

Лабораторная работа № 4
«Разработка консольных проектов Visual Studio
с использованием функций VC++»
по Разделу
«Функции VC++ и консольные проекты
Visual Studio»

Цель выполнения ЛР: *сформулировать самостоятельно*

4.1 Общее задание

- 1) Изучите структуру программного кода консольных проектов Visual Studio и правила определения, объявления и вызова функций VC++.
- 2) Выберите индивидуальный вариант задания из таблицы 3.1.
- 3) Решите задачу вычисления заданного арифметического выражения с использованием функций VC++ (без использования функций она уже решена в предыдущей работе 2). Для этого разработайте три варианта схем алгоритмов и соответствующих процедур, реализующих решения задачи:
 - схемы алгоритмов для вычисления заданного арифметического выражения:
 - схему алгоритма процедуры с входными параметрами и возвращаемым значением;
 - схему алгоритма процедуры с входными и выходными параметрами и без возвращаемого значения;
 - схему алгоритма без параметров и без возвращаемого значения;
 - программные коды трех функций и функции `main` в соответствии со схемами алгоритмов.
- 4) Создайте консольное решение, содержащее пять проектов, каждый из которых содержит одну из разработанных функций п.3 и главную функцию `main`, в которой осуществляется ввод исходных данных, вызов соответствующей функции п.3 и вывод результата:
 - функция с параметрами и возвращаемым значением, причем определение функции должно быть записано перед функцией `main`.
 - функция с параметрами и возвращаемым значением, причем определение функции должно быть записано после функции `main`.
 - функция с параметрами и без возвращаемого значения.
 - функция без параметров и без возвращаемого значения (с глобальными переменными).
 - функция с параметрами и возвращаемым значением, причем определение функции и `main` должны находиться в разных файлах.

Каждый способ должен быть реализован в отдельном проекте, а все пять проектов должны быть объединены в одном решении.

- 5) Выполните созданные проекты и получите результаты. Убедитесь в идентичности и правильности результатов, полученных при выполнении каждого из пяти проектов.
- 6) Проведите эксперименты, описанные в примере выполнения задания. Внесение изменений в программный код выполняйте путем комментирования исходного кода с последующим удалением комментария для возврата к исходному состоянию. После внесения изменений выполняйте повторную компиляцию и перестроение решения

4.2 Варианты индивидуальных заданий

Таблица 4.1 – Варианты индивидуальных заданий

№	Формулы для вычислений	Исходные данные	
		x	y
1)	$t = \cos \frac{\pi}{7} * \frac{\sin^2(x - 8y)}{2,7(x - \pi)}$	3,59	17,53
2)	$d = \frac{(1 - e^{xy})^2}{0,7 \lg 1 - x^2 }$	1,674	-0,533
3)	$h = \frac{xy + \sin x}{ 1 - y * \ln x}$	32,01	-0,4917
4)	$c = \frac{(yx^2 - 1)^2}{2} \cdot (\cos^2 y - \sin x^2)$	2,123	-1,89
5)	$b = \sqrt[3]{\frac{x + y}{0,2x}} \cdot \sin(\operatorname{tg}^2 x)$	8,402	-0,22226
6)	$d = \frac{xe^{xy} + 8 \sin^2 x}{x(x - y)(3x + y)}$	1,002	-0,5321
7)	$z = \frac{\pi}{2} - \sqrt{2x} - \frac{x + y^2}{0,75 \operatorname{tg} x + y }$	12,003	-5,408
8)	$d = \frac{xy^2 - \sqrt{ x^2 - 2,5 \cdot 10^{-3} y }}{2 \sin xy} + 0.5$	1,3802	-1,9
9)	$f = 5,2^3 \cdot \frac{\lg(x + y)}{x - \frac{1}{0,45 \sin(x - 8y)}} + 0.5$	12,678	6,9
10)	$a = 0,8 \cdot 10^{-5} (xe^{-x(y-1,2)} - yx)^3$	82,578	1,4517
11)	$d = \frac{\sqrt{ x } + e^{-y}}{5,8 \cdot \cos y^3}$	-4.62	-0.32
12)	$f = -\frac{2x^2 - \sin x^2}{2 - e^{-y}}$	5.305,	-0.084

