

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ УКРАИНЫ

**НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ХАРЬКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ»**

МЕТОДИЧЕСКИЕ УКАЗАНИЯ

для выполнения практических и лабораторных работ

**ОСНОВЫ ПРОГРАММИРОВАНИЯ НА ЯЗЫКЕ C++.
ЛИНЕЙНЫЕ И РАЗВЕТВЛЯЮЩИЕСЯ
ПРОГРАММЫ**

по курсу
«ИНФОРМАТИКА»

для студентов факультета «Автоматика и приборостроение»
дневной и заочной форм обучения

Утверждено
редакционно-издательским
советом университета,
протокол № 2 от 7.06.2013 г.

Харьков
НТУ «ХПИ»
2013

Методические указания для выполнения практических и лабораторных работ «Основы программирования С++. Линейные и разветвляющиеся программы» по курсу «Информатика» для студентов факультета «Автоматика и приборостроение» дневной и заочной форм обучения /Сост. Тверитникова Е. Е. , Крылова В.А.– Х.: НТУ «ХПИ», 2013. –52с.

Авторы Е. Е. Тверитникова
 В.А. Крылова

Рецензент А. П. Давиденко

Кафедра информационно-измерительных технологий и систем

Вступление

Развитие современных технологий невозможно без использования компьютерной техники и программного обеспечения. Подготовка специалистов в области автоматизации и медицинской техники требует обширных знаний и навыков владения вычислительной техникой, а также знания основ алгоритмизации и программирования на языках высокого уровня, таких как C++.

Методические указания предназначены для изучения языка программирования C++ на практических и лабораторных работах, а также самостоятельного освоения. В методических указаниях на примерах рассматриваются основные средства программирования, а также способы создания блок-схем алгоритма программы на языке C++, а именно: интегрированная среда разработки *Microsoft Visual Studio*, простейшие программы на языке C++, операторы ветвления *if* и *switch*. Приведены индивидуальные задания для выполнения лабораторных работ, задания для самостоятельного изучения основ программирования. Рассмотренные примеры и задания помогут эффективному освоению основ программирования на языке C++.

ЛАБОРАТОРНАЯ РАБОТА 1

ЗНАКОМСТВО С ИНТЕГРИРОВАННОЙ СРЕДОЙ РАЗРАБОТКИ *MICROSOFT VISUALSTUDIO*

Цель работы– знакомство с интегрированной средой разработки *Microsoft Visual Studio* и создание проекта в среде *Microsoft Visual Studio*.
Компиляция, запуски отладка программы.

Интегрированная среда разработки Microsoft Visual Studio

Интегрированная среда разработки (*Integrated Development Environment*) – это программный продукт, объединяющий текстовый редактор, компилятор, отладчик и справочную систему. Любая программа, создаваемая в среде *Microsoft Visual Studio*, всегда оформляется как отдельный проект (*project*). *Проект* – набор взаимосвязанных исходных файлов, компиляция и компоновка которых позволяют создать исполняемую программу. Внешний вид рабочей области проекта (*projectworkspace*) представлен на рис. 1.1.

