Министерство сельского хозяйства Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ АГРАРНЫЙ УНИВЕРСИТЕТ имени И.Т. ТРУБИЛИНА»

Кафедра системного анализа и обработки информации

**Отчет**

по учебной практике

по получению первичных профессиональных умений и навыков, в том числе первичных умений и навыков научно-исследовательской деятельности («Программирование»)

Выполнил студент

1 курса БИ1602

Белоусов Юрий Эдуардович

Руководитель: Кузьмина Э.В.

Краснодар 2017

**СОДЕРЖАНИЕ**

[Практическое задание №1. Использование регрессионного и корреляционного анализа для прогнозирования экономических показателей 3](#_Toc487719397)

[Практическое задание №2. Моделирование систем поддержки принятия решений. Теория принятия решений при полной определенности 4](#_Toc487719398)

[Практическое задание №3. Программирование систем поддержки принятия решений в условиях неопределенности 5](#_Toc487719399)

[Практическое задание №4. Решение экономических задач. 6](#_Toc487719400)

[Задача 1. Составление плана погашения кредита 6](#_Toc487719401)

[Задача 2. Расчёт цены единицы товара 6](#_Toc487719402)

[Практическое задание №5. Расчет производственных затрат 7](#_Toc487719403)

[Задача 1. Расчёт общих затрат предприятия при выпуске продукции 7](#_Toc487719404)

[Задача 2. Расчёт издержек производства продукции предприятия 7](#_Toc487719405)

[Список использованной литературы 8](#_Toc487719406)

# **Практическое задание №1. Использование регрессионного и корреляционного анализа для прогнозирования экономических показателей**

**Содержание задания**

Задание: написать программу на языке С++ для нахождения уравнение регрессии и построения графика, отображающего корреляционное поле и линию регрессии.

Имеются следующие данные разных стран об индексе розничных цен на продукты питания (х) и об индексе промышленного производства (у).

Таблица 1 - сведения об индексе розничных цен на продукты питания (х) и об индексе промышленного производства (у).

|  |  |  |
| --- | --- | --- |
| i | x | y |
| 1 | 100 | 70 |
| 2 | 105 | 79 |
| 3 | 108 | 85 |
| 4 | 113 | 84 |
| 5 | 118 | 85 |
| 6 | 118 | 85 |
| 7 | 110 | 96 |
| 8 | 115 | 99 |
| 9 | 119 | 100 |
| 10 | 118 | 98 |
| 11 | 120 | 99 |
| 12 | 124 | 102 |
| 13 | 129 | 105 |
| 14 | 132 | 112 |

**Код программы**

//---------------------------------------------------------------------------

#include <vcl.h>

#pragma hdrstop

#include "Unit1.h"

#include "Math.h"

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma link "Chart"

#pragma link "TeEngine"

#pragma link "TeeProcs"

#pragma link "BubbleCh"

#pragma link "Series"

#pragma resource "\*.dfm"

//---------------------------------------------------------------------------

TForm1 \*Form1;

const int n = 14;

int X[n] = {100, 105, 108, 113, 118, 118, 110, 115, 119, 118, 120, 124, 129, 132};

int Y[n] = {70, 79, 85, 84, 85, 85, 96, 99, 100, 98, 99, 102, 105, 112};

double sredX= 0, sredY = 0;

int xMin = 100;

int xMax = 132;

double dX, dY, sX, sY;

double R;

//---------------------------------------------------------------------------

\_\_fastcall TForm1::TForm1(TComponent\* Owner)

: TForm(Owner)

{

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::Button1Click(TObject \*Sender)

{

setDefault();

fillXY();

countAverageValue();

countDesperssion();

countRegression();

drawLineRegress();

drawBubbleField();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::Button2Click(TObject \*Sender)

{

printInfo();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::FormCreate(TObject \*Sender)

{

Memo1->Text = "";

fillGrid();

}

//---------------------------------------------------------------------------

void TForm1::drawBubbleField()

{

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

{

Series2->AddBubble(X[i], Y[i], 1, i, clTeeColor);

}

}

//---------------------------------------------------------------------------

void TForm1::countAverageValue()

{

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

{

sredX += X[i];

sredY += Y[i];

}

sredX /= n;

sredY /= n;

}

//---------------------------------------------------------------------------

void TForm1::countDesperssion()

{

int sumX = 0;

int sumY = 0;

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

{

sumX += pow(X[i] - sredX, 2);

sumY += pow(Y[i] - sredY, 2);

}

dX = sumX / n;

dY = sumY / n;

sX = sqrt(dX);

sY = sqrt(dY);

}

//---------------------------------------------------------------------------

void TForm1::countRegression()

{

int sumXY = 0;

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

sumXY += X[i] \* Y[i];

R = (sumXY - (n \* sredX \* sredY)) / (n \* sX \* sY);

}

//---------------------------------------------------------------------------

void TForm1::drawLineRegress()

{

float x = xMin;

float y = 0;

while (x < xMax)

{

y = R \* (sX / sY) \* (x - sredX) + sredY;

Series1->AddXY(x, y, "", clTeeColor);

x += 0.01;

}

}

//---------------------------------------------------------------------------

void TForm1::printInfo()

{

Memo1->Lines->Add("Среднее X = " + FloatToStr(sredX));

Memo1->Lines->Add("Среднее Y = " + FloatToStr(sredY));

Memo1->Lines->Add("Дисперсия X = " + FloatToStr(dX));

Memo1->Lines->Add("Дисперсия Y = " + FloatToStr(dY));

Memo1->Lines->Add("Отклонение X = " + FloatToStr(sX));

Memo1->Lines->Add("Отклонение Y = " + FloatToStr(sY));

Memo1->Lines->Add("Корреляция = " + FloatToStr(R));

Memo1->Lines->Add("------------------------------------------");

}

//---------------------------------------------------------------------------

void TForm1::fillGrid()

{

StringGrid1->Cells[0][0] = "X";

StringGrid1->Cells[1][0] = "Y";

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

{

StringGrid1->Cells[0][i+1] = X[i];

StringGrid1->Cells[1][i+1] = Y[i];

}

}

//---------------------------------------------------------------------------

void TForm1::fillXY()

{

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

{

X[i] = StrToInt(StringGrid1->Cells[0][i+1]);

Y[i] = StrToInt(StringGrid1->Cells[1][i+1]);

}

}

//---------------------------------------------------------------------------

void TForm1::setDefault()

{

sredX= 0;

sredY = 0;

dX = 0;

dY = 0;

sX = 0;

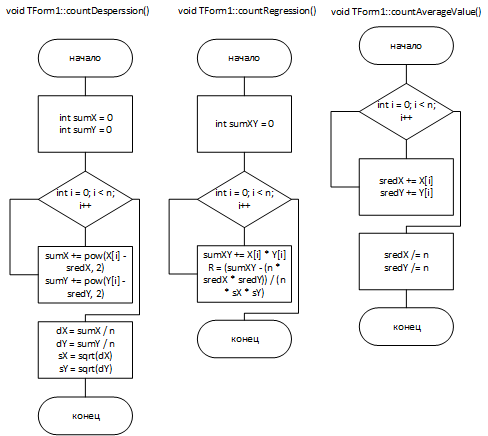
sY = 0;

R = 0;

}

//---------------------------------------------------------------------------

**Блок – схема**

****

**Результат работы**

Программа работает без ошибок. В процессе работы происходит нахождение уравнения регрессии и построение графика, отображающего корреляционное поле и линию регрессии.

