
}


0      3      4      -3

2      3      5      10

[30]    -1     4      7

[5]     [15]   3      9

3      [5]    [35]   9

-2     -2     [20]   [30]


2 4

4 4

4 3

3 3

3 2

2 2


----------------------

0      3      13     6

2      -6     -4     1

[30]    -10    -5     -2

[5]     -9     -6     [15]

12     [20]   [20]   9

7      -2     [35]   [15]


3 1
```

```
2 1

2 4

4 4

4 3

3 3


----------------------

0      -9     1      -6

2      6      8      13

[30]   2      7      10

-12    -9     -6     [20]

[5]    [20]   [15]   9

-5     -2     [40]   [10]


1 4

4 4

4 3

3 3

3 1

1 1


----------------------

0      1      1      -6

2      6      8      3

[20]   2      7      [10]
```

```
-2      1       4       [20]

[15]    [20]    [5]     -1

-5      -2      [50]    -10


1 3

3 3

3 1

1 1


----------------------

0       1       1       1

2       6       1       3

[15]    2       [5]     [10]

-2      1       -3      [20]

[20]    [20]    -7      -1

2       5       [50]    -3


4 2

3 2

3 1

1 1

1 3

4 3


----------------------
```

```
0       1       6       1

-3      1       1       3

-5      -3      [20]    [10]

-7      -4      -3      [20]

[35]    [5]     -2      4

-3      [15]    [35]    -3


3 4

1 4

1 3

4 3

4 2

3 2


----------------------

0       1       2       1

1       1       1       3

-1      -3      [25]    [5]

-3      -4      -3      [20]

[35]    -4      -6      [5]

1       [20]    [30]    -3


4 1

3 1

3 4
```

```
1 4

1 3

4 3


----------------------

0       2       3       1

0       1       1       2

-2      -3      [30]    -1

-3      -3      -2      [20]

[30]    -3      -5      [10]

[5]     [20]    [25]    -4


RESULT: 345
```