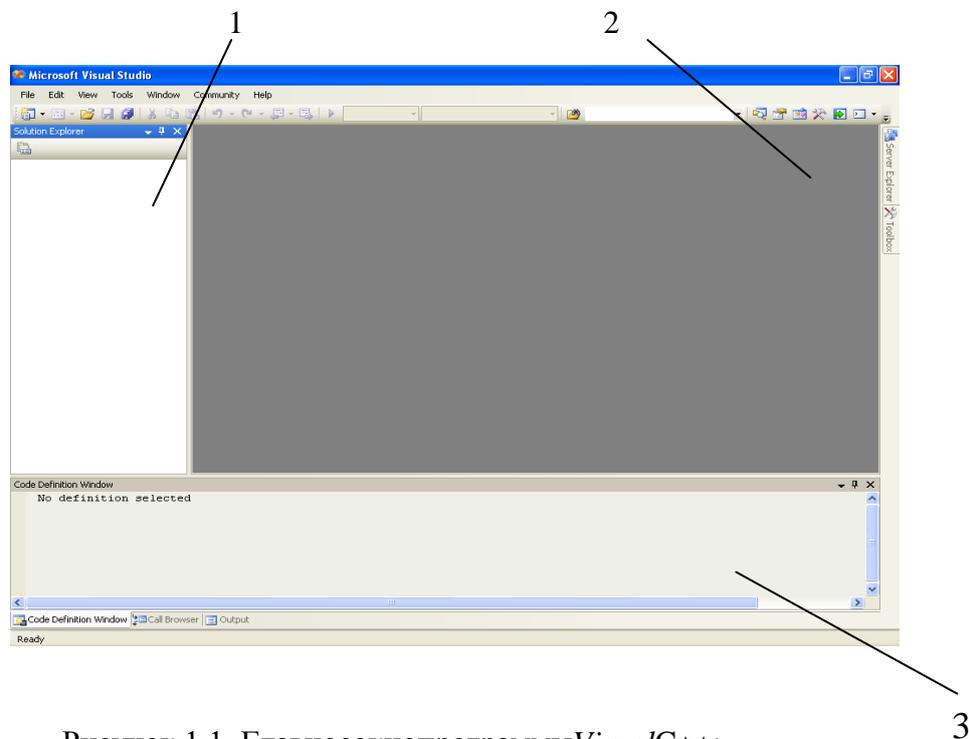


Рисунок 1.1–Главное окно программы *VisualC++*:

1 – Окно «*ProjectWorkspace*»; 2 – Окно *Editor*; 3 – Окно *Output*

Рабочая область проекта может содержать любое количество различных проектов, сгруппированных вместе для согласованной разработки: от отдельного приложения до библиотеки функций или целого программного продукта.

Рабочий стол *VisualC++* включает в себя три окна:

- Окно *Project Workspace* (окно рабочей области), предназначенное для оказания помощи при описании и сопровождении больших многофайловых программ.

- Окно *Editor* (окно редактора) используется для ввода и проверки исходного кода.

- Окно *Output* (окно вывода) служит для вывода сообщений о ходе компиляции, сборки и выполнения программы. В частности, сообщения о возникающих ошибках появляются именно в этом окне.

Интегрированная среда *Visual C++* позволяет строить проекты разных типов, ориентированные на разные сферы применения, также предусмотрена работа с так называемыми консольными приложениями. При запуске консольного приложения операционная система создает так называемое консольное окно, через которое идет весь ввод-вывод программы. Внешне это напоминает работу в режиме командного процессора операционной системы *Windows*.

Создание нового проекта

Для создания проекта типа «консольное приложение» необходимо выполнить следующие действия:

1. Запустить программу *MSVisualStudio*. «Пуск» – «Программы» – «*Microsoft Visual Studio*» – *Microsoft Visual Studio*.

2. Выбрать в строке меню главного окна команду *File/New/Project*.

3. В открывшемся диалоговом окне *New Project* необходимо:

- на панели *Project types* выбрать тип проекта *VisualC++/Win32* (поле 1, рис. 1.2);

- на панели *Templates* выбрать тип проекта *Win32Project* (поле 2, рис. 1.2);

- на текстовом поле *Name* ввести имя проекта – *lab1_фамилия* (поле 3, рис. 1.2);

- в текстовом поле *Location* указать папку для размещения файлов проекта – папка группы (поле 4, рис. 1.2).
- левой кнопкой мыши щелкнуть *Ok* (поле 5, рис. 1.2).

