**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра информационных систем**

отчет

**по практической работе №3**

**по дисциплине «Программирование»**

|  |  |  |
| --- | --- | --- |
| Студент гр. 2373 |  | Чесноков М. А. |
| Преподаватель |  | Глущенко А. Г. |

Санкт-Петербург

2023

**Цель работы.**

Изучение структур в c++, массивов структур, создание прототипа баз данный в c++ при помощи структур, добавление функционала.

**Основные теоретические положения.**

В языке c++, структура — композитный тип данных, инкапсулирующий без сокрытия набор значений различных типов. Порядок размещения значений в памяти задаётся при определении типа и сохраняется на протяжении времени жизни объектов, что даёт возможность косвенного доступа. Пример объявления структуры: struct str\_name

Структуры можно использовать в качестве параметров функций, как и обычные переменные. Для структур поддерживаются все три механизма передачи данных: по значению, через указатели и по ссылке.

**Постановка задачи**

Необходимо создать программу, позволяющую:

1.   Создание новой записи о студенте.

2.   Внесение изменений в уже имеющуюся запись.

3.   Вывод всех данных о студентах.

4.   Вывод информации обо всех студентах группы *N*. *N* – инициализируется пользователем.

5.   Вывод топа самых успешных студентов с наивысшим по рейтингу средним баллом за прошедшую сессию.

6.   Вывод количества студентов мужского и женского пола.

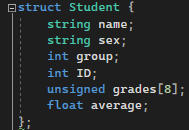
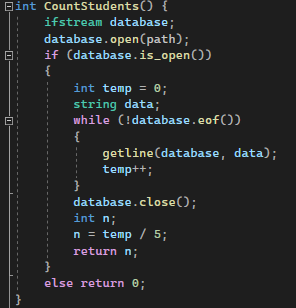
7.   Вывод данных о студентах, которые не получают стипендию; учатся только на «хорошо» и «отлично»; учатся только на «отлично»;

8.   Вывод данных о студентах, имеющих номер в списке – *k*.

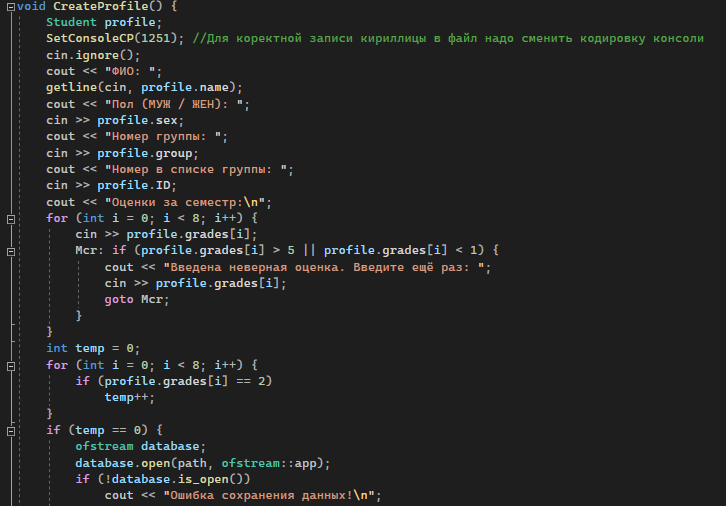
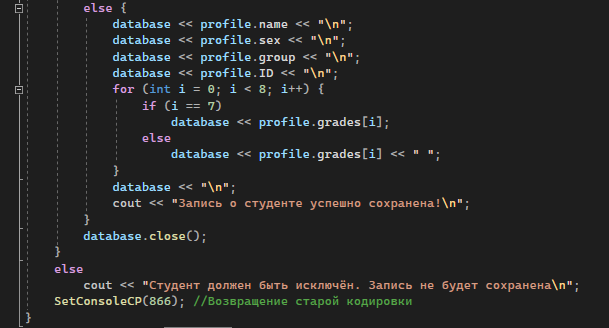
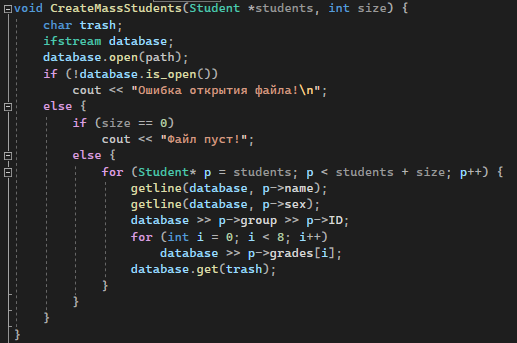
**Выполнение работы.**

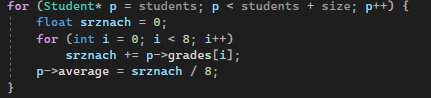
Код программы представлен в приложении А.

