ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ



МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

ВЫСШАЯ ШКОЛА ПЕЧАТИ И МЕДИАИНДУСТРИИ

*Институт Принтмедиа и информационных технологий*

*Кафедра Информатики и информационных технологий*

направление подготовки

09.03.02 «Информационные системы и технологии»,

ЛАБОРАТОРНАЯ РАБОТА № 5

Дисциплина: ОБЪЕКТНО-ОРИЕНТИРОВАННОЕ ПРОГРАММИРОВАНИЕ

Тема: Объекты и классы

Выполнил(а): студент(ка) группы 181-722

Голиков Р.О.

(Фамилия И.О.)

Дата, подпись \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ *\_\_\_\_\_\_\_\_\_\_\_*

(Дата) (Подпись)

Проверил: Ктн Доцент Арсентьев Д.А.*\_\_\_\_\_\_\_\_\_\_*

(Фамилия И.О. степень, звание) (Оценка)

Дата, подпись \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ *\_\_\_\_\_\_\_\_\_\_\_*

(Дата) (Подпись)

Замечания: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Москва

2019

**Задание 1**

Создайте класс Int, имитирующий стандартный тип int. Единственное поле этого класса должно иметь тип int. Создайте методы, которые будут устанавливать значение поля, равным нулю, инициализировать его целым значением, выводить значение поля на экран и складывать два значения типа Int.

Напишите программу, в которой будут созданы три объекта класса Int, два из которых будут инициализированы. Сложите два инициализированных объекта, присвойте результат третьему, а затем отобразите результат на экране.

**Код:**

#include <iostream>

using namespace std;

class Int //(не то же самое, что int)

{

private:

int i;

public:

Int() //создание Int

{

i = 0;

}

Int(int ii) //создание и инициализаци Int

{

i = ii;

}

void add(Int i2, Int i3) //складывает два значени

//типа Int

{

i = i2.i + i3.i;

}

void display() //вывести Int

{

cout << i;

}

};

int main()

{

Int Int1(7); //создать и инициализировать Int

Int Int2(11); //создать и инициализировать Int

Int Int3; //создать Int

Int3.add(Int1, Int2); //сложение двух переменных типа Int

cout << "\nInt3 ="; Int3.display(); //вывести результат

cout << endl;

return 0;

}

**Вывод:**  


**Задание 2**

Представьте пункт для взимания платежей за проезд по автостраде. Каждая проезжающая машина должна заплатить за проезд 50 центов, однако часть машин платит за проезд, а часть проезжает бесплатно. В кассе ведется учет числа проехавших машин и суммарная выручка от платы за проезд.

Создайте модель такой кассы с помощью класса toLLBooth. Класс должен содержать два поля. Одно из них, типа unsigned int, предназначено для учета количества проехавших автомобилей, а второе, имеющее тип double, будет содержать суммарную выручку от оплаты проезда. Конструктор должен инициализировать оба поля нулевыми значениями. Метод payingCar() инкрементирует число машин и увеличивает на 0,50 суммарную выручку. Другой метод, порауСаг(), увеличивает на единицу число автомобилей, но оставляет без изменения выручку. Метод display() выводит оба значения на экран. Там, где это возможно, сделайте методы константными.

Создайте программу, которая продемонстрирует работу класса. Программа должна предложить пользователю нажать одну клавишу для того, чтобы сымитировать заплатившего автолюбителя, и другую клавишу, чтобы сымитировать недобросовестного водителя. Нажатие клавиши Esc должно привести к выдаче текущих значений количества машин и выручки и завершению программы.

**Код:**

#include <iostream>

#include <conio.h>

using namespace std;

const char ESC = 27;//Код клавиши ESC

const double TOLL = 0.5;//пошлина равна 50 центами

class tollBooth

{

private:

unsigned int totalCars; //всего машин за день

double totalCash; //всего денег за день

public: //конструктор

tollBooth() : totalCars(0), totalCash(0.0)

{}

void payingCar() //а car paid

{

totalCars++; totalCash += TOLL;

}

void nopayCar() //а car didn 't pay

{

totalCars++;

}

void display()const //display totals

{

cout << "\n\_Машины: " << totalCars << endl

<< "Деньги: " << totalCash

<< endl;

}

};

int main()

{

setlocale(LC\_ALL, "");

tollBooth booth1; //создает турникет

char ch;

cout << "\nНажмите 0 для машины без оплаты,"