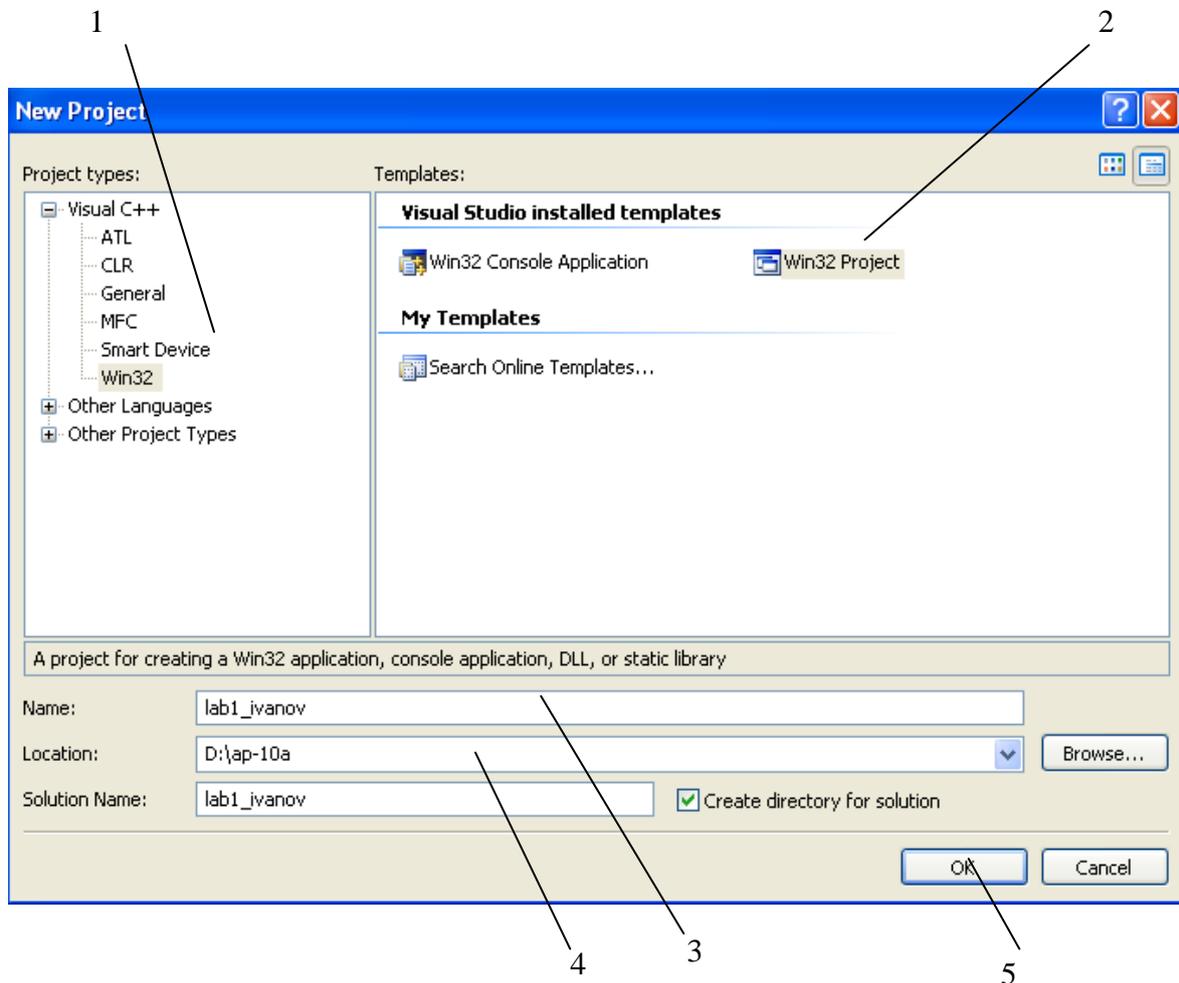


Рисунок 1.2 – Внешний вид диалогового окна *NewProject*

4. После щелчка запускается мастер приложений *Application Wizard*. В открывшемся диалоговом окне необходимо выбрать закладку *ApplicationSettings* – дополнительные установки (поле 1, рис. 1.3):

- в поле *Application type* выбрать необходимый подтип приложения. Для консольного приложения в предлагаемом списке необходимо выбрать *ConsoleApplication* (поле 2, рис. 1.3);
- нажать кнопку *Finish* (поле 3, рис. 1.3).