**Блок описания кода и использованных алгоритмов:**

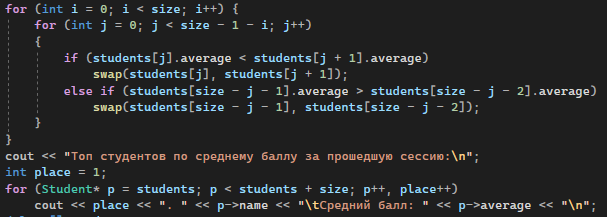
Создаём динамический массив структур, размер которого узнаём, считывая количество строк в изначальном файле Students.txt.

1. Создаётся элемент структуры, данные в который записывает пользователь, и сохраняется в файл.

Перед выполнением любого другого пункта программы создаётся динамический массив, который удаляется после выполнения пункта.

1. Пользователь вводит имя студента, данные которого хочет изменить, а затем меняет их.
2. Циклом for по очереди выводим всех студентов в списке.
3. Пользователь вводит номер группы и перебором выводятся студенты с соответствующим номером группы.
4. Сначала создаём функцию для высчитывания среднего значения оценок обучающегося.

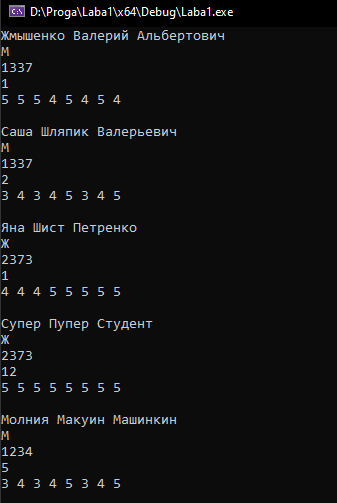
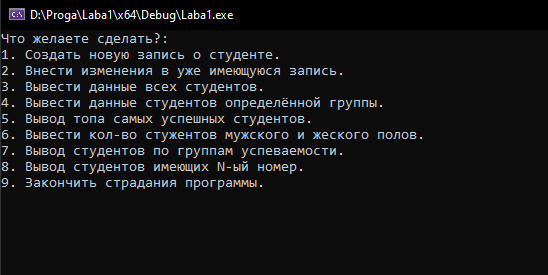
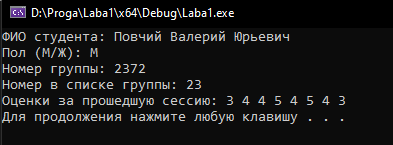
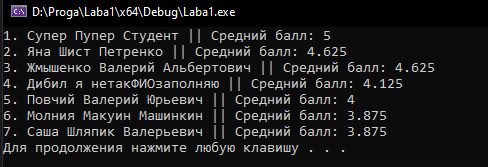
Далее сортируем массив алгоритмом Bubble sort.



1. Перебором проходимся по всему массиву и увеличиваем счётчики обеих полов.
2. Делаем проверку оценок студентов. Если находим соответствие, то выводим в консоль.
3. Пользователь вводит номер и перебором выводятся студенты с соответствующим номеров в группе.

**Выводы.**

Мы научились использовать структуры c++, написали прототип баз данный c++, научились использовать функционал структур.

**Блок скриншотов работы программы**

Приложение А

рабочий код

#include <iostream>

#include <string>

#include <stack>

using namespace std;

int Prior(char sign) {

switch (sign) {

case '\*': case '/': return 2;

case '+': case '-': return 1;

case '(': case ')': return 0;

}

return -1;

}

int Initialization(char letter) {

int num;

cout << letter << " = ";

cin >> num;

return num;

}

float Math(stack <float>& stackOut, char sign, float s1, float s2) {

if (sign == '/' && s2 == 0) {

cout << endl;

throw '0';

}

switch (sign) {

case '+':

cout << (s1 + s2) << endl;

return s1 + s2;

case '-':

cout << (s1 - s2) << endl;

return s1 - s2;

case '\*':

cout << (s1 \* s2) << endl;

return s1 \* s2;

case '/':

cout << (s1 / s2) << endl;

return s1 / s2;