13)	$h = \frac{\sin^3 x + e^{-\sin y}}{0,6x^2y^2}$	0.68	-0.848
14)	$a = 10 \cdot \frac{\ln y^2 - \sqrt[4]{ x-y }}{1 - \cos^3 y}$	12.43	-0.2
15)	$c = \frac{1}{2\pi} - x\sqrt{2,5 \cdot 10^3 y} \cdot \cos x^3 $	-1.462	0.577
16)	$b = \frac{\lg x - \sin^2 xy}{0,8 \cdot \ln(1-x)^2}$	0.501	1.7
17)	$d = 10^4 \cdot \frac{e^{\frac{x}{2y}} + \sqrt{ \sin y^3 }}{2,5 \cos^2 x}$	-0.45	24.3
18)	$f = \frac{\frac{\pi}{3} + \ln x^3}{3y - x} + x \cdot \sin y^2$	8.8	5.8
19)	$h = \frac{208 \cdot \lg x + x^2}{ x - y^2 - e^{-y}}$	0.6485	3.9
20)	$a = 10^5 \cdot \lg 0.8x \cdot e^{\frac{-x^2}{2xy}}$	125.04	5.55
21)	$b = \frac{x^y}{1 - \frac{1}{e^{-x+\sin y}}}$	7.302	5.782
22)	$c = x \cdot \lg x-6 - \frac{\sin x^2}{yx^3}$	-3.356	0.049
23)	$a = \frac{14 \cdot \sin x + y^2}{0.92 \cdot \cos^3 x}$	1.892	-0.61
24)	$a = \frac{x^2 - xy}{0.7 \sin \ln x }$	45.892	0.601
25)	$c = \frac{2.71x^2 - \cos y}{\operatorname{tg}(x^2) \cdot e^{-y}}$	13	0.54321
26)	$d = \frac{1 - \operatorname{tg} xy^2}{\sqrt[3]{x}} + 4\sqrt{x^2 - 0,1}$	4213	104.2.654
27)	$f = 0.5 + \frac{1}{2} \cdot \cos \frac{1 - \sin xy^2}{1 + \sin^2 xy}$	29.999	165.091
28)	$g = x \cdot e^{-y} + \frac{(x+y)^2}{2 \cdot \cos^3 x}$	-9.20	-5,892
29)	$z = \frac{x-y}{\sqrt{x+y}} + \frac{xy^2}{\sin x^2 \cdot \cos^2 y}$	4.51	-0.25
30)	$b = \left \pi - \frac{x}{3} \right \cdot e^{\frac{1 - \sin e^{-y}}{2x}}$	27	222.1

4.3 Содержание отчета

Отчет по ЛР состоять из следующих пунктов:

Титульный лист с указанием номера и названия работы, варианта индивидуального задания, группы и Ф.И.О. студента, Ф.И.О. преподавателя.

Название ЛР.

Цель выполнения ЛР.

- 1) Общее задание.
- 2) Индивидуальное задание.
- 3) Формализация и уточнение задания.
- 4) Программные проекты и результаты их выполнения (рисунки, таблицы и скриншоты, отражающие этапы разработки проектов):
 - схемы алгоритмов процедур;
 - программные проекты, включающую функцию VC++, реализованную различными (пятью) способами, отличающимися способами обмена данными и местоположением функций;
 - результаты выполнения проектов;
 - скриншоты с результатами экспериментов или сообщениями об ошибках, их объяснение.
- 5) Доказательство правильности результатов.

4.4 Пример выполнения задания

1) Общее задания на разработку программного проекта

...

2) Индивидуальное задания на разработку программного проекта

Создать решение (например, с именем **Lab5**), состоящее из пяти программных проектов, для вычисления арифметического выражения:

$$z = \frac{0.002 - e^{xy}}{(100 - y) \cdot (x + \pi)}$$

при значениях исходных данных $x=22.5$ и $y=0.7$ с использованием различных способов обмена данными и местоположения функций в соответствии с общим заданием.

3) Формализация и уточнение задания

Для формализации и уточнения задания определим, что исходные данные x , y – вещественного типа **double**. Результаты вычислений – переменная z также должна быть вещественного типа **double**. Операция вычисления z будет записываться следующим оператором VC++:

$$z = (0.002 - \exp(\text{pow}(x, y))) / ((100 - y) * (x + M_PI));$$

Вычисление z реализуем в функциях VC++ тремя различными способами в соответствии с общим заданием.

