Мета роботи: вивчення симплекс методудля знаходження розв’язку задач лінійного програмування .

**План**

1. Короткі теоретичні відомості.
2. Постановка задачі

Симплекс метод розв’язаний за допомогою симлекс таблиць з додаванням змінних.

1. Розв’язання за допомогою пакету програм SimplexWin
2. Розв’язання за допомогою власної програми.

**Хід роботи**

**1. Короткі теоретичні відомості**

Симплекс метод - універсальний метод для вирішення лінійної системи рівнянь або нерівностей і лінійного функціоналу.

Для приведення системи обмежень нерівностей до канонічного виду, необхідно в системі обмежень виділити одиничний базис.

I. Обмеження виду «****» - ресурсні обмеження. Справа знаходиться те що ми використовуємо на виробництві, ліворуч - те що отримуємо. При таких обмеженнях вводять додаткові змінні з коефіцієнтом «+1», що утворюють одиничний базис. У цільову функцію ці змінні увійдуть з коефіцієнтом «0».

II. Обмеження виду «=». Часто буває, що незважаючи на те що обмеження мають вигляд рівності, одиничний базис не виділяється або важко виділяється. В цьому випадку вводяться штучні змінні для створення одиничного базису - Yi. В систему обмежень вони входять з коефіцієнтом «1», а в цільову функцію з коефіцієнтом «M», що прагнуть до нескінченності (при Fmin - «+ M», при Fmax - «-M»).

III. Обмеження виду «****» - планові обмеження. Додаткові змінні (X), що несуть певний економічний зміст - перевитрата ресурсів або перевиконання плану, перевиробництво, додаються з коефіцієнтом «-1», в цільову функцію - з коефіцієнтом «0». А штучні змінні (Y) як у попередньому випадку.

**Алгоритм симплекс методу**

Нехай система приведена до канонічного виду.

X1+ q1,m+1 Xm+1 + …. + q1,m+n Xm+n = h1

X2+ q1,m+1 Xm+1 + …. + q1,m+n Xm+n = h1 (1)

X3+ q1,m+1 Xm+1 + …. + q1,m+n Xm+n = h1

……………………………………………………………….

Xm+ qm,m+1 Xm+1 + …. + qm,m+n Xm+n =hm

У ній m базисних змінних, k вільних змінних. m + k = n - всього змінних.

Fmin = C1X1 + C2X2 + C3X3 + .... + CnXn

Всі hi повинні бути більше або дорівнюють нулю, де i = 1,2 ... m. На першому кроці в якості допустимого рішення приймаємо всі Xj = 0 (j = m +1, m +2, ..., m + k). При цьому всі базисні змінні Xi = Hi.

Для подальших міркувань обчислень будемо користуватися першою симплекс таблицею.

Таблица 1 –Перша симплес таблиця .

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| C | Б | H | C1 | C2 | … | Cm | Cm+1 | … | Cm+k |
|  |  |  | X1 | X2 | … | Xm | Xm+1 | … | Xm+k |
| C1 C2  C3  :  :  Cm | X1 X2  X3  :  :  Xm | h1 h2  h3  :  :  hm | 1  0  0  :  :  0 | 0  1  0  :  :  0 | :  :  :  :  :  : | 0  0  0  :  :  0 | q1,m+1  q2,m+1  q3,m+1  :  :  qm,m+1 | :  :  :  :  :  : | q1,m+k  q2,m+k  q3,m+k  :  :  qm,m+k |
|  | F= | F0 |  |  | … | m | m+1 | … | m+k |

Перший стовпчик-коефіцієнти в цільової функції при базисних змінних.

Другий стовпчик - базисні змінні.

Третій стовпчик - вільні члени (hi****0).

Самий верхній рядок - коефіцієнти при цільовій функції.

Другий верхній рядок - змінні, що входять в цільову функцію і в систему обмежень.

Основне поле симплекс методу - система коефіцієнтів з рівняння.

Останній рядок - служить для того, щоб відповісти на питання: «оптимальний план чи ні».

**2. Постановка задачі**

**Розв’язати задачу лінійного програмування симплекс методом** (номер завдання відповідає двом останнім цифрам залікової книжки студента, крім цифр 00 – які відповідають завданню під номером100)

***Варіант 32***

#### F(x1,x2) = x1 + x2 max ;

3*x*1 - 2*x*2 -6,

*x*1+ 2*x*2  3,

3*x*1  0, 5 *x*2 0.