<< "\n1 для каждой оплачивающей машины,"

<< "\nEsc для выхода.\n";

do {

ch = \_getche(); //получить символ

if (ch == '0 ') //если это 0, машина не оплачивала

booth1.nopayCar();

if (ch == '1') //если 1, машина оплачивала

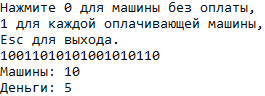
booth1.payingCar();

} while (ch != ESC); //выход из цикла по Esc

booth1.display(); //вывод отчета

return 0;

}

**Вывод:**  


**Задание 3**

Создайте класс с именем time, содержащий три поля типа int, предназначенные для хранения часов, минут и секунд. Один из конструкторов класса должен инициализировать поля нулевыми значениями, а другой конструктор - заданным набором значений. Создайте метод класса, который будет выводить значения полей на экран в формате 11:59:59, и метод, складывающий значения двух объектов типа time, передаваемых в качестве аргументов.

В функции main() следует создать два инициализированных объекта (подумайте, должны ли они быть константными) и один неинициализированный объект. Затем сложите два инициализированных значения, а результат присвойте третьему объекту и выведите его значение на экран. Где возможно, сделайте методы константными

**Код:**

#include "pch.h"

#include <iostream>

using namespace std;

class time

{

private:

int hrs, mins, secs;

public:

time() :hrs(0), mins(0), secs(0) //конструктор без аргументов

{}

//конструктор с тремя аргументами

time(int h, int m, int s) :hrs(h), mins(m), secs(s)

{}

void display()const //формат 11:59:59

{

cout << hrs << ":" << mins << ":" << secs;

}

void add\_time(time t1, time t2)//сложить две переменные

{

secs = t1.secs + t2.secs; //сложить секунды

if (secs > 59) //если перебор,

{

secs -= 60; mins++;

} //прибавить минуту

mins += t1.mins + t2.mins; //сложить минуты

if (mins > 59) //если слишком много минут,

{

mins -= 60; hrs++;

} //прибавить час

hrs += t1.hrs + t2.hrs; //сложить часы

}

};

int main()

{

const time time1(5, 59, 59); //создание и инициализаци€

const time time2(4, 30, 30); //двух переменных

time time3;//создать еще одну переменную

time3.add\_time(time1, time2); //сложить две переменные

cout << "time3 = "; time3.display();//вывести результат

cout << endl;

return 0;

}

**Вывод:**  


**Задание 4**

Создайте класс employee, используя задачу 4 лабораторной работы 2. Класс должен включать поле типа int для хранения номера сотрудника и поле типа float для хранения величины его оклада. Методы класса должны позволять пользователю вводить и отображать данные класса. Напишите функцию main(), которая запросит пользователя ввести данные для трех сотрудников и выведет полученную информацию на экран.

**Код:**

#include <iostream>

using namespace std;

class employee

{

private:

int \_id;

float \_salary;

public:

employee(int id, float salary) : \_id(id), \_salary(salary)

{

}

employee()

{

}

void display()

{

cout << "Сотрудник " << \_id << ", зарплата - $" << \_salary << ".\n";

}

};

int main()

{

setlocale(LC\_ALL, "");

employee e1, e2, e3;

int id;

float sal;

cout << "Введите данные трёх сотрудников:" << endl;

cin >> id >> sal;

e1 = employee(id, sal);

cin >> id >> sal;

e2 = employee(id, sal);

cin >> id >> sal;

e3 = employee(id, sal);

cout << "Введённые сотрудники: " << endl;

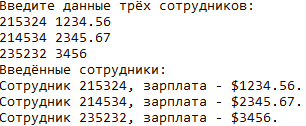
e1.display();

e2.display();

e3.display();

return 0;

}

**Вывод:**  


**Задание 5**

Взяв в качестве основы структуру из задачи 5 лабораторной работы 2, создайте класс date. Его данные должны размещаться в трех полях типа int: month, day и year. Метод класса getdate() должен принимать значение для объекта в формате 12/31/02, а метод showdate() - выводить данные на экран.