}

}

//Проверки

void Check(string& in) {

int cIn = 0, cOp = 0, cCl = 0, cNum = 0, cSig = 0;

char sign = '.';

while (in[cIn] != '\0') {

if (in[cIn] == '(') {

cOp++;

if (in[cIn + 1] == ')') throw exception();

cIn++;

}

else if (in[cIn] == ')') {

cCl++;

cIn++;

}

else if (in[cIn] == '+' || in[cIn] == '-' || in[cIn] == '\*' || in[cIn] == '/') {

if (in[cIn + 1] == ' ') {

sign = in[cIn];

while (in[cIn + 1] == ' ') cIn++;

}

if (in[cIn + 1] == '+' || in[cIn + 1] == '-' || in[cIn + 1] == '\*' || in[cIn + 1] == '/') throw exception();

if ((in[cIn] == '/' || sign == '/') && in[cIn + 1] == '0') throw '0';

cIn++;

cSig++;

}

else if (in[cIn] >= 'a' && in[cIn] <= 'z') {

if (in[cIn + 1] == ' ') while (in[cIn + 1] == ' ') cIn++;

if ((in[cIn+1] >= 'a' && in[cIn+1] <= 'z') || (in[cIn] >= '0' && in[cIn] <= '9')) throw exception();

cIn++;

cNum++;

}

else if (in[cIn] >= '0' && in[cIn] <= '9') {

cIn++;

while (in[cIn] >= '0' && in[cIn] <= '9') {

cIn++;

}

cNum++;

if (in[cIn] == ' ') while (in[cIn] == ' ') cIn++;

if (in[cIn] >= '0' && in[cIn] <= '9') throw exception();

}

else cIn++;

}

if (cOp != cCl) throw exception();

if (cNum - cSig != 1) throw exception();

}

void CheckPol(string& in) {

int cIn = 0, cNum = 0, cSig = 0;

for (; in[cIn] == ' '; cIn++);

if (in[cIn] == '+' || in[cIn] == '-' || in[cIn] == '\*' || in[cIn] == '/') cIn = 0;

else throw exception();

for (; in[cIn] != '\0'; cIn++);

for (; in[cIn] == ' '; cIn--);

cIn--;

if (in[cIn] == '+' || in[cIn] == '-' || in[cIn] == '\*' || in[cIn] == '/') throw exception();

else cIn = 0;

while (in[cIn] != '\0') {

if (in[cIn] == '+' || in[cIn] == '-' || in[cIn] == '\*' || in[cIn] == '/') {

cSig++;

if (in[cIn] == '/' && in[cIn+4] == '0') throw '0';

cIn++;

}

else if (in[cIn] >= 'a' && in[cIn] <= 'z') cNum++;

else if (in[cIn] == '(' || in[cIn] == ')') throw exception();

else if (in[cIn] >= '0' && in[cIn] <= '9') {

cIn++;

while (in[cIn] >= '0' && in[cIn] <= '9') {

cIn++;

}

cNum++;

}

else cIn++;

}

if (cNum - cSig != 1) throw exception();

}

void CheckRevPol(string& in) {

int cIn = 0, cNum = 0, cSig = 0;

for (; in[cIn] == ' '; cIn++);

if (in[cIn] == '+' || in[cIn] == '-' || in[cIn] == '\*' || in[cIn] == '/') throw exception();

else cIn = 0;

for (; in[cIn] != '\0'; cIn++);

for (; in[cIn] == ' '; cIn--);

cIn--;

if (in[cIn] == '+' || in[cIn] == '-' || in[cIn] == '\*' || in[cIn] == '/') cIn = 0;

else throw exception();

while (in[cIn] != '\0') {

if (in[cIn] == '+' || in[cIn] == '-' || in[cIn] == '\*' || in[cIn] == '/') {

cSig++;

if (in[cIn] == '/' && in[cIn - 2] == '0') throw '0';

cIn++;

}

else if (in[cIn] >= 'a' && in[cIn] <= 'z') cNum++;

else if (in[cIn] == '(' || in[cIn] == ')') throw exception();

else if (in[cIn] >= '0' && in[cIn] <= '9') {

cIn++;

while (in[cIn] >= '0' && in[cIn] <= '9') {

cIn++;