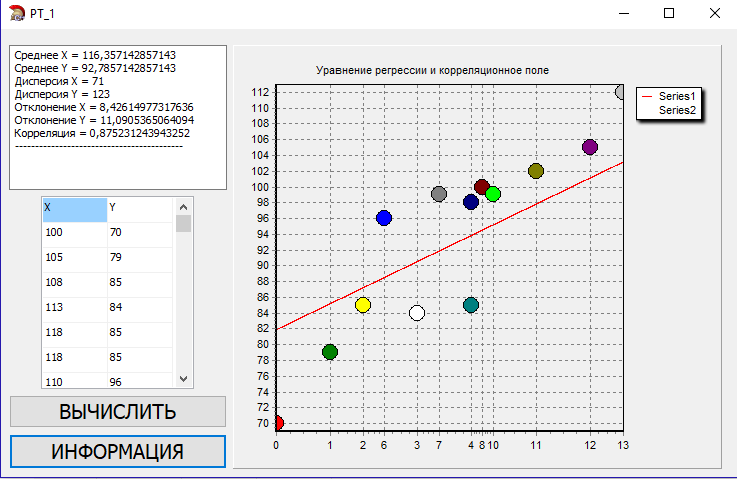


Рисунок 1– Результат работы программы

# **Практическое задание №2. Моделирование систем поддержки принятия решений. Теория принятия решений при полной определенности**

**Содержание задания**

Задание: написать программу на языке С++ для выбора наиболее эффективно работающего предприятия.

Показатели эффективности работы предприятий приведены в следующей таблице:

Таблица 2 - Показатели эффективности работы предприятий.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| №  Предприятия | Показатели эффективности работы предприятий | | | | |
| Прибыль,  Д.е. | Себестоимость единицы продукции,  Д.е. | Доходы  Д.е. | Фондоотдача  У.е. | Производительность  У.е. |
| 1 | 30,0 | 40,0 | 20,0 | 0,2 | 300 |
| 2 | 25,0 | 20,0 | 30,0 | 0,3 | 200 |
| 3 | 40,0 | 45,0 | 54,0 | 0,1 | 250 |
| 4 | 28,0 | 30,0 | 35,0 | 0,4 | 160 |
| 5 | 15,0 | 12,0 | 20,0 | 0,25 | 280 |
| 6 | 50,0 | 30,0 | 40,0 | 0,21 | 120 |
| Весовые коэффициенты | 0,32 | 0,23 | 0,15 | 0,20 | 0,10 |

Выберите наиболее эффективно работающее предприятие.

**Код программы**

//---------------------------------------------------------------------------

#include <vcl.h>

#pragma hdrstop

#include "Unit1.h"

#include "Math.h"

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma resource "\*.dfm"

//---------------------------------------------------------------------------

TForm1 \*Form1;

const int n = 6;

const int m = 5;

float maxF = 0;

float max[m];

float F[n];

float H[m] = {0.32, 0.23, 0.15, 0.20, 0.10};

float A[n][m] = {{30.0, 40.0, 20.0, 0.2, 300},

{25.0, 20.0, 30.0, 0.3, 200},

{40.0, 45.0, 54.0, 0.1, 250},

{28.0, 30.0, 35.0, 0.4, 160},

{15.0, 12.0, 20.0, 0.25, 280},

{50.0, 30.0, 40.0, 0.21, 120}};

//---------------------------------------------------------------------------

\_\_fastcall TForm1::TForm1(TComponent\* Owner)

: TForm(Owner)

{

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::FormCreate(TObject \*Sender)

{

StringGrid1->Visible = true;

StringGrid1->Cells[0][0] = "№ Предприятия";

StringGrid1->Cells[1][0] = "Прибыль";

StringGrid1->Cells[2][0] = "Себестоимость";

StringGrid1->Cells[3][0] = "Доходы";

StringGrid1->Cells[4][0] = "Фондотдача";

StringGrid1->Cells[5][0] = "Производительность";

StringGrid1->Cells[0][7] = "Коэффициенты";

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

StringGrid1->Cells[0][i] = IntToStr(i);

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

StringGrid1->ColWidths[i] = StringGrid1->Cells[i][0].Length()\*8;

Memo1->Text = "";

fillGrid();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::Button1Click(TObject \*Sender)

{

fillMatrix();

find();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::Button2Click(TObject \*Sender)

{

printInfo();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::Button3Click(TObject \*Sender)

{

normolize();

fillGrid();

}

//---------------------------------------------------------------------------

void TForm1::fillGrid()

{

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

{

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

{

StringGrid1->Cells[j+1][i+1] = FloatToStr(A[i][j]).SubString(0, 4);

}

StringGrid1->Cells[i+1][7] = FloatToStr(H[i]).SubString(0, 4);

}

}

//---------------------------------------------------------------------------

void TForm1::fillMatrix()

{

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

{

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

{

A[i][j] = StrToFloat(StringGrid1->Cells[j+1][i+1]);

}

H[i] = StrToFloat(StringGrid1->Cells[i+1][7]);

}

}

//---------------------------------------------------------------------------

void TForm1::normolize()

{

float sred = 0;

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

{

for (int j = 0; j < n; j++)

{

if (A[j][i] > max[i])

{

max[i] = A[j][i];

}

}

sred += max[i];

Memo1->Lines->Add("MAX." + IntToStr(i) + " = " + FloatToStr(max[i]));

}

sred /= m;

Memo1->Lines->Add("SRED = " + FloatToStr(sred));

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

{

for (int j = 0; j < n; j++)

{

if(max[i] >= sred)

{

A[j][i] = A[j][i] / max[i];

}

else

{

A[j][i] = 1 - (A[j][i] / max[i]);

}

}

}

}

//---------------------------------------------------------------------------

void TForm1::printInfo()

{

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

{

Memo1->Lines->Add("F" + IntToStr(i + 1) + " = " + FloatToStr(F[i]));

}

}

//---------------------------------------------------------------------------

void TForm1::find()

{

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

{

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

{

F[i] += H[j] \* A[i][j];

}

}

int index = 0;

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

{

if (F[i] > maxF)

{

maxF = F[i];

index = i + 1;