**Код:**

#include <iostream>

#include <string>

using namespace std;

class date

{

private:

int \_day;

int \_month;

int \_year;

public:

void getdate(string datestr)

{

\_day = (datestr[0] - '0') \* 10 + (datestr[1] - '0');

\_month = (datestr[3] - '0') \* 10 + (datestr[4] - '0');

\_year = (datestr[6] - '0') \* 10 + (datestr[7] - '0');

}

void display()

{

cout << (\_day < 10 ? '0' : '\0') << \_day << '/' <<

(\_month < 10 ? '0' : '\0') << \_month << '/' <<

(\_year < 10 ? '0' : '\0') << \_year << endl;

}

};

int main()

{

setlocale(LC\_ALL, "");

string str;

cout << "Введите дату:\n";

cin >> str;

date d;

d.getdate(str);

cout << "Введённая дата:";

d.display();

return 0;

}

**Вывод:**  


**Задание 6**

Расширьте содержание класса employee из задачи 4, включив в него класс date и перечисление etype (см. задачу 6 лабораторной работы 2). Объект класса date будет использоваться для хранения даты приема сотрудника на работу. Перечисление будет использовано для хранения статуса сотрудника: лаборант, секретарь, менеджер и т. д. Последние два поля данных должны быть закрытыми в определении класса employee, как и номер и оклад сотрудника. Вам будет необходимо разработать методы getemploy() и putemploy(), предназначенные соответственно для ввода и отображения информации о сотруднике. Возможно, при создании методов вам понадобится ветвление switch для работы с перечисляемым типом etype. Напишите функцию main(), которая попросит пользователя ввести данные о трех сотрудниках, а затем выведет эти данные на экран.

**Код:**

#include <iostream>

#include <string>

using namespace std;

enum etype

{

laborer,

secretary,

manager,

accountant,

executive,

researcher

};

class date

{

private:

int \_day;

int \_month;

int \_year;

public:

void getdate(string datestr)

{

\_day = (datestr[0] - '0') \* 10 + (datestr[1] - '0');

\_month = (datestr[3] - '0') \* 10 + (datestr[4] - '0');

\_year = (datestr[6] - '0') \* 10 + (datestr[7] - '0');

}

void display()

{

cout << (\_day < 10 ? '0' : '\0') << \_day << '/' <<

(\_month < 10 ? '0' : '\0') << \_month << '/' <<

(\_year < 10 ? '0' : '\0') << \_year;

}

};

class employee

{

private:

int \_id;

float \_salary;

date \_date;

etype \_etype;

public:

void putemployee(int id, float salary, date d, char et)

{

\_id = id;

\_salary = salary;

\_date = d;

switch (et)

{

case 'l': \_etype = laborer; break;

case 's': \_etype = secretary; break;

case 'm': \_etype = manager; break;

case 'a': \_etype = accountant; break;

case 'e': \_etype = executive; break;

case 'r': \_etype = researcher; break;

}

}

void display()

{

cout << "Сотрудник " << \_id << ", зарплата - $" << \_salary << ", должность - ";

switch (\_etype)

{

case 0: cout << "laborer"; break;

case 1: cout << "secretary"; break;

case 2: cout << "manager"; break;

case 3: cout << "accountant"; break;

case 4: cout << "executive"; break;

case 5: cout << "researcher"; break;

}

cout << ", дата - ";

\_date.display();

cout << endl;

}

};

int main()

{

setlocale(LC\_ALL, "");

employee e1, e2, e3;

date d;

string str;

char a;

int id;

float sal;

cout << "Введите даные 3 сотрудников\n" <<

"(номер, величину зарплаты, должность(первую букву) и дату принятия на работу)\n";

cin >> id >> sal >> a >> str;

d.getdate(str);

e1.putemployee(id, sal, d, a);

cin >> id >> sal >> a >> str;

d.getdate(str);

e2.putemployee(id, sal, d, a);

cin >> id >> sal >> a >> str;

d.getdate(str);

e3.putemployee(id, sal, d, a);

cout << "Введённые сотрудники: " << endl;

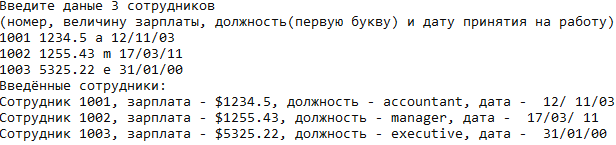
e1.display();

e2.display();

e3.display();

return 0;

}

**Вывод:**  


**Задание 7**

В морской навигации координаты точки измеряются в градусах и минутах широты и долготы. Например, координаты бухты Панити на о. Таити равны 149 градусов 34.8 минут восточной долготы и 17 градусов 31.5 минут южной широты. Это записывается как 149°34.8' W, 17°31.5' S. Один градус равен 60 минутам (устаревшая система также делила одну минуту на 60 секунд, но сейчас минуту делят на обычные десятичные доли). Долгота измеряется величиной от 0 до 180 градусов восточнее или западнее Гринвича. Широта принимает значения от 0 до 90 градусов севернее или южнее экватора.