4) Разработка пяти программных проектов в одном решении и получение результатов их работы

Создадим пять проектов в одном решении. Для этого, *в отличие от предыдущих лабораторных работ*, при создании первого проекта необходимо поставить галочку в переключателе **Создать каталог для решения**. Проектам и решению необходимо дать разные имена. Решению дадим имя **lab5**, а проекту – имя **pr1** (рисунок 4.1).

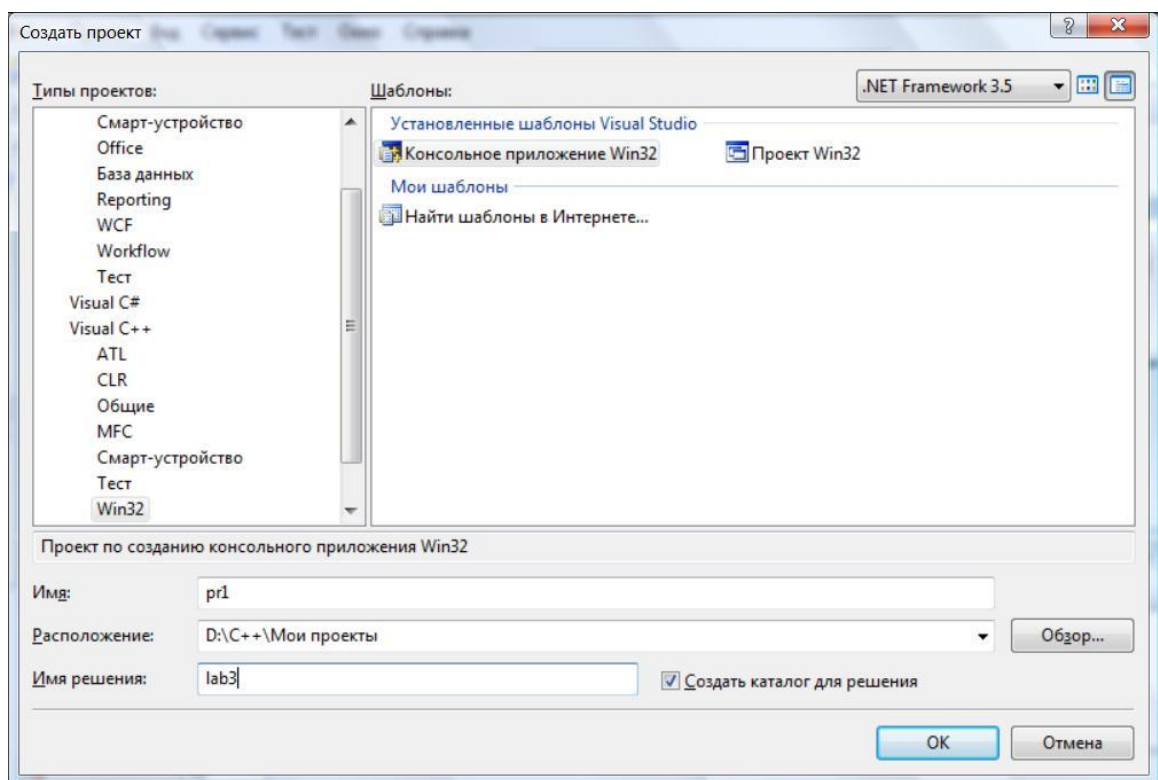


Рисунок 4.1 – Создание нового проекта и решения

Далее, создадим пустой проект и добавим в него файл с пустым исходным кодом и с именем **zad1.cpp**.

Реализация 1-го проекта

Алгоритм главной процедуры не зависит от способа обмена данными и приведен на рисунке 4.2.

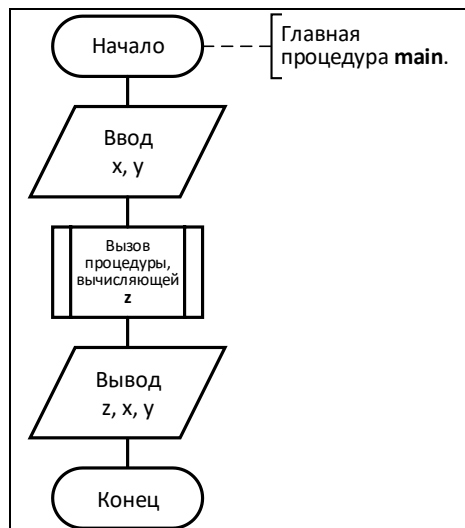


Рисунок 4.2 – Схема алгоритма главной процедуры **main** для всех проектов

Схема алгоритма процедуры **func1** с параметрами и возвращаемым значением представлена на рисунке 4.3.