}

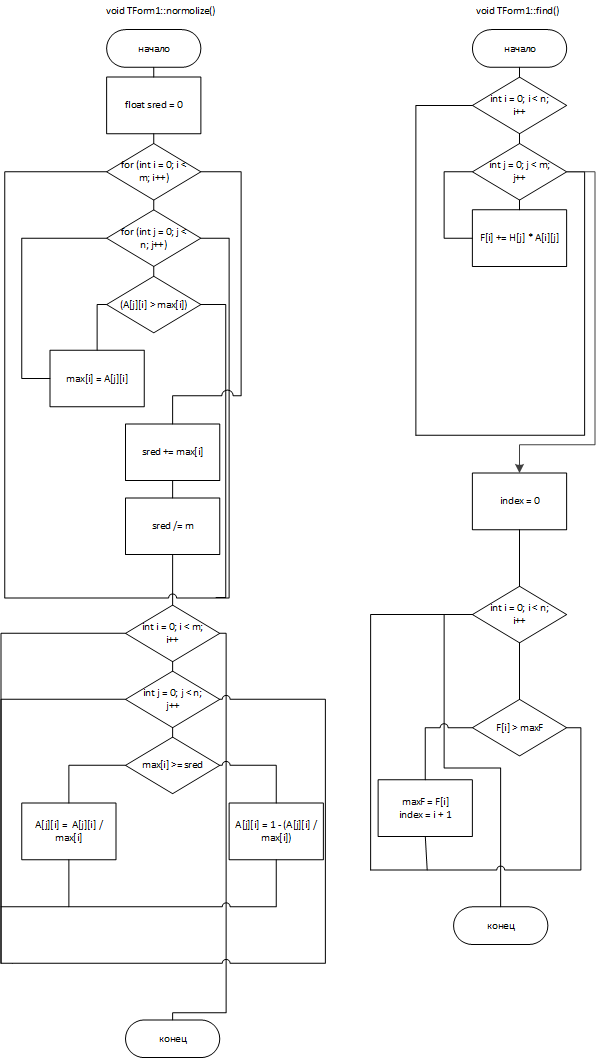
}

Label1->Caption = "Предприятие № " + IntToStr(index) + " :" + FloatToStr(maxF).SubString(0, 5);

}

//---------------------------------------------------------------------------

**Блок – схема**

****

**Результат работы**

Программа работает без ошибок. В процессе работы происходит нахождение наиболее эффективно работающего предприятия.

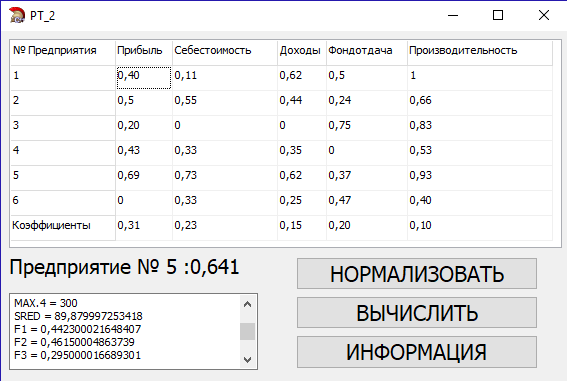
****

Рисунок 2– Результат работы программы

# **Практическое задание №3. Программирование систем поддержки принятия решений в условиях неопределенности**

**Содержание задания**

Задание: написать программу на языке С++ для выбора наиболее эффективного проекта легкового автомобиля для производства, используя критерии Лапласа, Вальда, Сэвиджа и Гурвица.

Намечается крупномасштабное производство легковых автомобилей. Имеются четыре варианта проекта автомобиля

Определена экономическая эффективность Vji каждого проекта в зависимости от рентабельности производства. По истечению трех сроков рассматриваются как некоторые состояния среды (природы). Значения экономической эффективности для различных проектов и состояний природы приведены в следующей таблице (д.е.):

Таблица 3- исходные данные.

|  |  |  |  |
| --- | --- | --- | --- |
| Проекты | Состояния природы | | |
| S1 | S2 | S3 |
| R1 | 20 | 25 | 15 |
| R2 | 25 | 24 | 10 |
| R3 | 15 | 28 | 12 |
| R | 9 | 30 | 20 |

Требуется выбрать лучший проект для производства, используя критерии Лапласа, Вальда, Сэвиджа и Гурвица при ɑ=0,1. Сравните решения и сделайте выводы.

**Код программы**

//---------------------------------------------------------------------------

#include <vcl.h>

#pragma hdrstop

#include "Unit1.h"

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma resource "\*.dfm"

//---------------------------------------------------------------------------

TForm1 \*Form1;

const int R = 4;

const int S = 3;

float a = 0.1;

float p = 0.25;

float res;

float minV;

float maxV;

int index;

int max[R];

int min[R];

int V[R][S] = {{20, 25, 15},

{25, 24, 10},

{15, 28, 12},

{9, 30, 20}};

//---------------------------------------------------------------------------

\_\_fastcall TForm1::TForm1(TComponent\* Owner)

: TForm(Owner)

{

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::FormCreate(TObject \*Sender)

{

StringGrid1->Visible = true;

StringGrid1->Cells[0][0] = "Проекты / Состояния";

StringGrid1->ColWidths[0] = 120;

for (int i = 1; i <= R; i++)

StringGrid1->Cells[0][i] = "R" + IntToStr(i);

for (int i = 1; i <= S; i++)

StringGrid1->Cells[i][0] = "S" + IntToStr(i);

Memo1->Text = "";

fillGrid();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::Button1Click(TObject \*Sender)

{

fillMatrix();

float W[R];

float minW;

printIfno(Button1->Caption + ":");

for (int i = 0; i < R; i++)

{

W[i] = 0;

for (int j = 0; j < S; j++)

{

W[i] += V[i][j];

}

W[i] = W[i] \* p;

if (W[i] < minW || i == 0)

{

minW = W[i];

index = i + 1;

}

printIfno("W{R" + IntToStr(i + 1) + "} = " + FloatToStr(W[i]));

}

printIfno("Min W = " + FloatToStr(minW));

printIfno("--------------------");

res = minW;

Label1->Caption = "R" + IntToStr(index) + " = " + FloatToStr(res);

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::Button2Click(TObject \*Sender)

{

fillMatrix();

findMaxAndMinR();

printIfno(Button2->Caption + ":");

int minR = 9999;

for (int i = 0; i < R; i++)

{

if (max[i] < minR || i == 0)

{

minR = max[i];

index = i + 1;

}

}

res = minR;

printIfno("Min R = " + FloatToStr(minR));

printIfno("--------------------");

Label2->Caption = "R" + IntToStr(index) + " = " + FloatToStr(res);

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::Button3Click(TObject \*Sender)

{

fillMatrix();

findMaxAndMinV();

findMaxAndMinR();

float r[R][S];

printIfno(Button3->Caption + ":");

printIfno("Matrix r :");

for (int i = 0; i < R; i++)