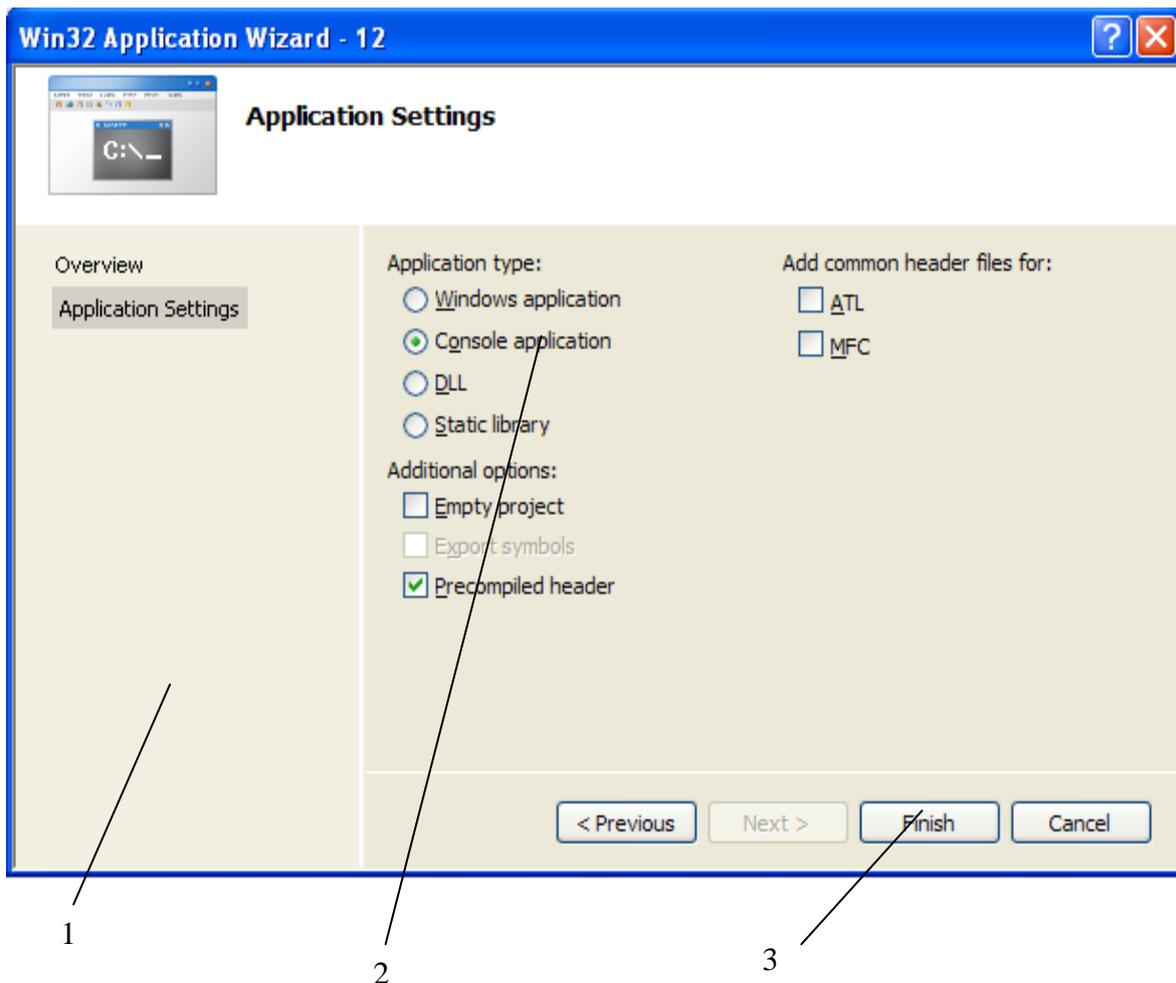


Рисунок 1.3 – Внешний вид диалогового окна *Application Wizard*

5. После щелчка появится рабочая область созданного проекта (рис. 1.4). В окне *Project Workspace* необходимо открыть менеджер проектов (*Solution Explorer*), если он закрыт. Для этого в главном меню выбрать *View/Solution Explorer* (рис. 1.5).