2. Розв’язання без використання пакетів програм.

***Cимплекс метод розв’язаний за допомогою сиплекс таблиць***

***З додаванням додаткових змінних .***

*Умова:*

F(X) = 14x1 + 10x2

7x1 + 5x2≥7

7x1 - 5x2≥35

x1 - x2≤0

Вирішимо пряму задачу лінійного програмування симплексним методом, з використанням симплексного таблиці.  
Оскільки в правій частині присутні негативні значення, помножимо відповідні рядки на (-1).  
Визначимо максимальне значення цільової функції F (X) = x1 + x2 при наступних умовах-обмежень.  
- 3x1 + 2x2 ≤ 6  
x1 + 2x2 ≥ 3  
Для побудови першого опорного плану систему нерівностей приведемо до системи рівнянь шляхом введення додаткових змінних (перехід до канонічної формі).  
-3x1 + 2x2 + 1x3 + 0x4 = 6  
1x1 + 2x2 + 0x3-1x4 = 3  
Введемо штучні змінні x: в 2-м рівність вводимо змінну x5;  
-3x1 + 2x2 + 1x3 + 0x4 + 0x5 = 6  
1x1 + 2x2 + 0x3-1x4 + 1x5 = 3  
Для постановки задачі на максимум цільову функцію запишемо так:  
F (X) = x1 + x2 - Mx5 → max  
З рівнянь виражаємо штучні змінні:  
 x5 = 3-x1-2x2 + x4  
які підставимо в цільову функцію:  
F (X) = (1 +1 M) x1 + (1 +2 M) x2 + (-1M) x4 + (-3M) → max  
Матриця коефіцієнтів A = a (ij) цієї системи рівнянь має вигляд:

Вирішимо систему рівнянь щодо базисних змінних:  
x3, x5,  
Вважаючи, що вільні змінні рівні 0, отримаємо перший опорний план:  
X1 = (0,0,6,0,3)

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Базис | БП | x 1 | x 2 | x 3 | x 4 | x 5 | x 6 | z 1 |
| x3 | 6 | -3 | 2 | 1 | 0 | 0 | 0 | 0 |
| z1 | 3 | 1 | 2 | 0 | -1 | 0 | 0 | 1 |
| x5 | 3 | 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| x6 | 5 | 0 | 1 | 0 | 0 | 0 | 1 | 0 |
| ИС | -3M | -M-1 | -2M-1 | 0 | M | 0 | 0 | 0 |

**Переходимо до основного алгоритму симплекс-метода.**

Ітерація № 0.  
Поточний опорний план неоптимальний, тому що в індексному рядку знаходяться негативні коефіцієнти.  
В якості ведучого виберемо стовпець, відповідний змінної x2, так як це найбільший коефіцієнт за модулем.  
Обчислимо значення Di по рядках як частка від ділення: bi / ai2  
і з них виберемо найменше:  
Отже, 2-ий рядок є провідною.  
Дозволяє елемент дорівнює (2) і стоїть на перетині провідного стовпця і ведучою рядка

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Базис | B | x1 | x2 | x3 | x4 | x5 | min |
| x3 | 6 | -3 | 2 | 1 | 0 | 0 | 3 |
| x5 | 3 | 1 | **2** | 0 | -1 | 1 | **11/2** |
| F(X1) | -3M | -1-1M | **-1-2M** | 0 | 1M | 0 | 0 |

Отримуємо нову симплекс-таблицю:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Базис | B | x1 | x2 | x3 | x4 | x5 |
| x3 | 3 | -4 | 0 | 1 | 1 | -1 |
| x2 | 11/2 | 1/2 | 1 | 0 | -1/2 | 1/2 |
| F(X1) | 11/2 | -1/2 | 0 | 0 | -1/2 | 1/2+1M |

Ітерація № 1.  
Поточний опорний план неоптимальний, тому що в індексному рядку знаходяться негативні коефіцієнти.  
В якості ведучого виберемо стовпець, відповідний змінної x4, так як це найбільший коефіцієнт за модулем.  
Обчислимо значення Di по рядках як частка від ділення: bi / ai4  
і з них виберемо найменше:  
Отже, 1-а рядок є провідною.  
Дозволяє елемент дорівнює (1) і стоїть на перетині провідного стовпця і ведучою рядка.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Базис | БП | x 1 | x 2 | x 3 | x 4 | x 5 | x 6 | z 1 |
| x3 | 3 | -4 | 0 | 1 | 1 | 0 | 0 | -1 |
| x2 | 3/2 | 1/2 | 1 | 0 | -1/2 | 0 | 0 | 1/2 |
| x5 | 3 | 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| x6 | 7/2 | -1/2 | 0 | 0 | 1/2 | 0 | 1 | -1/2 |
| ИС | 3/2 | -1/2 | 0 | 0 | -1/2 | 0 | 0 | M+1/2 |