{

AnsiString str = "";

for (int j = 0; j < S; j++)

{

r[i][j] = V[i][j] - minV;

str += FloatToStr(r[i][j]) + " ";

}

printIfno(str);

}

printIfno("--------------------");

for (int i = 0; i < R; i++)

{

int rMax = max[i] - minV;

printIfno("r" + IntToStr(i + 1) + " max = " + IntToStr(rMax));

}

printIfno("--------------------");

float minR = 9999;

for (int i = 0; i < R; i++)

{

if ((max[i]-minV) < minR || i == 0)

{

minR = max[i]-minV;

index = i + 1;

}

}

res = minR;

Label3->Caption = "R" + IntToStr(index) + " = " + FloatToStr(res);

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::Button4Click(TObject \*Sender)

{

fillMatrix();

findMaxAndMinR();

printIfno(Button4->Caption + ":");

float minW = 9999;

for (int i = 0; i < R; i++)

{

float W = (a \* min[i]) + ((1 - a) \* max[i]);

if (W < minW)

{

minW = W;

index = i;

}

printIfno("W" + IntToStr(i + 1) + " = " + FloatToStr(W));

}

res = minW;

printIfno("W min = " + FloatToStr(minW));

Label4->Caption = "R" + IntToStr(index) + " = " + FloatToStr(res);

}

//---------------------------------------------------------------------------

void TForm1::printIfno(AnsiString str)

{

Memo1->Lines->Add(str);

}

//---------------------------------------------------------------------------

void TForm1::fillGrid()

{

for (int i = 0; i < R; i++)

{

for (int j = 0; j < S; j++)

{

StringGrid1->Cells[j+1][i+1] = V[i][j];

}

}

}

//---------------------------------------------------------------------------

void TForm1::fillMatrix()

{

for (int i = 0; i < R; i++)

{

for (int j = 0; j < S; j++)

{

V[i][j] = StrToInt(StringGrid1->Cells[j+1][i+1]);

}

}

}

//---------------------------------------------------------------------------

void TForm1::findMaxAndMinR()

{

for (int i = 0; i < R; i++)

{

for (int j = 0; j < S; j++)

{

if (V[i][j] > max[i])

{

max[i] = V[i][j];

}

if (V[i][j] < min[i] || j == 0)

{

min[i] = V[i][j];

}

}

printIfno("R" + IntToStr(i + 1) + " max = " + IntToStr(max[i]));

printIfno("R" + IntToStr(i + 1) + " min = " + IntToStr(min[i]));

}

printIfno("--------------------");

}

//---------------------------------------------------------------------------

void TForm1::findMaxAndMinV()

{

minV = 9999;

for (int i = 0; i < R; i++)

{

for (int j = 0; j < S; j++)

{

if (V[i][j] > maxV)

maxV = V[i][j];

if (V[i][j] < minV)

minV = V[i][j];

}

}

printIfno("maxV = " + FloatToStr(maxV));

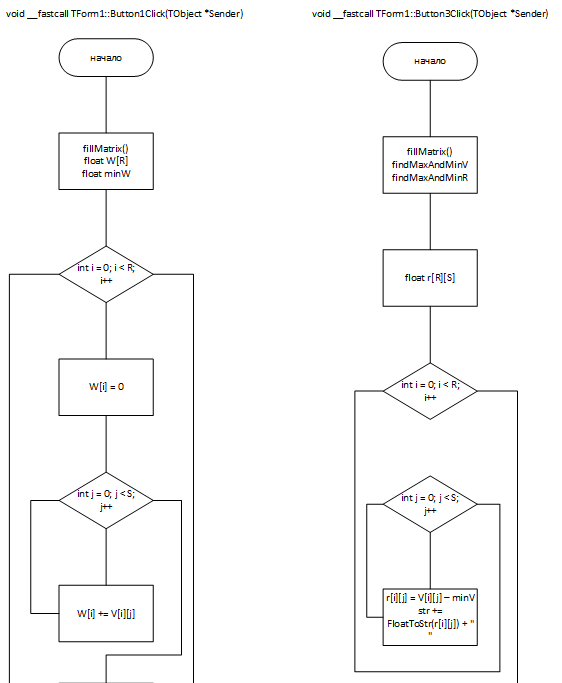
printIfno("minV = " + FloatToStr(minV));

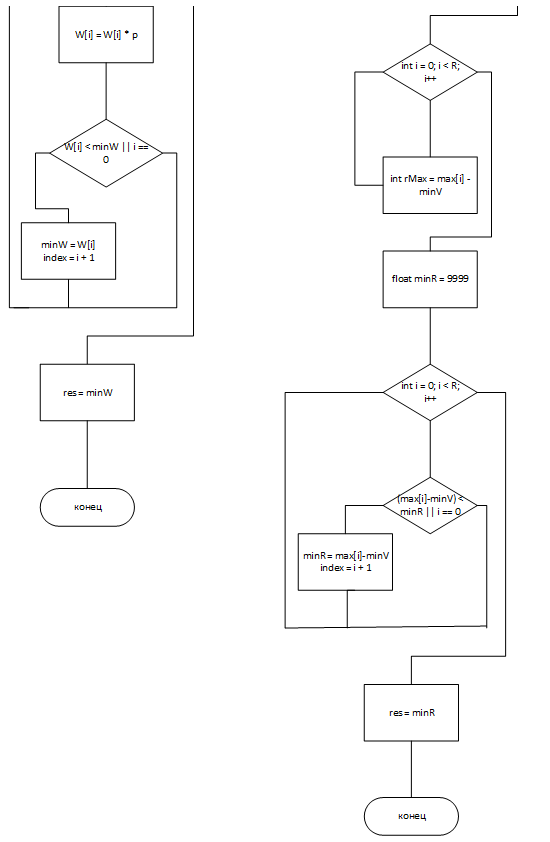
printIfno("--------------------");