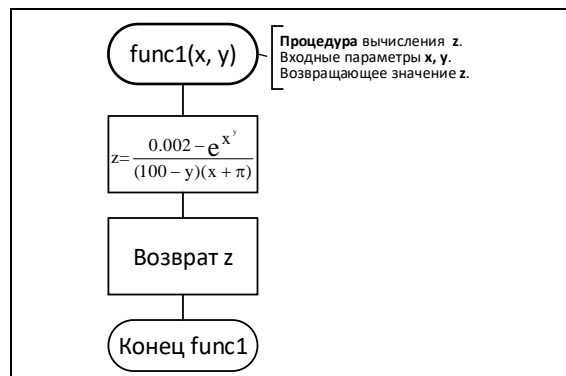


Рисунок 4.3 – Схема алгоритма процедуры **func1** с параметрами и возвращаемым значением для первого проекта

Программные коды разработанных функций запишем в файл с именем **zad1.cpp** в следующем порядке (рисунок 4.4):

- сначала определение функции **func1** с параметрами и возвращаемым значением;
- после него главную функцию **main**.

```

// «Разработка консольных проектов и решения VS
// линейной структуры с использованием функций VC++»

// 1 проект – функция func1 с входными параметрами и возвращаемым
// значением вначале, а после нее программный код главной функции main
#include <iostream>
#include <cmath>
#define _USE_MATH_DEFINES
#include "math.h"
using namespace std;
  
```

```

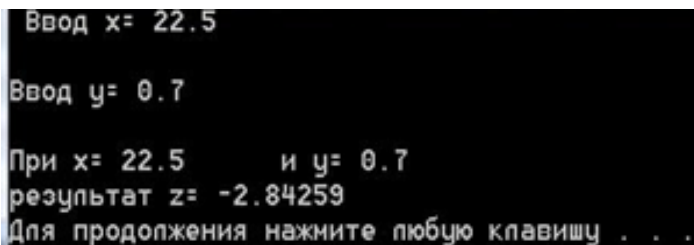
// Определение функции func1
double func1(double x, double y)
{
    double z;    // Локальный объект
    z = (0.02 - exp(pow(x, y)))/((100 - y) * (x + M_PI));
    return z;    // Возврат z в вызывающую функцию
}

int main()
{
    setlocale(LC_ALL, "rus");
    double x, y, z;
    cout<<" Ввод x= ";
    cin>>x;
    cout<<endl<<"Ввод y= ";
    cin>>y;
    z = func1(x, y); // Вызов функции func1
    cout<<endl<<"результат z= "<<z<<endl;
    system("PAUSE");
    return 0;
}

```

Рисунок 4.4 – Программный код первого проекта

Откомпилируем файл **zad1.cpp**, выполним построение решения **lab5** и выполнение проекта **pr1**. Получим следующие результаты при заданных значениях исходных данных (рисунок 4.5).



```

Ввод x= 22.5
Ввод y= 0.7
При x= 22.5    и y= 0.7
результат z= -2.84259
Для продолжения нажмите любую клавишу . . .

```

Рисунок 4.5 – Результаты выполнения проекта **pr1**

Проведем следующие эксперименты, используя при необходимости пошаговую отладку:

- проверим, зависит ли результат выполнения проекта от порядка фактических параметров в функции **main**, заменив оператор вызова функции **z=func1(x, y)** на оператор **z=func1(y, x)**.

...Скриншот с результатами и их объяснение...

- проверим, изменится ли значение переменной **x** в функции **main**, если внутри функции **func1** до оператора **return z** изменить значение **x**, например, добавив оператор **x++**:

```
// Определение функции
double func1(double x, double y)
{
    double z; // Локальный объект
    z = (0.02 - exp(pow(x, y)))/((100 - y) * (x + M_PI));
    x++; // Изменение значения формального параметра x
    return z; // Возврат z в вызывающую функцию
}
```

...Скриншот с результатами и их объяснение...