Отримуємо нову симплекс-таблицю:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Базис | БП | x 1 | x 2 | x 3 | x 4 | x 5 | x 6 | z 1 |
| x3 | 15 | 0 | 8 | 1 | -3 | 0 | 0 | 3 |
| x1 | 3 | 1 | 2 | 0 | -1 | 0 | 0 | 1 |
| x5 | 0 | 0 | -2 | 0 | 1 | 1 | 0 | -1 |
| x6 | 5 | 0 | 1 | 0 | 0 | 0 | 1 | 0 |
| ИС | 3 | 0 | 1 | 0 | -1 | 0 | 0 | M+1 |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Базис | БП | x 1 | x 2 | x 3 | x 4 | x 5 | x 6 | z 1 |
| x3 | 15 | 0 | 2 | 1 | 0 | 3 | 0 | 0 |
| x1 | 3 | 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| x4 | 0 | 0 | -2 | 0 | 1 | 1 | 0 | -1 |
| x6 | 5 | 0 | 1 | 0 | 0 | 0 | 1 | 0 |
| ИС | 3 | 0 | -1 | 0 | 0 | 1 | 0 | M |

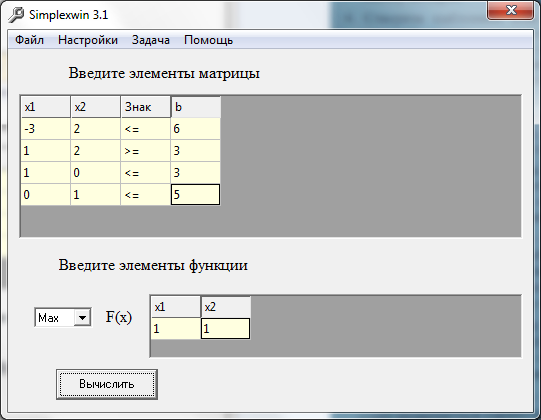
Остаточний варіант симплекс-таблиці:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Базис | БП | x 1 | x 2 | x 3 | x 4 | x 5 | x 6 | z 1 |
| x3 | 5 | 0 | 0 | 1 | 0 | 3 | -2 | 0 |
| x1 | 3 | 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| x4 | 10 | 0 | 0 | 0 | 1 | 1 | 2 | -1 |
| x2 | 5 | 0 | 1 | 0 | 0 | 0 | 1 | 0 |
| ИС | 8 | 0 | 0 | 0 | 0 | 1 | 1 | M |

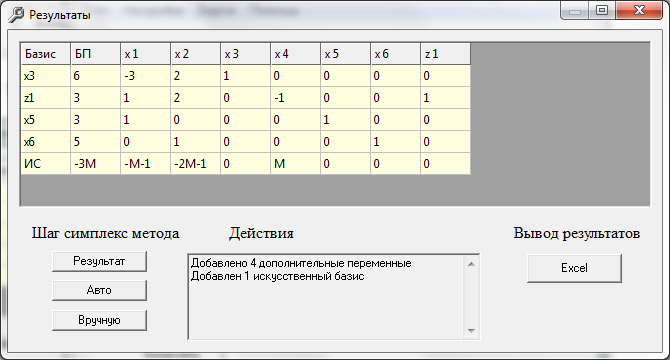
3. Розв’язання з допомогою пакету програм SimplexWin

Cимплекс метод з додаванням змінних , розв’язаний за допомогою сиплекс таблиць .