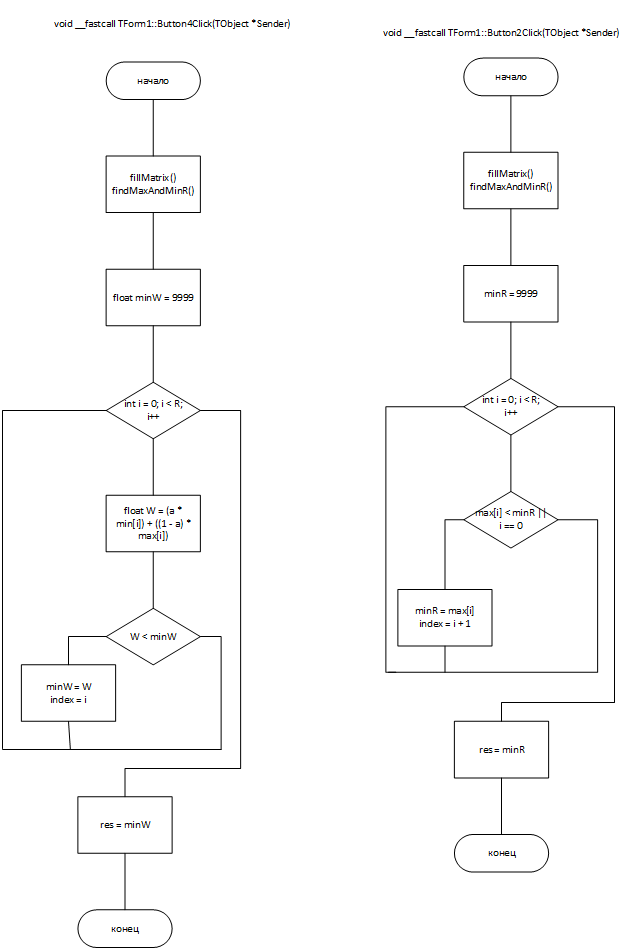
}

//---------------------------------------------------------------------------

**Блок – схема**

****

****

****

**Результат работы**

Программа работает без ошибок. В процессе работы происходит нахождение эффективного проекта легкового автомобиля для производства, используя критерии Лапласа, Вальда, Сэвиджа и Гурвица.

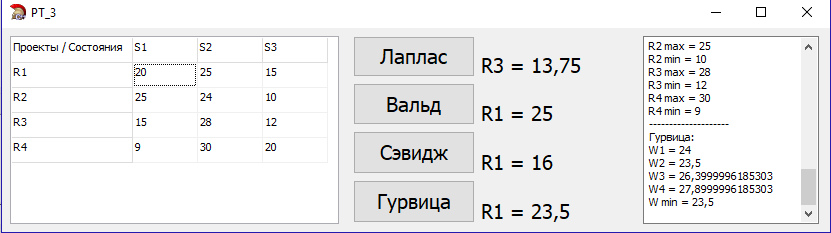
****

Рисунок 3– Результат работы программы

# **Практическое задание №4. Решение экономических задач.**

## **Задача 1. Составление плана погашения кредита**

**Содержание задания**

Задание: Сбербанк РФ предоставляет потребительский кредит размером 120 тыс. руб. на 12 месяцев под 13,5% годовых. Долг погашается ежемесячно равными частями, проценты начисляются на остаток долга и выплачиваются ежемесячно. Составьте план погашения кредита. Составить блок-схему и написать программу на языке С++ для решения задачи погашения кредита.

**Код программы**

//---------------------------------------------------------------------------

#include <vcl.h>

#pragma hdrstop

#include "Unit1.h"

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma resource "\*.dfm"

//---------------------------------------------------------------------------

TForm1 \*Form1;

float D, m, n, I;

int row, col;

//---------------------------------------------------------------------------

\_\_fastcall TForm1::TForm1(TComponent\* Owner)

: TForm(Owner)

{

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::FormCreate(TObject \*Sender)

{

Edit1->Text = "120000";

Edit2->Text = "12";

Edit3->Text = "1";

Edit4->Text = "13,5";

readOfEdits();

updateCells();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::Button1Click(TObject \*Sender)

{

readOfEdits();

updateCells();

float monthSum = D / (n \* m);

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

{

float pay = (D - (i - 1) \* monthSum) \* (I / 100) / m;

StringGrid1->Cells[1][i] = FloatToStr(D - (i \* monthSum));

StringGrid1->Cells[2][i] = FloatToStr(pay);

StringGrid1->Cells[3][i] = FloatToStr(monthSum);

StringGrid1->Cells[4][i] = FloatToStr(pay + StrToFloat(StringGrid1->Cells[3][i]));

}

float tmp;

for (int i = 2; i <= 4; i++)

{

tmp = 0;

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

{

tmp += StrToFloat(StringGrid1->Cells[i][j]);

}

StringGrid1->Cells[i][m+1] = FloatToStr(tmp);

}

}

//---------------------------------------------------------------------------

void TForm1::readOfEdits()

{

D = StrToFloat(Edit1->Text);

m = StrToFloat(Edit2->Text);

n = StrToFloat(Edit3->Text);

I = StrToFloat(Edit4->Text);

m = m \* n;

}

//---------------------------------------------------------------------------

void TForm1::updateCells()

{

StringGrid1->RowCount = m + 2;

row = StringGrid1->RowCount;

col = StringGrid1->ColCount;

StringGrid1->Visible = true;

StringGrid1->Cells[0][0] = "Месяц";

StringGrid1->Cells[1][0] = "Непогашенный долг";

StringGrid1->Cells[2][0] = "Процентные платежи";

StringGrid1->Cells[3][0] = "Выплота долга";

StringGrid1->Cells[4][0] = "Сумма взноса";

StringGrid1->Cells[0][m+1] = "ИТОГО";

for (int i = 1; i < row - 1; i++)

StringGrid1->Cells[0][i] = IntToStr(i);

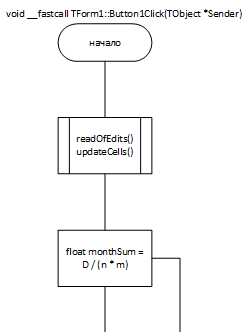
for (int i = 0; i < col; i++)

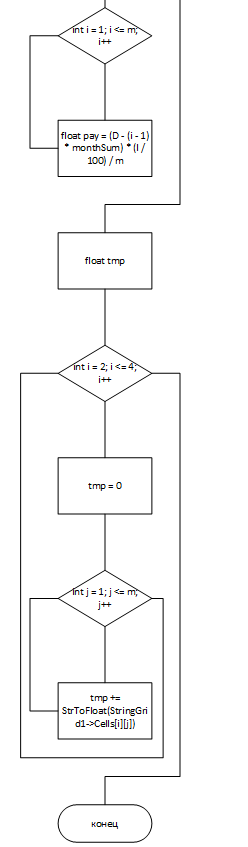
StringGrid1->ColWidths[i] = StringGrid1->Cells[i][0].Length()\*7;

}

//---------------------------------------------------------------------------

**Блок – схема**

****

****

**Результат работы**

Программа работает без ошибок. В процессе работы происходит составление плана погашения кредита