Создайте класс angle, включающий следующие три поля: типа int для числа градусов, типа float для числа минут и типа char для указания направления (N, S, Е или W). Объект этого класса может содержать значение как широты, так и долготы. Создайте метод, позволяющий ввести координату точки, направление, в котором она измеряется, и метод, выводящий на экран значение этой координаты, например 179°59.9' Е. Кроме того, напишите конструктор, принимающий три аргумента. Напишите функцию main(), которая сначала создает переменную с помощью трехаргументного конструктора и выводит ее значение на экран, а затем циклически запрашивает пользователя ввести значение координаты и отображает введенное значение на экране. Для вывода символа градусов (°) можно воспользоваться символьной константой '\xF8'.

**Код:**

#include <iostream>

#include <conio.h>

using namespace std;

class angle

{

private:

int \_degrees;

float \_minutes;

char \_direction;

public:

angle(int degrees, float minutes, char direction) :

\_degrees(degrees), \_minutes(minutes), \_direction(direction)

{

}

void input()

{

char ch, direction;

int degrees;

float minutes;

cin >> degrees >> ch >> minutes >> ch >> direction;

\_degrees = degrees;

\_minutes = minutes;

\_direction = direction;

}

void display()

{

cout << \_degrees << '°' << \_minutes << "' " << \_direction << endl;

}

};

int main()

{

setlocale(LC\_ALL, "");

angle a1 = angle(0, 0, 'W');

angle a2 = angle(0, 0, 'S');

char ch;

do

{

cout << "\nВведите координаты точки:\n";

a1.input();

cin >> ch;

a2.input();

cout << "Вы ввели:" << endl;

a1.display();

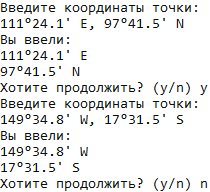
a2.display();

cout << "Хотите продолжить? (y/n) ";

} while (ch = \_getche() == 'y');

return 0;

}

**Вывод:**  


**Задание 8**

Создайте класс, одно из полей которого хранит «порядковый номер» объекта, то есть для первого созданного объекта значение этого поля равно 1, для второго созданного объекта значение равно 2 и т. д.

Для того чтобы создать такое поле, вам необходимо иметь еще одно поле, в которое будет записываться количество созданных объектов класса (это означает, что последнее поле должно относиться не к отдельным объектам класса, а ко всему классу в целом. Вспомните, какое ключевое слово необходимо при описании такого поля.). Каждый раз при создании нового объекта конструктор может получить значение этого поля и в соответствии с ним назначить объекту индивидуальный порядковый номер.

В класс следует включить метод, который будет выводить на экран порядковый номер объекта. Создайте функцию main(), в которой будут созданы три объекта, и каждый объект выведет на экран свой порядковый номер, например: Мой порядковый номер: 2 и т. п.

**Код:**

#include <iostream>

using namespace std;

class object

{

private:

static int \_amount;

int \_id;

public:

object()

{

\_id = ++\_amount;

}

void display()

{

cout << "Мой порядковый номер: " << \_id << endl;

}

};

int object::\_amount = 0;

int main()

{

setlocale(LC\_ALL, "");

object o1, o2, o3;

o1.display();

o2.display();

o3.display();

return 0;

}

**Вывод:**  


**Задание 9**

На основе структуры fraction из упражнения 8 главы 4 создайте класс fraction. Данные класса должны быть представлены двумя полями: числителем и знаменателем. Методы класса должны получать от пользователя значения числителя и знаменателя дроби в форме 3/5 и выводить значение дроби в этом же формате. Кроме того, должен быть разработан метод, складывающий значения двух дробей. Напишите функцию main(), которая циклически запрашивает у пользователя ввод пары дробей, затем складывает их и выводит результат на экран. После каждой такой операции программа должна спрашивать пользователя, следует ли продолжать цикл.