}

cNum++;

}

else cIn++;

}

if (cNum - cSig != 1) throw exception();

}

//Обратная польская запись

void RevPush(stack<char>& stackSign, char sign, string& out, string& stline) {

int prior = Prior(sign);

if (sign == '(') {

stackSign.push(sign);

stline += sign;

cout << "Вывод: " << out << "\tСтек: ";

for (int i = 0; stline[i] != '\0'; i++) cout << stline[i] << " ";

cout << endl;

return;

}

if (sign == ')') {

while (stackSign.top() != '(') {

out += stackSign.top();

out += ' ';

stackSign.pop();

stline.erase((stline.end() - 1));

cout << "Вывод: " << out << "\tСтек: ";

for (int i = 0; stline[i] != '\0'; i++) cout << stline[i] << " ";

cout << endl;

}

stackSign.pop();

stline.erase((stline.end() - 1));

cout << "Вывод: " << out << "\tСтек: ";

for (int i = 0; stline[i] != '\0'; i++) cout << stline[i] << " ";

cout << endl;

return;

}

if (!stackSign.empty() && prior <= Prior(stackSign.top())) {

while (!stackSign.empty() && prior <= Prior(stackSign.top())) {

out += stackSign.top();

out += ' ';

stackSign.pop();

stline.erase((stline.end() - 1));

cout << "Вывод: " << out << "\tСтек: ";

for (int i = 0; stline[i] != '\0'; i++) cout << stline[i] << " ";

cout << endl;

}

}

stackSign.push(sign);

stline += sign;

cout << "Вывод: " << out << "\tСтек: ";

for (int i = 0; stline[i] != '\0'; i++) cout << stline[i] << " ";

cout << endl;

return;

}

void RevOut(stack<char>& stackSign, string& out, string& stline) {

while (!stackSign.empty()) {

out += stackSign.top();

out += ' ';

stackSign.pop();

stline.erase((stline.end() - 1));

cout << "Вывод: " << out << "\tСтек: ";

for (int i = 0; stline[i] != '\0'; i++) cout << stline[i] << " ";

cout << endl;

}

cout << "Обратная польская запись: " << out << endl;

}

void ReversePolishNotation(stack<char>& stackSign, string& in, string& out) {

int cIn = 0;

string stline;

while (in[cIn] != '\0') {

if (in[cIn] >= 'a' && in[cIn] <= 'z') {

out += in[cIn++];

out += ' ';

cout << "Вывод: " << out << "\tСтек: ";

for (int i = 0; stline[i] != '\0'; i++) cout << stline[i] << " ";

cout << endl;

}

else if (in[cIn] >= '0' && in[cIn] <= '9') {

out += in[cIn++];

while (in[cIn] >= '0' && in[cIn] <= '9') {

out += in[cIn++];

}

out += ' ';

cout << "Вывод: " << out << "\tСтек: ";

for (int i = 0; stline[i] != '\0'; i++) cout << stline[i] << " ";

cout << endl;

}

else if ((in[cIn] >= '(' && in[cIn] <= '+') || in[cIn] == '-' || in[cIn] == '/')

{

RevPush(stackSign, in[cIn++], out, stline);

}

else cIn++;

}

RevOut(stackSign, out, stline);

}

void RevPolCount(string& out) {

stack <float> stackOut;

string num;

float s1, s2;

int cOut = 0, x;

while (out[cOut] != '\0') {

if (out[cOut] == ' ') cOut++;

else if (out[cOut] >= '0' && out[cOut] <= '9') {

num += out[cOut++];

while (out[cOut] >= '0' && out[cOut] <= '9') {

num += out[cOut++];

}

x = stoi(num);

stackOut.push(x);

num.clear();

}

else if (out[cOut] >= 'a' && out[cOut] <= 'z')

stackOut.push(Initialization(out[cOut++]));

else if (out[cOut] == '+' || out[cOut] == '-' || out[cOut] == '\*' || out[cOut] == '/') {

s2 = stackOut.top();

stackOut.pop();

s1 = stackOut.top();

stackOut.pop();

cout << s1 << out[cOut] << s2 << " = ";

stackOut.push(Math(stackOut, out[cOut++], s1, s2));

}

}

cout << "Ответ: " << stackOut.top() << endl;