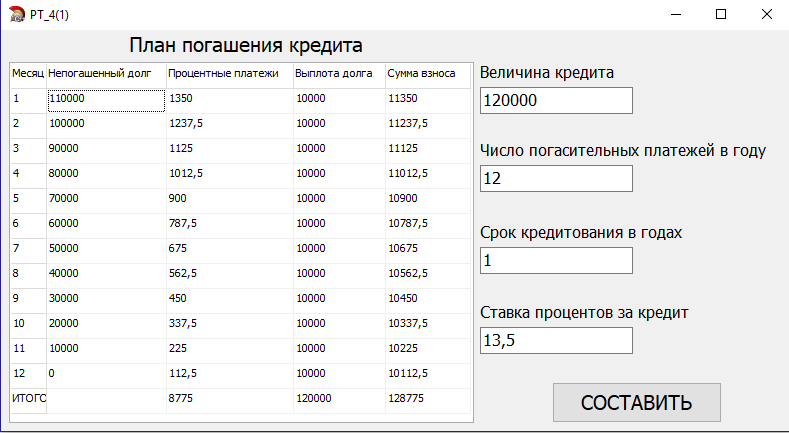


Рисунок 4– Результат работы программы

## **Задача 2. Расчёт цены единицы товара**

**Содержание задания**

Задание: Организация реализует 70 000 единиц продукции по цене 12 000 руб. за единицу. Переменные расходы на единицу составляют 7 000 руб., общая сумма постоянных затрат - 250 000 тыс. руб. В связи с увеличением спроса объем продаж может вырасти на 7%. В тоже время из-за инфляции ожидается повышение переменных затрат на 9%, постоянных - на 10%.

Какой уровень цены необходимо установить, чтобы компенсировать негативный инфляционный фактор и увеличить прибыль на 10%.Составить блок-схему и написать программу на языке С++ для решения данной задачи

**Код программы**

//---------------------------------------------------------------------------

#include <vcl.h>

#pragma hdrstop

#include "Unit1.h"

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TForm1 \*Form1;

double P, Q, AVC, FC;

double var, stat, vol;

double P1, P2, P3;

double res;

//---------------------------------------------------------------------------

\_\_fastcall TForm1::TForm1(TComponent\* Owner)

: TForm(Owner)

{

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::FormCreate(TObject \*Sender)

{

P = 12000;

Q = 70000;

AVC = 7000;

FC = 25000;

vol = 1.07;

var = 1.09;

stat = 1.1;

Edit1->Text = "12000";

Edit2->Text = "70000";

Edit3->Text = "7000";

Edit4->Text = "25000";

Edit5->Text = "7";

Edit6->Text = "9";

Edit7->Text = "10";

}

//---------------------------------------------------------------------------

void TForm1::readOfEdits()

{

P = StrToFloat(Edit1->Text);

Q = StrToFloat(Edit2->Text);

AVC = StrToFloat(Edit3->Text);

FC = StrToFloat(Edit4->Text);

vol = StrToFloat(Edit5->Text) / 100 + 1;

var = StrToFloat(Edit6->Text) / 100 + 1;

stat = StrToFloat(Edit7->Text) / 100 + 1;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::Button1Click(TObject \*Sender)

{

readOfEdits();

P1 = P \* Q - (AVC \* Q + FC);

P2 = P1 \* stat;

P3 = P \* Q \* vol - (AVC \* Q \* var + FC \* stat);

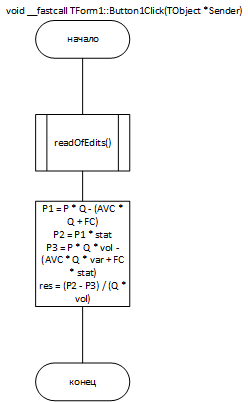
res = (P2 - P3) / (Q \* vol);

Label9->Caption = FloatToStr(res).SubString(0, 6) + " руб";

}

//---------------------------------------------------------------------------

**Блок – схема**

****

**Результат работы**

Программа работает без ошибок. В процессе работы происходит определение уровня цены необходимо установить, чтобы компенсировать негативный инфляционный фактор и увеличить прибыль на 10%.

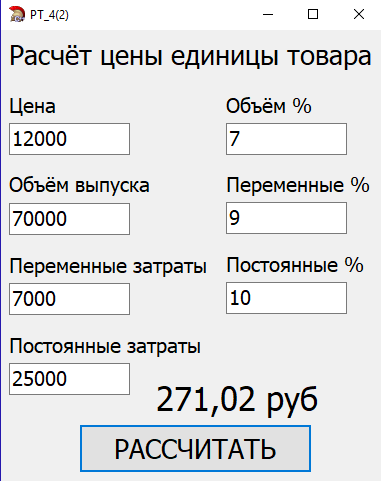


Рисунок 5– Результат работы программы

# **Практическое задание №5. Расчет производственных затрат**

## **Задача 1. Расчёт общих затрат предприятия при выпуске продукции**

**Содержание задания**

Задание: в таблице показана зависимость общих затрат предприятия от выпуска продукции. Рассчитайте затраты: постоянные, переменные, средние общие, средние постоянные, средние переменные. В таблице заполните графы FC, VC, MC, ATC, AFC, AVC:

Таблица 4- исходные данные.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Выпуск в единицу времени, Q, шт. | Общие затраты, TC, р. | FС | VVC | MC | ATC | AVC | AFC |
| 0 | 60 |  |  |  |  |  |  |
| 1 | 130 |  |  |  |  |  |  |
| 2 | 180 |  |  |  |  |  |  |
| 3 | 230 |  |  |  |  |  |  |
| 4 | 300 |  |  |  |  |  |  |

Составить блок-схему и написать программу на языке С++ для решения расчёта общих затрат предприятия.

**Код программы**

//---------------------------------------------------------------------------

#include <vcl.h>

#pragma hdrstop

#include "Unit1.h"

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TForm1 \*Form1;

const int n = 5;

const int m = 7;

int row, col;

int TC = 0, FC = 1, VC = 2, MC = 3, ATC = 4, AVC = 5, AFC = 6;

float M[n][m];

//---------------------------------------------------------------------------

\_\_fastcall TForm1::TForm1(TComponent\* Owner)

: TForm(Owner)

{

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::FormCreate(TObject \*Sender)

{

row = StringGrid1->RowCount;

col = StringGrid1->ColCount;

StringGrid1->Visible = true;

StringGrid1->Cells[0][0] = "Q";

StringGrid1->Cells[1][0] = "TC";

StringGrid1->Cells[2][0] = "FC";

StringGrid1->Cells[3][0] = "VC";

StringGrid1->Cells[4][0] = "MC";

StringGrid1->Cells[5][0] = "ATC";

StringGrid1->Cells[6][0] = "AVC";

StringGrid1->Cells[7][0] = "AFC";

M[0][TC] = 60;

M[1][TC] = 130;

M[2][TC] = 180;

M[3][TC] = 230;

M[4][TC] = 300;

for (int i = 1; i <= row; i++)

StringGrid1->Cells[0][i] = IntToStr(i - 1);

for (int i = 0; i < col; i++)

StringGrid1->ColWidths[i] = StringGrid1->Cells[i][0].Length()\*20;

fillGrid();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::Button1Click(TObject \*Sender)

{

fillMatrix();

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

{

M[i][FC] = M[0][TC];

M[i][VC] = M[i][TC] - M[i][FC];

M[i+1][MC] = M[i+1][TC] - M[i][TC];

if(i != 0)