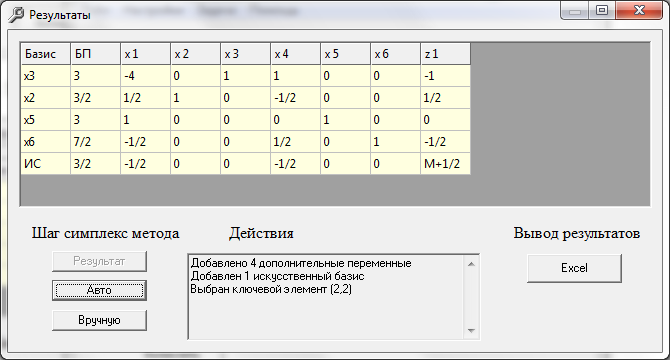
Ввід даних :



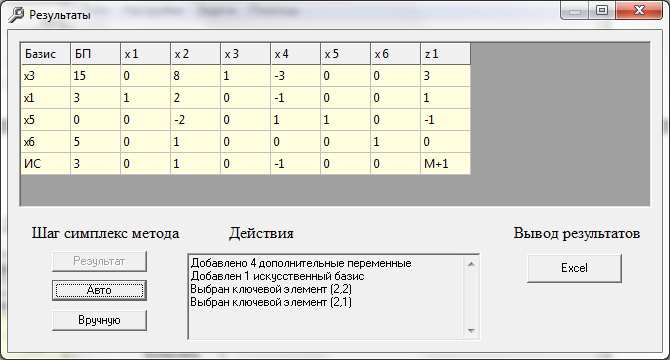
Крок 1:



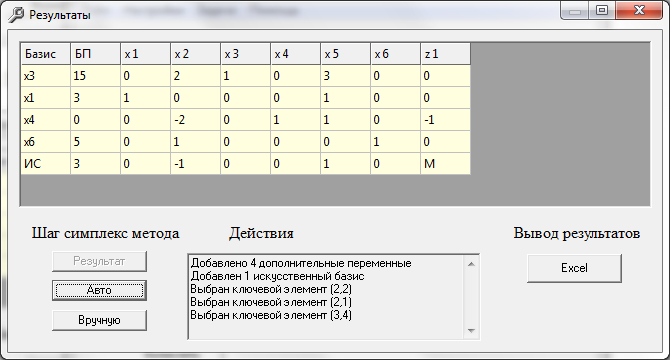
Крок 2:



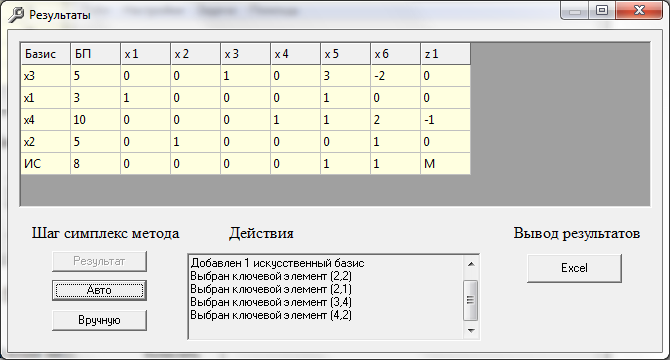
Крок 3:

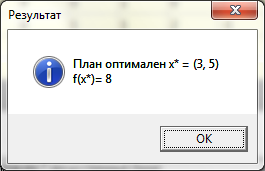


Крок 4:



Крок 5:





4. Розв’язання за допомогою власної програми.

Програма написана мовою С++

Для правильного вирішення система лінійних обмежень повинна бути приведена до канонічного виду.

При запуску програми потрібно ввести кількість рівнянь обмежень (rows) і найбільший індекс змінної (cols).

Після цього потрібно послідовно ввести коефіцієнти рівнянь (xi) та їх праві частини (b), а також коефіцієнти цільової функції. Якщо який-небудь коефіцієнт відсутній, слід ввести нуль.

Для виведення результату потрібно натиснути пробіл.

Якщо для вирішення завдання були введені додаткові перемінні або штучні базиси, то їх потрібно виключити з підсумкового відповіді.

## Лістинг програми main.cpp

#include "stdafx.h"

#include <iostream>

#include <cmath>

#include <fstream>

#include <sstream>

#include <string>

#include "user\_data.h"

#include "simplex.h"

using std::cout;

using std::endl;

void simplex::init()

{

int i, j;

func = 0;

sv = new double \*[num\_l];

for (i = 0; i < num\_l; i++)

sv[i] = new double [num\_v \* 2];

for (i = 0; i < num\_l; i++) {

for (j = 0; j < num\_v; j++)

sv[i][j] = system[i][j];

for (; j < num\_v \* 2; j++)

if (i + num\_v == j)

if (way)

sv[i][j] = 1;

else

sv[i][j] = -1;

else

sv[i][j] = 0;