- проверим, можно ли при вызове функции в списке фактических параметров указывать не имя переменной, а константу или выражение, например, для следующих операторов:

```
z = func1(x+2, y-0.5);
// или
z = func1(22.5, 0.7);
```

...Скриншот с результатами и их объяснение...

- проверим, изменится ли результат работы проекта, если, ничего не меняя в главной функции **main**, изменить имена формальных параметров при определении функции **func1** следующим образом:

```
// Определение функции func1
double func1(double a, double b)
{
    double z;
    z = (0.02 - exp(pow(a, b)))/((100 - b) * (a + M_PI));
    return z;
}
```

...Скриншот с результатами и их объяснение...

Реализация 2-го проекта

Создадим второй пустой проект с именем **pr2** в уже существующем решении. Для этого выполним команду **Файл/Добавить/Создать проект**. В открывшемся окне **Добавить новый проект** зададим имя проекта **pr2**.

Запишем в файл с именем **zad2.cpp** программные коды разработанных в первом проекте функций (переименовав функцию **func1** в **func2**) в следующем порядке:

- сначала программный код главной функции **main**;
- после него определение функции **func2** с параметрами и возвращаемым значением.

Перед кодом главной функции **main** запишем прототип функции **func2**:

```
double func1(double, double);
```

Откомпилируем файл **zad2.cpp** и выполним перестроение решения **lab5**.

Чтобы выполнить второй проект, его надо назначить запускаемым проектом. Для этого необходимо выделить в окне **Обозреватель решений** имя проекта **pr2** и нажать правую кнопку мыши, а затем выполнить команду **Назначить запускаемым проектом** (рисунок 4.6).

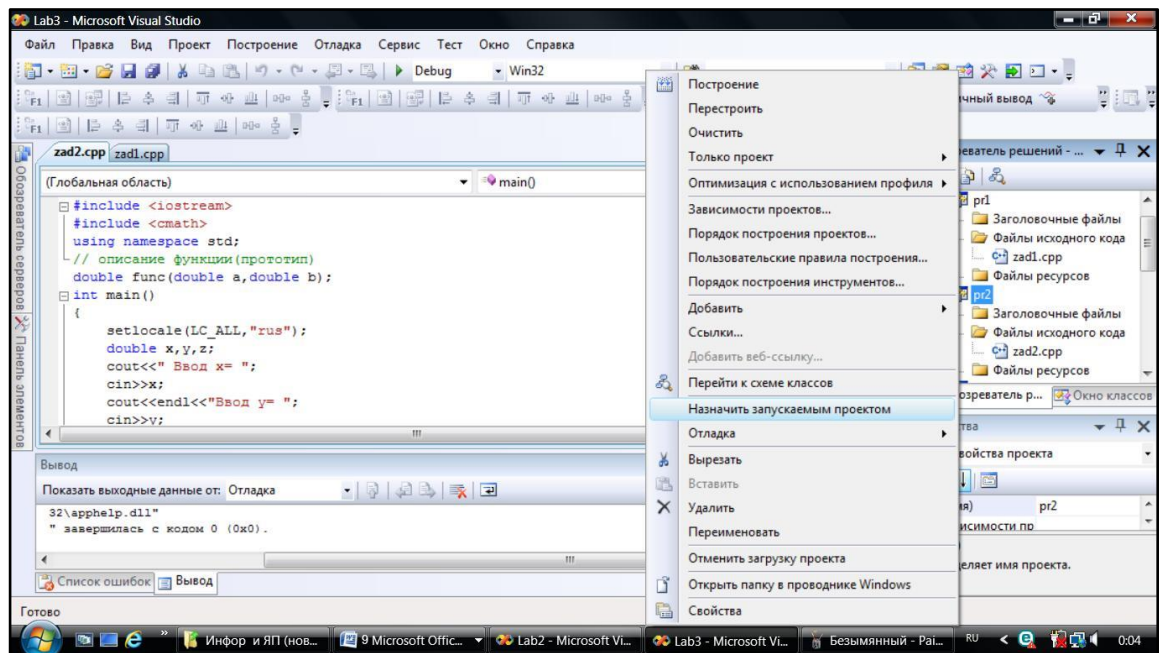


Рисунок 4.6 – Назначение запускаемым проектом

Выполним проект **pr2**. Получим следующие результаты при заданных значениях исходных данных (рисунок 4.7).

```
Ввод x= 22.5
Ввод y= 0.7
При x= 22.5    и y= 0.7
результат z= -2.84259
Для продолжения нажмите любую клавишу . . .
```

Рисунок 4.7 – Результаты выполнения проекта **pr2**

Проведем следующий эксперимент: закомментируем прототип функции **func2** перед кодом главной функции **main**

...Скриншот с результатами и их объяснение...