{

M[i][ATC] = M[i][TC] / (i);

M[i][AVC] = M[i][VC] / (i);

M[i][AFC] = M[i][FC] / (i);

}

}

fillGrid();

}

//---------------------------------------------------------------------------

void TForm1::fillGrid()

{

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

{

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

{

StringGrid1->Cells[j+1][i+1] = FloatToStr(M[i][j]).SubString(0, 4);

}

}

}

//---------------------------------------------------------------------------

void TForm1::fillMatrix()

{

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

{

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

{

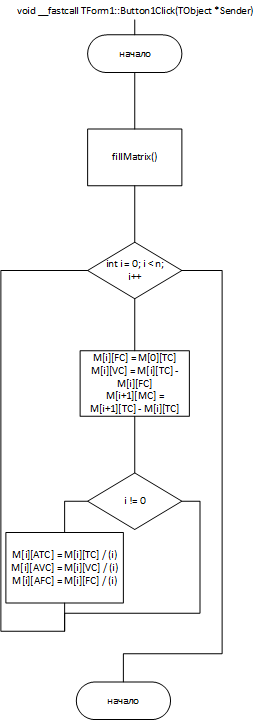
M[i][j] = StrToInt(StringGrid1->Cells[j+1][i+1]);

}

}

}

**Блок – схема**

****

**Результат работы**

Программа работает без ошибок. В процессе работы происходит расчёт затрат: постоянные, переменные, средние общие, средние постоянные, средние переменные.

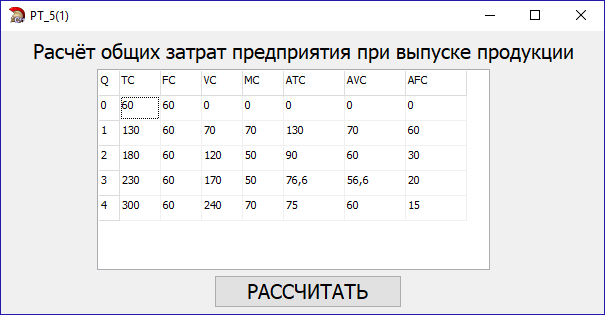


Рисунок 6– Результат работы программы

## **Задача 2. Расчёт издержек производства продукции предприятия**

**Содержание задания**

Задание: Информация о деятельности фирмы, кроме представленной в таблице, оказалась утраченной. Восстановите недостающую информацию об издержках фирмы.

Q – количество, TC – общие затраты, VC – переменные затраты, FC – фиксированные затраты, AC – средние затраты, AVC - средние переменные затраты, AFC – средние фиксированные затраты, MC – предельные затраты.

Таблица 5 - исходные данные.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Q** | **TC** | **VC** | **FC** | **AC** | **AVC** | **AFC** | **MC** |
| **1** | 30 |  |  |  |  |  |  |
| **2** |  | 18 |  |  |  |  |  |
| **3** |  |  |  | 15 |  |  |  |
| **4** |  |  |  |  | 7 |  |  |
| **5** |  |  |  |  |  | 4 | 2 |
|  |  |  |  |  | 3,5 |  | 1 |

Составить блок-схему и написать программу на языке С++ для решения расчёта цены единицы товара.

**Код программы**

//---------------------------------------------------------------------------

#include <vcl.h>

#pragma hdrstop

#include "Unit1.h"

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TForm1 \*Form1;

const int n = 6;

const int m = 7;

int row, col;

int Q;

int TC = 0, VC = 1, FC = 2, AC = 3, AVC = 4, AFC = 5, MC = 6;

float M[n][m];

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::FormCreate(TObject \*Sender)

{

Q = StrToInt(Edit1->Text);

row = StringGrid1->RowCount;

col = StringGrid1->ColCount;

StringGrid1->Visible = true;

StringGrid1->Cells[0][0] = "Q";

StringGrid1->Cells[1][0] = "TC";

StringGrid1->Cells[2][0] = "VC";

StringGrid1->Cells[3][0] = "FC";

StringGrid1->Cells[4][0] = "AC";

StringGrid1->Cells[5][0] = "AVC";

StringGrid1->Cells[6][0] = "AFC";

StringGrid1->Cells[7][0] = "MC";

updateCells();

for (int i = 1; i < row; i++)

StringGrid1->Cells[0][i] = IntToStr(i);

StringGrid1->Cells[0][6] = IntToStr(Q);

for (int i = 0; i < col; i++)

StringGrid1->ColWidths[i] = StringGrid1->Cells[i][0].Length()\*20;

fillGrid();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::Button1Click(TObject \*Sender)

{

Q = StrToInt(Edit1->Text);

StringGrid1->Cells[0][6] = IntToStr(Q);

fillMatrix();

findFC();

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

{

int index = StrToInt(StringGrid1->Cells[0][i + 1]);

if (M[i][TC] != 0 || M[i][VC] != 0)

{

if(M[i][TC] != 0)

M[i][VC] = M[i][TC] - M[i][FC];

if(M[i][VC] != 0)

M[i][TC] = M[i][VC] + M[i][FC];

M[i][AC] = M[i][TC] / (index);

M[i][AVC] = M[i][VC] / (index);

}

if (M[i][AC] != 0 || M[i][AVC] != 0)

{

if (M[i][AC] != 0 && M[i][AVC] == 0)

M[i][AVC] = (M[i][AC] \* (index) - M[i][FC]) / (index);

if (M[i][AVC] != 0 && M[i][AC] == 0)

M[i][AC] = (M[i][AVC] \* (index) + M[i][FC]) / (index);

M[i][TC] = M[i][AC] \* (index);

M[i][VC] = M[i][AVC] \* (index);

}

if (M[i][MC] != 0)

{

if (M[i][TC] == 0)

M[i][TC] = M[i][MC] + M[i - 1][TC];

if (M[i][VC] == 0)

M[i][VC] = M[i][MC] + M[i - 1][VC];

M[i][AC] = M[i][TC] / (index);

M[i][AVC] = M[i][VC] / (index);

}

M[i][MC] = M[i][TC] - M[i - 1][TC];

M[i][AFC] = M[i][FC] / (index);

}

M[5][MC] = (M[5][TC] - M[4][TC]) / (Q - 5);

fillGrid();

}

//---------------------------------------------------------------------------

void TForm1::fillGrid()

{

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

{

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

{

StringGrid1->Cells[j+1][i+1] = FloatToStr(M[i][j]).SubString(0, 4);

}

}

}

//---------------------------------------------------------------------------

void TForm1::fillMatrix()

{

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

{

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

{

M[i][j] = StrToFloat(StringGrid1->Cells[j+1][i+1]);

}

}

}

//---------------------------------------------------------------------------

void TForm1::findFC()

{

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

{

if (M[i][AFC] != 0)

{

for (int j = 0; j < n; j++)