}

//Польская запись

void PolPush(stack<char>& stackSign, char sign, string& out, string& stline) {

int prior = Prior(sign);

if (sign == ')') {

stackSign.push(sign);

stline += sign;

cout << "Вывод: " << out << "\tСтек: ";

for (int i = 0; stline[i] != '\0'; i++) cout << stline[i] << " ";

cout << endl;

return;

}

if (sign == '(') {

while (stackSign.top() != ')') {

out += stackSign.top();

out += ' ';

stackSign.pop();

stline.erase((stline.end() - 1));

cout << "Вывод: " << out << "\tСтек: ";

for (int i = 0; stline[i] != '\0'; i++) cout << stline[i] << " ";

cout << endl;

}

stackSign.pop();

stline.erase((stline.end() - 1));

cout << "Вывод: " << out << "\tСтек: ";

for (int i = 0; stline[i] != '\0'; i++) cout << stline[i] << " ";

cout << endl;

return;

}

if (!stackSign.empty() && prior < Prior(stackSign.top())) {

while (!stackSign.empty() && prior < Prior(stackSign.top())) {

out += stackSign.top();

out += ' ';

stackSign.pop();

stline.erase((stline.end() - 1));

cout << "Вывод: " << out << "\tСтек: ";

for (int i = 0; stline[i] != '\0'; i++) cout << stline[i] << " ";

cout << endl;

}

}

stackSign.push(sign);

stline += sign;

cout << "Вывод: " << out << "\tСтек: ";

for (int i = 0; stline[i] != '\0'; i++) cout << stline[i] << " ";

cout << endl;

return;

}

void PolOut(stack<char>& stackSign, string& out, string& stline) {

while (!stackSign.empty()) {

out += stackSign.top();

out += ' ';

stackSign.pop();

stline.erase((stline.end() - 1));

cout << "Вывод: " << out << "\tСтек: ";

for (int i = 0; stline[i] != '\0'; i++) cout << stline[i] << " ";

cout << endl;

}

reverse(out.begin(), out.end());

cout << "Перевернём получившийся вывод: " << out << endl;

cout << "Польская запись: " << out << endl;

}

void PolishNotation(stack<char>& stackSign, string& in, string& out) {

int cIn = 0;

string stline;

reverse(in.begin(), in.end());

cout << "Перевернём введённое выражение: " << in << endl;

while (in[cIn] != '\0') {

if (in[cIn] >= 'a' && in[cIn] <= 'z') {

out += in[cIn++];

out += ' ';

cout << "Вывод: " << out << "\tСтек: ";

for (int i = 0; stline[i] != '\0'; i++) cout << stline[i] << " ";

cout << endl;

}

else if (in[cIn] >= '0' && in[cIn] <= '9') {

out += in[cIn++];

while (in[cIn] >= '0' && in[cIn] <= '9') {

out += in[cIn++];

}

out += ' ';

cout << "Вывод: " << out << "\tСтек: ";

for (int i = 0; stline[i] != '\0'; i++) cout << stline[i] << " ";

cout << endl;

}

else if (in[cIn] == '(' || in[cIn] == ')' || in[cIn] == '+' || in[cIn] == '-' || in[cIn] == '\*' || in[cIn] == '/')

{

PolPush(stackSign, in[cIn++], out, stline);

}

else cIn++;

}

PolOut(stackSign, out, stline);

}

void PolCount(string& out) {

stack <float> stackOut;

string num;

float s1, s2;

int cOut = 0, x;

reverse(out.begin(), out.end());

while (out[cOut] != '\0') {

if (out[cOut] >= '0' && out[cOut] <= '9') {

while (out[cOut] != ' ')

num += out[cOut++];

reverse(num.begin(), num.end());

x = stoi(num);

stackOut.push(x);

num.clear();

}

else if (out[cOut] >= 'a' && out[cOut] <= 'z')

stackOut.push(Initialization(out[cOut++]));

else if (out[cOut] == '+' || out[cOut] == '-' || out[cOut] == '\*' || out[cOut] == '/') {

s1 = stackOut.top();

stackOut.pop();

s2 = stackOut.top();

stackOut.pop();

cout << s1 << out[cOut] << s2 << " = ";

stackOut.push(Math(stackOut, out[cOut++], s1, s2));

}

else if (out[cOut] == ' ') cOut++;

}

reverse(out.begin(), out.end());

cout << "Ответ: " << stackOut.top() << endl;