**Код:**

#include <iostream>

#include <string>

#include <cmath>

using namespace std;

class fraction

{

private:

int \_dividend;

int \_divider;

public:

void parse(string str)

{

\_dividend = 0;

\_divider = 0;

int i, j;

int len = str.length();

for (i = 0; i < len && str[i] != '/'; i++);

for (j = i - 1; j >= 0; j--)

\_dividend += (str[j] - '0') \* pow(10, i - j - 1);

for (j = len - 1; j > i; j--)

\_divider += (str[j] - '0') \* pow(10, len - j - 1);

}

void add(fraction f)

{

\_dividend = \_dividend \* f.\_divider + \_divider \* f.\_dividend;

\_divider = \_divider \* f.\_divider;

}

void display()

{

cout << \_dividend << '/' << \_divider << endl;

}

};

int main()

{

setlocale(LC\_ALL, "");

fraction f1, f2;

char ch;

string str;

do {

cout << "Введите две дроби:" << endl;

cin >> str;

f1.parse(str);

cin >> str;

f2.parse(str);

f1.add(f2);

cout << "Сумма: ";

f1.display();

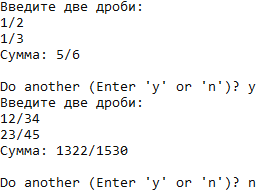
cout << "\nDo another (Enter 'y' or 'n')? ";

cin >> ch;

} while (ch != 'n');

return 0;

}

**Вывод:**  


**Задание 10**

Создайте класс с именем ship, который будет содержать данные об учетном номере корабля и координатах его расположения. Для задания номера корабля следует использовать механизм, аналогичный описанному в упражнении 8. Для хранения координат используйте два поля типа angle (см. упражнение 7). Разработайте метод, который будет сохранять в объекте данные о корабле, вводимые пользователем, и метод, выводящий данные о корабле на экран. Напишите функцию main(), создающую три объекта класса ship, затем запрашивающую ввод пользователем информации о каждом из кораблей и выводящую на экран всю полученную информацию.

**Код:**

#include <iostream>

using namespace std;

class angle

{

private:

int \_degrees;

float \_minutes;

char \_direction;

public:

angle()

{

}

angle(int degrees, float minutes, char direction) :

\_degrees(degrees), \_minutes(minutes), \_direction(direction)

{

}

void input()

{

char ch, direction;

int degrees;

float minutes;

cin >> degrees >> ch >> minutes >> ch >> direction;

\_degrees = degrees;

\_minutes = minutes;

\_direction = direction;

}

void display()

{

cout << \_degrees << '°' << \_minutes << "' " << \_direction;

}

};

class ship

{

private:

static int \_amount;

int \_id;

angle \_latitude;

angle \_longitude;

public:

ship()

{

\_id = ++\_amount;

}

void display()

{

cout << "Корабль №" << \_id << " с координатами ";

\_latitude.display();

cout << ", ";

\_longitude.display();

cout << endl;

}

void input()

{

char ch;

\_latitude.input();

cin >> ch;

\_longitude.input();

}

};

int ship::\_amount = 0;

int main()

{

setlocale(LC\_ALL, "");

ship s1, s2, s3;

cout << "Введите координаты трёх кораблей:\n";

s1.input();

s2.input();

s3.input();

cout << "Введённые данные:\n";

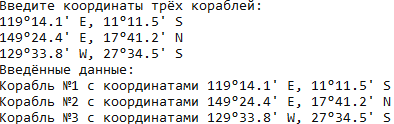
s1.display();

s2.display();

s3.display();

return 0;

}

**Вывод:**  


**Задание 11**

Модифицируйте калькулятор, созданный в упражнении 12 главы 5 так, чтобы вместо структуры fraction использовался одноименный класс. Класс должен содержать методы для ввода и вывода данных объектов, а также для выполнения арифметических операций. Кроме того, необходимо включить в состав класса функцию, приводящую дробь к несократимому виду. Функция должна находить наибольший общий делитель числителя и знаменателя и делить числитель и знаменатель на это значение. Можно вызывать данную функцию в конце каждого метода, выполняющего арифметическую операцию, либо непосредственно перед выводом на экран результата. Кроме перечисленных методов, вы можете включить в класс конструктор с двумя аргументами, что также будет полезно.