{

M[j][FC] = M[i][AFC] \* StrToInt(StringGrid1->Cells[0][i+1]);

}

}

}

}

//---------------------------------------------------------------------------

void TForm1::updateCells()

{

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

{

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

{

M[i][j] = 0;

}

}

M[0][TC] = 30;

M[1][VC] = 18;

M[2][AC] = 15;

M[3][AVC] = 7;

M[4][AFC] = 4;

M[5][AVC] = 3.5;

M[4][MC] = 2;

M[5][MC] = 1;

fillGrid();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::Button2Click(TObject \*Sender)

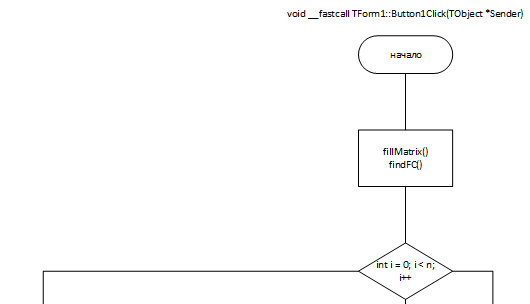
{

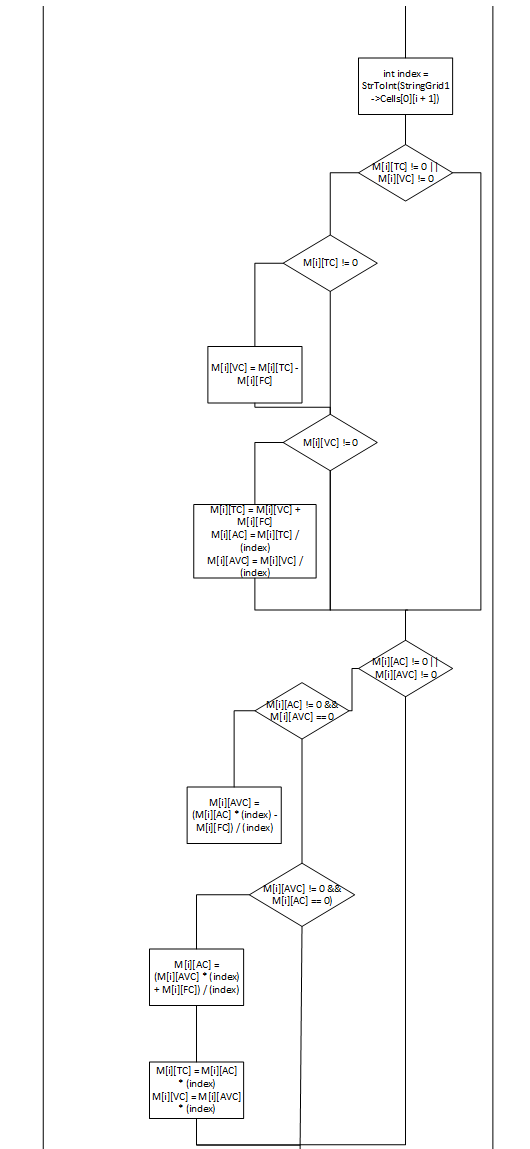
updateCells();

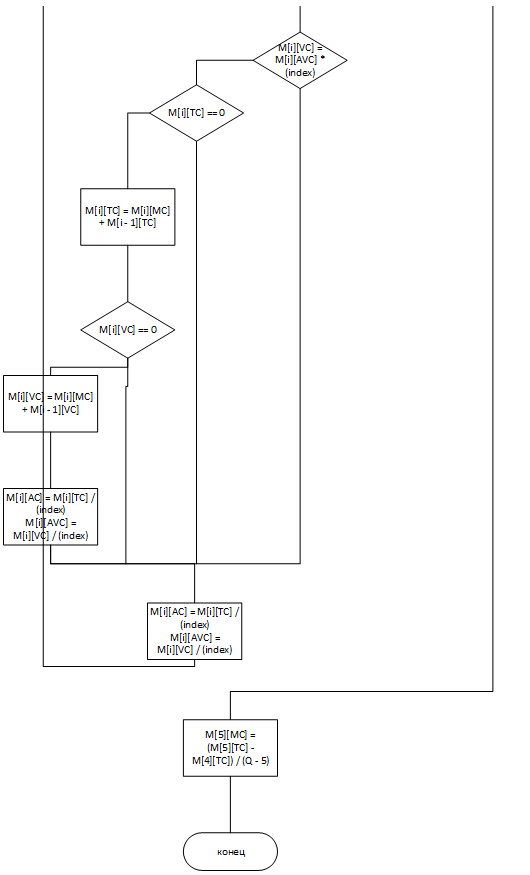
}

//---------------------------------------------------------------------------

**Блок – схема**

****

****

****

**Результат работы**

Программа работает без ошибок. В процессе работы происходит восстановление недостающей информации об издержках фирмы. Составлен алгоритм для восстановления утерянных данных из таблицы.

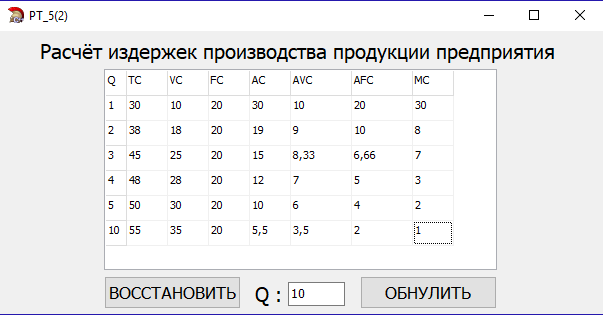


Рисунок 7– Результат работы программы

# **Список использованной литературы**

1. Е.В. Бережная, В.И. Бережной «Математические методы моделирования экономических систем». Москва: «Финансы и статистика», 2016 - 432 с.
2. «Финансы и кредит» под редакцией А.М. Ковалевой. Москва: «Финансы и статистика», 2005 - 512 с.
3. Айвор Хортон «Visual C++ 2010». Москва: «Вильямс» 2013 - 624с.
4. Н.А. Литвиненко «Технология программирования на С++. Win32 API-приложения». Санкт-Петербург: «БХВ-Петербург», 2010 - 0288с.
5. С. Прата «Язык программирования С++. Лекции и упражнения» . Москва: Издательский дом «Вильямс», 2012 – 1248 с.
6. Т.А.Павловская, «C/C++. Программирование на языке высокого уровня».– СПб.: Питер,2012.
7. И.Ш.Хабибуллин, «Программирование на языке высокого уровня. C/C++». – СПб.: БХВ-Петербург, 2013.
8. Г.Шилдт, «C++ базовый курс»: Пер. с англ. – М.: Вильямс, 2012.
9. Т.А.Павловская, Ю.А.Щупак, «C/C++. Структурное программирование: Практикум» – СПб.: Питер, 2013.