}

void main() {

setlocale(LC\_ALL, "rus");

stack <char> stackSign;

string in, out;

char choice1, choice2;

while (true) {

cout << "\t\tМеню\n"

<< "1 - Перевести выражение в прямую или обратную польскую запись.\n"

<< "2 - Посчитать выражение в любой записи.\n"

<< "0 - Выход.\n"

<< "Ваш выбор: ";

cin >> choice1;

switch (choice1) {

case '1':

system("cls");

cout << "1 - Перевести выражение в прямую польскую запись.\n"

<< "2 - Перевести выражение в обратную польскую запись.\n"

<< "0 - Вернуться в меню.\n"

<< "Ваш выбор: ";

cin >> choice2;

switch (choice2) {

case '1':

system("cls");

cout << "Введите выражение: ";

cin.ignore();

getline(cin, in);

try {

Check(in);

PolishNotation(stackSign, in, out);

in.clear();

out.clear();

}

catch (exception& exp) {

cout << "Допущена ошибка в записи выражения. Вы будете возвращены в меню.\n";

}

catch (char& exp) {

cout << "Делить на 0 нельзя. Вы будете возвращены в меню.\n";

}

system("pause");

system("cls");

break;

case '2':

system("cls");

cout << "Введите выражение: ";

cin.ignore();

getline(cin, in);

try {

Check(in);

ReversePolishNotation(stackSign, in, out);

in.clear();

out.clear();

}

catch (exception& exp) {

cout << "Допущена ошибка в записи выражения. Вы будете возвращены в меню.\n";

}

catch (char& exp) {

cout << "Делить на 0 нельзя. Вы будете возвращены в меню.\n";

}

system("pause");

system("cls");

break;

case '0':

cout << "Возвращаю в меню.\n";

system("pause");

system("cls");

break;

default:

cout << "Пункт выбран неправильно. Вы будете возвращены в меню.\n";

system("pause");

system("cls");

break;

}

break;

case '2':

system("cls");

cout << "Как вы хотите записать выражение?\n"

<< "1 - Обычная запись.\n"

<< "2 - Прямая польская запись\n"

<< "3 - Обратная польская запись\n"

<< "0 - Вернуться в меню.\n"

<< "Ваш выбор: ";

cin >> choice2;

switch (choice2) {

case '1':

system("cls");

cout << "Введите выражение: ";

cin.ignore();

getline(cin, in);

try {

Check(in);

cout << "Переведём выражение в обратную польскую запись.\n";

ReversePolishNotation(stackSign, in, out);

cout << endl;

cout << "Посчитаем выражение в обратной польской записи.\n";

RevPolCount(out);

in.clear();

out.clear();

}

catch (exception& exp) {

cout << "Допущена ошибка в записи выражения. Вы будете возвращены в меню.\n";

}

catch (char& exp) {

cout << "Делить на 0 нельзя. Вы будете возвращены в меню.\n";

}

system("pause");

system("cls");

break;

case '2':

system("cls");

cout << "Введите выражение: ";

cin.ignore();

getline(cin, in);

try {

CheckPol(in);

cout << "Посчитаем выражение в прямой польской записи.\n";

PolCount(in);

in.clear();

}

catch (exception& exp) {

cout << "Допущена ошибка в записи выражения. Вы будете возвращены в меню.\n";

}

catch (char& exp) {

cout << "Делить на 0 нельзя. Вы будете возвращены в меню.\n";

}

system("pause");

system("cls");

break;

case '3':

system("cls");

cout << "Введите выражение: ";

cin.ignore();

getline(cin, in);

try {

CheckRevPol(in);

cout << "Посчитаем выражение в обратной польской записи.\n";

RevPolCount(in);

in.clear();

}

catch (exception& exp) {

cout << "Допущена ошибка в записи выражения. Вы будете возвращены в меню.\n";

}

catch (char& exp) {

cout << "Делить на 0 нельзя. Вы будете возвращены в меню.\n";

}

system("pause");

system("cls");

break;

case '0':

cout << "Возвращаю в меню.\n";

system("pause");

system("cls");

break;

default:

cout << "Пункт выбран неправильно. Вы будете возвращены в меню.\n";

system("pause");

system("cls");

break;

}

break;

case '0':

cout << "\nПрограмма завершила свою работу.\n";

return;

default:

cout << "Пункт выбран неправильно!\n";

system("pause");

system("cls");

break;

}

}

}