}

istr = new double [num\_v \* 2];

bv = new double \*[num\_l];

for (i = 0; i < num\_l; i++) {

bv[i] = new double [2];

bv[i][0] = i + num\_v;

bv[i][1] = fm[i];

}

for (i = 0; i < num\_v \* 2; i++)

if (i < num\_v)

istr[i] = function[i] \* -1;

else

istr[i] = 0;

i\_lcol = 0;

for (i = 0; i < num\_v \* 2 - 1; i++) {

if (istr[i] < 0)

if (fabs(istr[i + 1]) > fabs(istr[i]))

i\_lcol = i + 1;

}

th = new double [num\_l];

for (i = 0; i < num\_l; i++)

th[i] = bv[i][1] / sv[i][i\_lcol];

i\_lrow = 0;

for (i = 0; i < num\_l - 1; i++)

if (th[i] > th[i + 1])

i\_lrow = i + 1;

alm = sv[i\_lrow][i\_lcol];

print\_result\_to\_file(0);

}

bool simplex::plane\_is\_valid()

{

int i;

bool result = true;

if (way)

for (i = 0; i < num\_v \* 2; i++)

if (istr[i] < 0) {

result = false;

break;

}

if (!way)

for (i = 0; i < num\_v \* 2; i++)

if (istr[i] >= 0) {

result = false;

break;

}

return result;

}

bool simplex::function\_is\_undefined()

{

int i;

for (i = 0; i < num\_l; i++)

if (th[i] < 0) {

return false;

}

return true;

}

void simplex::gen\_plane()

{

int i, j, it\_num = 0;

double A, B;

while (!plane\_is\_valid() && function\_is\_undefined()) {

A = bv[i\_lrow][1];

B = istr[i\_lcol];

func -= A \* B / alm;

double \*tmp\_bv = new double [num\_l];

bv [i\_lrow][0] = i\_lcol;

A = bv[i\_lrow][1];

for (i = 0; i < num\_l; i++) {

B = sv[i][i\_lcol];

tmp\_bv[i] = bv[i\_lrow][1];

if (i != i\_lrow)

tmp\_bv[i] = bv[i][1] - A \* B / alm;

else

tmp\_bv[i] /= alm;

}

for (i = 0; i < num\_l; i++)

bv[i][1] = tmp\_bv[i];

double \*tmp\_istr = istr;

B = istr[i\_lcol];

for (i = 0; i < num\_v \* 2; i++) {

A = sv[i\_lrow][i];

tmp\_istr[i] = istr[i] - A \* B / alm;

}

istr = tmp\_istr;

double \*\*tmp\_sv = new double \*[num\_l];

for (i = 0; i < num\_l; i++)

tmp\_sv[i] = new double [num\_v \* 2];

for (i = 0; i < num\_l; i++)

for (j = 0; j < num\_v \* 2; j++) {

tmp\_sv[i][j] = sv[i][j];

A = sv[i\_lrow][j];

B = sv[i][i\_lcol];

if (i == i\_lrow)

tmp\_sv[i][j] /= alm;

else

tmp\_sv[i][j] = sv[i][j] - A \* B / alm;

}

sv = tmp\_sv;

i\_lcol = 0;

for (i = 0; i < num\_l; i++)

th[i] = bv[i][1] / sv[i][i\_lcol];

i\_lrow = 0;

for (i = 0; i < num\_l -1; i++)

if (th[i] > th[i + 1])

i\_lrow = i + 1;

alm = sv[i\_lrow][i\_lcol];

it\_num++;

print\_result\_to\_file(it\_num);

}

if (!function\_is\_undefined())

cout << "zZilova funkzija ne obmeshena , dana zadasha nemaje rishenn" << endl;

else {

cout << "f(x) = " << func << "" << endl;

for (i = 0; i < num\_l; i++) {

cout << "x" << bv[i][0] + 1 << " = " << bv[i][1] << endl;

}

cout << "Resultati zapisani u table.txt" << endl;

}

}

void simplex::print\_result\_to\_file(int it\_num)

{

int i, j;

if (!it\_num) {

table << "Задана цільова функція:" << endl;

std::stringstream f\_x;

f\_x << "f(x) = ";

for (i = 0; i < num\_v; i++) {

if (!i)

f\_x << function[i] << "x" << i + 1 << " ";

else {

if (function[i] < 0)

f\_x << "- " << fabs(function[i]) << "x" << i + 1 << " ";

if (function[i] > 0)

f\_x << "+ " << function[i] << "x" << i + 1 << " ";