Реализация 3-го проекта

Создадим третий проект с именем **pr3** в уже имеющемся решении.

Разработаем алгоритм процедуры с параметрами и без возвращаемого значения. Схема алгоритма этой процедуры **func3** представлена на рисунке 4.8.



Рисунок 4.8 – Схема алгоритма процедуры **func3** с параметрами и без возвращаемого значения для третьего проекта

Разработаем программные коды двух функций в соответствии со схемами алгоритмов. Программные коды разработанных функций запишем в файл с именем **zad3.cpp** в следующем порядке (рисунок 4.9):

- объявление (прототип) функции **func3**;
- определение функции **main**;
- определение функции **func3** с входными и выходными параметрами без возвращаемого значения, где входные данные передаются по значению, а результат вычислений возвращается по ссылке.

```
// «Разработка консольных проектов и решения VS
// линейной структуры с использованием функций VC++»

// 3-й проект – без возвращаемого значения, со ссылкой на выходной параметр
#include <iostream>
#include <cmath>
#define _USE_MATH_DEFINES
#include "math.h"
using namespace std;

// Объявление (прототип) функции func3
void func3(double, double, double&);

int main()
{
    setlocale(LC_ALL, "rus");
    double x, y, z;
    cout<<" Ввод x= "; cin>>x;
    cout<<endl<<"Ввод y= ";
    cin>>y;
    func3(x, y, z); // Вызов функции func3
    cout<<endl<<"результат z= "<<z<<endl;
    system("PAUSE");
    return 0;
}
```

```
// Определение функции func3
void func3(double x, double y, double& z)
{
    z = (0.02 - exp(pow(x, y)))/((100 - y) * (x + M_PI));
}
```

Рисунок 4.9 – Программный код третьего проекта.

Откомпилируем файл **zad3.cpp**, выполним перестроение решения **lab5** и выполнение проекта **pr3**, назначив его запускаемым проектом. Получим следующие результаты при заданных значениях исходных данных (рисунок 4.10).

```
Ввод x= 22.5
Ввод y= 0.7
При x= 22.5      и y= 0.7
результат z= -2.84259
Для продолжения нажмите любую клавишу . . .
```

Рисунок 4.10 – Результаты выполнения проекта **pr3**

Проведем следующие эксперименты:

- проверим, можно ли при вызове функции в списке фактических параметров записывать выражения? Например:

```
func3(x+2, y-0.5, z+1); // Вызов функции func3
```

...Скриншот с результатами и их объяснение...

- проверим, изменится ли результат выполнения проекта, если при определении функции **func3** удалить знак **&** (операция взятия адреса) перед формальным параметром **z**:

```
void func3(double x, double y, double z)
```

...Скриншот с результатами и их объяснение...

Реализация 4-го проекта

Создадим четвертый проект с именем **pr4** в уже имеющемся решении.

Разработаем алгоритм процедуры без параметров и без возвращаемого значения. Схема алгоритма этой процедуры **func4** представлена на рисунке 4.11.

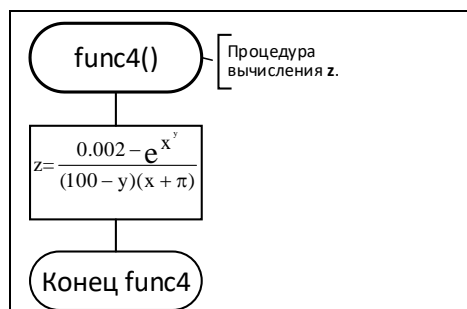


Рисунок 4.11 – Схема алгоритма процедуры **func4** без параметров и без возвращаемого значения для четвертого проекта

Разработаем программные коды двух функций в соответствии со схемами алгоритмов. Программные коды разработанных функций запишем в файл с именем **zad4.cpp** в следующем порядке (рисунок 4.12):