**Код:**

#include <iostream>

#include <string>

#include <cmath>

using namespace std;

class fraction

{

private:

int \_dividend;

int \_divider;

public:

fraction()

{

}

fraction(int dividend, int divider) : \_dividend(dividend), \_divider(divider)

{

lowterms();

}

void parse(string str)

{

\_dividend = 0;

\_divider = 0;

int i, j;

int len = str.length();

for (i = 0; i < len && str[i] != '/'; i++);

for (j = i - 1; j >= 0; j--)

\_dividend += (str[j] - '0') \* pow(10, i - j - 1);

for (j = len - 1; j > i; j--)

\_divider += (str[j] - '0') \* pow(10, len - j - 1);

}

void sum(fraction f)

{

\_dividend = \_dividend \* f.\_divider + \_divider \* f.\_dividend;

\_divider = \_divider \* f.\_divider;

lowterms();

}

void sub(fraction f)

{

\_dividend = \_dividend \* f.\_divider - \_divider \* f.\_dividend;

\_divider = \_divider \* f.\_divider;

lowterms();

}

void mul(fraction f)

{

\_dividend = \_dividend \* f.\_dividend;

\_divider = \_divider \* f.\_divider;

lowterms();

}

void div(fraction f)

{

\_dividend = \_dividend \* f.\_divider;

\_divider = \_divider \* f.\_dividend;

lowterms();

}

void display()

{

cout << \_dividend << '/' << \_divider << endl;

}

void lowterms() //Сокращение дроби

{

long tnum, tden, temp, gcd;

tnum = labs(\_dividend); // Используем неотрицательные значения

tden = labs(\_divider); // Нужен cmath

if (tden == 0) { // Проверка знаменателя

cout << "Недопустимый знаменатель"; exit(1);

}

else if (tnum == 0) {

\_divider = 1;

return;

}

// Нахождение наибольшего общего делителя

while (tnum != 0) {

if (tnum < tden) { // Если числитель больше знаменателя, меняем их местами.

temp = tnum;

tnum = tden;

tden = temp;

}

tnum -= tden;

}

gcd = tden; // Делим числитель и знаменатель на НОД.

\_dividend /= gcd;

\_divider /= gcd;

}

};

int main()

{

setlocale(LC\_ALL, "");

fraction f1, f2;

char oper, ch;

string str1, str2;

do {

cout << "\nВведите выражение с дробями: " << endl;

cin >> str1 >> oper >> str2;

f1.parse(str1);

f2.parse(str2);

switch (oper)

{

case '+':

f1.sum(f2);

break;

case '-':

f1.sub(f2);

break;

case '\*':

f1.mul(f2);

break;

case '/':

f1.div(f2);

break;

}

cout << "Answer = ";

f1.display();

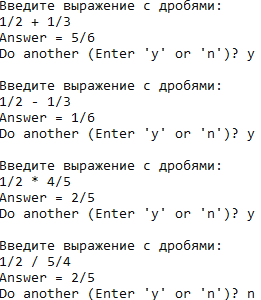
cout << "Do another (Enter 'y' or 'n')? ";

cin >> ch;

} while (ch != 'n');

return 0;

}

**Вывод:**  


**Задание 12**

Используйте преимущество ООП, заключающееся в том, что однажды созданный класс можно помещать в другие программы. Создайте новую программу, которая будет включать класс fraction, созданный в упражнении 11. Программа должна выводить аналог целочисленной таблицы умножения для дробей. Пользователь вводит знаменатель, а программа должна подобрать всевозможные целые значения числителя так, чтобы значения получаемых дробей находились между 0 и 1. Дроби из получившегося таким образом набора перемножаются друг с другом во всевозможных комбинациях, в результате чего получается таблица.

**Код:**

#include <iostream>

#include <iomanip>

#include "fraction.h"

using namespace std;

int getlen(int a)

{

int res = 1;

while (a / 10 != 0)

{

a /= 10;

res++;

}

return res;

}

// Дополнительный метод в классе fraction

//string get()

//{

// return to\_string(\_dividend) + '/' + to\_string(\_divider);

//}

int main()

{

setlocale(LC\_ALL, "");

fraction f;

int a, len;

cout << "Введите знаменатель дроби: ";

cin >> a;

len = getlen(a\*a) \* 2 + 2;

cout << setw(len) << " ";

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

{

f = fraction(i, a);

cout << setw(len) << f.get();

}

cout << endl;

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

{

f = fraction(i, a);

cout << setw(len) << f.get();

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

{

f = fraction(j, a);

f.mul(fraction(i, a));

cout << setw(len) << f.get();

}

cout << endl;

}

return 0;

}

**Вывод:**