}

}

std::string minmax;

if (way)

minmax = "max";

else

minmax = "min";

f\_x << "=> " << minmax << "" << endl;

table << f\_x.str();

table << "І система рівнянь :" << endl;

std::stringstream math\_sys;

std::string math\_sign;

for (i = 0; i < num\_l; i++) {

for (j = 0; j < num\_v; j++) {

if (!j)

math\_sys << system[i][j] << "x" << j + 1 << " ";

else

if (system[i][j] == 1)

if (!j)

math\_sys << "x" << j + 1 << " ";

else

math\_sys << "+ x" << j + 1 << " ";

else

if (system[i][j] == -1)

if (!j)

math\_sys << "-x" << j + 1 << " ";

else

math\_sys << "- x" << j + 1 << " ";

else {

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

math\_sys << "- " << fabs(system[i][j])<< "x" << j + 1 << " ";

else

math\_sys << "+ " << system[i][j] << "x" << i + 1 << " ";

if (!sign[i])

math\_sign = "<=";

if (sign[i] == 1)

math\_sign = "=";

if (sign[i] == 2)

math\_sign = ">=";

}

}

math\_sys << math\_sign << " " << fm[i];

math\_sys << endl;

}

std::string min\_or\_max;

if (way)

min\_or\_max = "максимум";

else

min\_or\_max = "мінімум";

table << math\_sys.str() << endl;

table << "Вирішимо дану задачу на " << min\_or\_max << " методом симплексних таблиц.Побудуємо перший опорний план (таблицю):" << endl;

}

for (i = 0; i < num\_l; i++) {

table << "x" << bv[i][0] + 1 << " " << bv[i][1] << " ";

for (j = 0; j < num\_v \* 2; j++)

table << sv[i][j] << " ";

if (!plane\_is\_valid())

table << th[i];

table << "" << endl;

}

table << "f(x) " << func << " ";

for (i = 0; i < num\_v \* 2; i++)

table << istr[i] << " ";

table << "";

if (plane\_is\_valid()) {

if (plane\_is\_valid() && function\_is\_undefined())

table << "Danij plan e optimalnim i ne potrrebuje pokrashennja. Rishennja najdeno." << endl;

std::ofstream outfile ("table.txt");

outfile << table.str();

}

else {

std::string ln\_or\_gn;

if (way)

ln\_or\_gn = "nehativne";

else

ln\_or\_gn = "pozetivne";

std::stringstream num\_of\_plane;

if (!it\_num)

num\_of\_plane << "1 opornij plan";

else

num\_of\_plane << it\_num + 1 << "-j plan takosh";

table << "" << num\_of\_plane.str() << " ne javlajetsja optimalnim , oskilki v indeksnomu rjadku znahodjantsja " << ln\_or\_gn << " elementi.Yoho neobhidno pokrash4iti." << endl;

}

}

int \_tmain(int argc, \_TCHAR\* argv[])

{

setlocale(LC\_ALL, "Russian");

simplex \*ud = new simplex;

ud->get\_data\_from\_user();

ud->init();

ud->gen\_plane();

return 0;

}

## Лістинг програми user\_data.cpp

#include "stdafx.h"

#include <iostream>

#include <string>

#include <cstdlib>

#include "user\_data.h"

#include <cmath>

#include <fstream>

#include <sstream>

#include "simplex.h"

using namespace std;

void error(int err\_no)

{

switch(err\_no) {

case 0:

cout << "Vi vveli nekorektne znashennja." << endl;

break;

case 1:

cout << "Vi ne moshete zadati menshe 2 rivnjan." << endl;

break;

case 2:

cout << "Vi ne moshete zadati bilshe 500 rivnjan." << endl;

break;

case 3:

cout << "Vi ne moshete zadati menshe 2 zminnih." << endl;

break;

case 4:

cout << "Vi ne moshete zadati bilshe 500 rivnjan." << endl;

break;

}

}

void user\_data::get\_data\_from\_user()

{

string num\_limits, num\_vars, s\_var, fr\_m, sn, func, w;

int i, j;

bool validator = false;

do {

cout << "Vvedit kilkist rivnjan sistemi: ";

getline(cin, num\_limits);

if (atoi(num\_limits.c\_str()) < 2)

error(1);

else if (atoi(num\_limits.c\_str()) > 500)

error(2);

else

validator = true;