- объявление (прототип) функции **func4**;
- определение глобальных переменных;
- определение функции **main**;
- определение функции **func4** без параметров и без возвращаемого значения.

```

// «Разработка консольных проектов и решения VS
   линейной структуры с использованием функций VC++»

// 4 проект – без параметров и без возвращаемого значения
#include <iostream>
#include <cmath>
#define _USE_MATH_DEFINES
#include "math.h"
using namespace std;

void func4(void ); // Объявление (прототип) функции func4

double x, y, z;    // Глобальные переменные

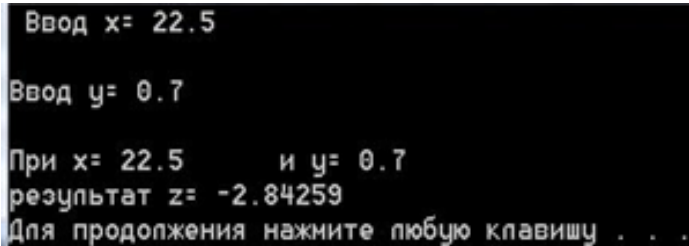
int main()
{
    setlocale(LC_ALL, "rus");
    cout<<" Ввод x= "; cin>>x;
    cout<<endl<<"Ввод y= "; cin>>y;
    func4(); // Вызов функции
    cout<<endl<<"результат z= "<<z<<endl;
    system("PAUSE");
    return 0;
}

// Определение функции func4
void func4()
{
    z = (0.02 - exp(pow(x, y)))/((100 - y) * (x + M_PI));
}

```

Рисунок 4.12 – Программный код четвертого проекта

Откомпилируем файл **zad4.cpp**, выполним перестроение решения **lab5** и выполнение проекта **pr4**, назначив его запускаемым проектом. Получим следующие результаты при заданных значениях исходных данных (рисунок 4.13).

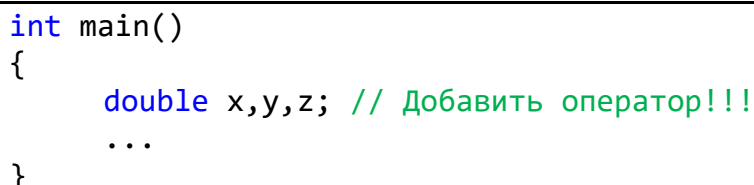


```
Ввод x= 22.5
Ввод y= 0.7
При x= 22.5 и y= 0.7
результат z= -2.84259
Для продолжения нажмите любую клавишу . . .
```

Рисунок 4.13 – Результаты выполнения проекта **pr4**

Проведем следующий эксперимент:

проверим, изменятся ли результаты выполнения проекта, если в тело главной функции **main** добавить определение переменных **x**, **y**, **z** до оператора ввода исходных данных:



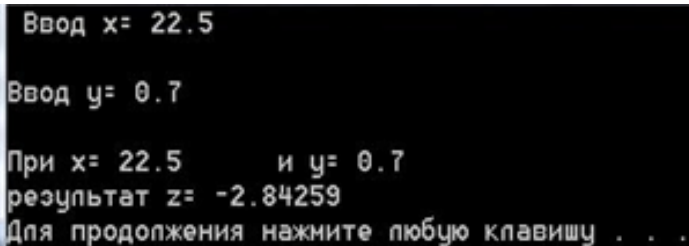
```
int main()
{
    double x,y,z; // Добавить оператор!!!
    ...
}
```

...Скриншот с результатами и их объяснение...

Реализация 5-го проекта

Создадим в уже имеющемся решении пятый проект с именем **pr5**, состоящий из двух файлов. В первый файл с именем **zad5_m.cpp** поместим текст функции **main** из второго проекта. Во второй файл с именем **zad5_f.cpp** поместим текст функции **func2** из того же проекта.

После раздельной компиляции файлов выполним их совместную компоновку (перестроение решения) и выполнение проекта. Результаты выполнения при заданных значениях исходных данных приведены на рисунке 4.14.



```
Ввод x= 22.5
Ввод y= 0.7
При x= 22.5 и y= 0.7
результат z= -2.84259
Для продолжения нажмите любую клавишу . . .
```

Рисунок 4.14 – Результаты выполнения проекта **pr5**

4) Доказательство правильности результатов

Результат выполнения всех пяти проектов одинаков и равен **-2.84259**. Выполним расчет арифметического выражения с использованием калькулятора или программы *Microsoft Excel* и получим совпадающий результат, что доказывает его правильность.