} while (!validator);

num\_l = atoi(num\_limits.c\_str());

validator = false;

do {

cout << "Vvedit kilkist zminnih pri zilovij funkziji: ";

getline(cin, num\_vars);

if (atoi(num\_vars.c\_str()) < 2)

error(3);

else if (atoi (num\_vars.c\_str()) > 500)

error(4);

else

validator = true;

} while (!validator);

num\_v = atoi(num\_vars.c\_str());

validator = false;

function = new double [num\_v];

system = new double \*[num\_l];

for (i = 0; i < num\_l; i++)

system[i] = new double [num\_v];

fm = new double [num\_l];

sign = new int [num\_l];

cout << "Zapovnit koephizienty pry zilovij funkzii" << endl;

for (i = 0; i < num\_v; i++) {

do {

cout << " Vvedit koephizienti zilovoji funkziji pri x" << i + 1 << ": ";

getline(cin, func);

if (atof(func.c\_str()) == 0)

error(0);

else {

validator = true;

function[i] = atof(func.c\_str());

}

} while (!validator);

validator = false;

}

do {

cout << "Vvedit naprjamlenna zilovoji funkziji ( min, max ) : ";

getline(cin, w);

if (w == "max" || w == "MAX" || w == "min" || w == "MIN") {

validator = true;

if (w == "max" || w == "MAX")

way = true;

else

way = false;

}

else

error (0);

} while (!validator);

cout << "Zapovnit sistemu rivnjan" << endl;

for (i = 0; i < num\_l; i++) {

cout << "Zapovnit " << i + 1 << "-е rivnjannja." << endl;

for (j = 0; j < num\_v; j++) {

do {

cout << "Vvedit koephizient pri x" << j + 1 << ": ";

getline(cin, s\_var);

if (atof(s\_var.c\_str()) == 0)

error (0);

else {

validator = true;

}

} while (!validator);

system[i][j] = atof(s\_var.c\_str());

validator = false;

}

do {

cout << "Vvedit znak pry " << i + 1 << "-mu rivnjanni ( <=, =, >= ) : ";

getline(cin, sn);

if (sn == "<=" || sn == "=" || sn == ">=") {

validator = true;

if (sn == "<=")

sign[i] = 0;

if (sn == "=")

sign[i] = 1;

if (sn == ">=")

sign[i] = 2;

}

else

error(0);

} while (!validator);

validator = false;

do {

cout << "Vvedit znashennja vilnoho chlena (Bi) pri " << i + 1 << "-mu rivnjanni: ";

getline(cin, fr\_m);

if (atof(fr\_m.c\_str()) == 0)

error(0);

else

validator = true;

} while (!validator);

fm[i] = atof(fr\_m.c\_str());

validator = false;

cout << endl;

}

}

Лістинг хідер файлу simplex.h

#ifndef \_SIMPLEX\_H\_

#define \_SIMPLEX\_H\_

#include <sstream>

#include "stdafx.h"

#include "user\_data.h"

class simplex : public user\_data {

public:

void init();

void gen\_plane();

bool plane\_is\_valid();

bool function\_is\_undefined();

void print\_result\_to\_file(int it\_num);

private:

double func;

double \*\*bv;

double \*\*sv;

double \*istr;

double \*th;

double alm;

int i\_lrow;

int i\_lcol;

std::stringstream table;

};

#endif /\* \_SIMPLEX\_H\_ \*/

Лістинг хідер файлу user\_data.h

#ifndef \_USER\_DATA\_H\_

#define \_USER\_DATA\_H\_

#include "stdafx.h"

class user\_data {

public:

void get\_data\_from\_user();

void user\_data\_is\_valid();

protected:

double \*function;

double \*fm;

double \*\*system;

int \*sign;

int num\_v;

int num\_l;

bool way;

};

#endif /\* \_USER\_DATA\_H\_ \*/

**Висновок:** на цій лабораторній роботі я набув навиків розв’язку задачі лінійного програмування симплекс-методом.

**Міністерство освіти та науки, молоді та спорту України**

**НУ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»**

Кафедра АСУ



**Лабораторна робота №3-4**

з дисципліни “*Математичні методи дослідження операцій*”

**Виконав:**

студент групи КН-22

Саврук Сергій

**Прийняв:**

старший викладач

Балич Б. І.

Львів – 2012