6. В папке *HeaderFiles* двойным нажатием левой кнопкой мыши открыть файл *stdafx.h* (см. рис. 1.5). Необходимо дописать в конец файла следующие строки

```
#include<fstream>
#include<iostream>
using namespace std;
```

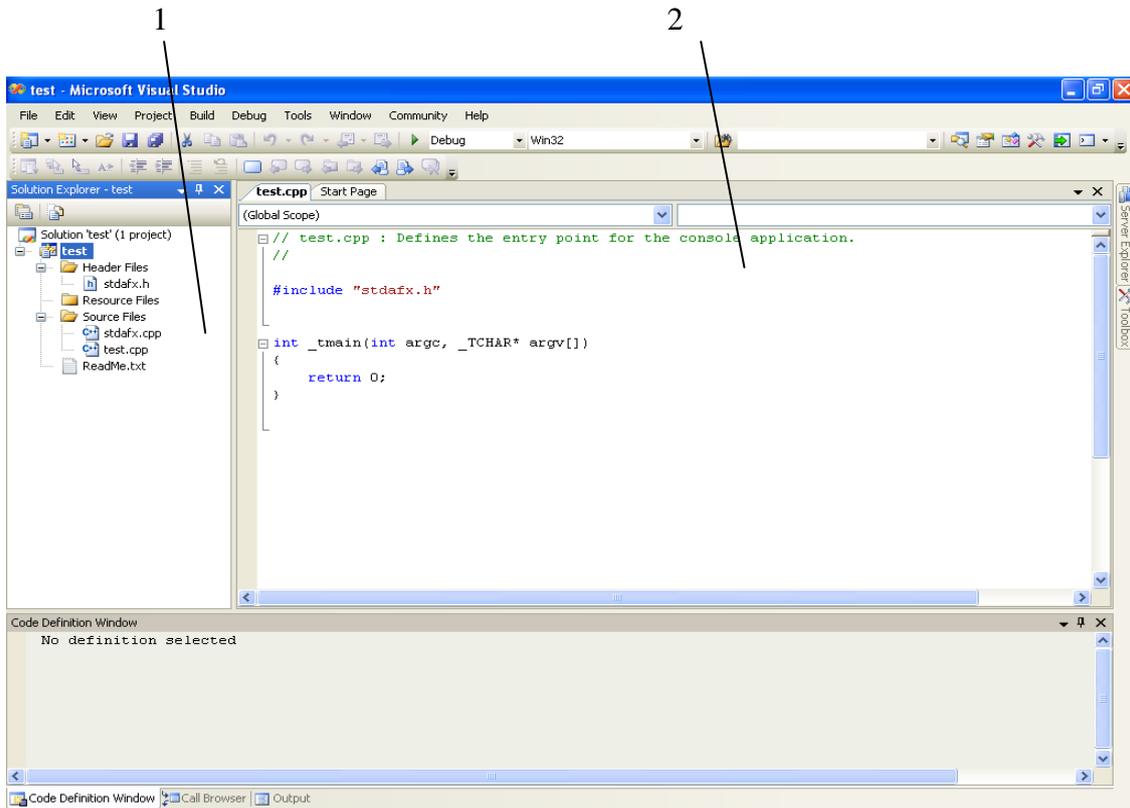


Рисунок 1.4 – Внешний вид рабочей области создаваемого проекта

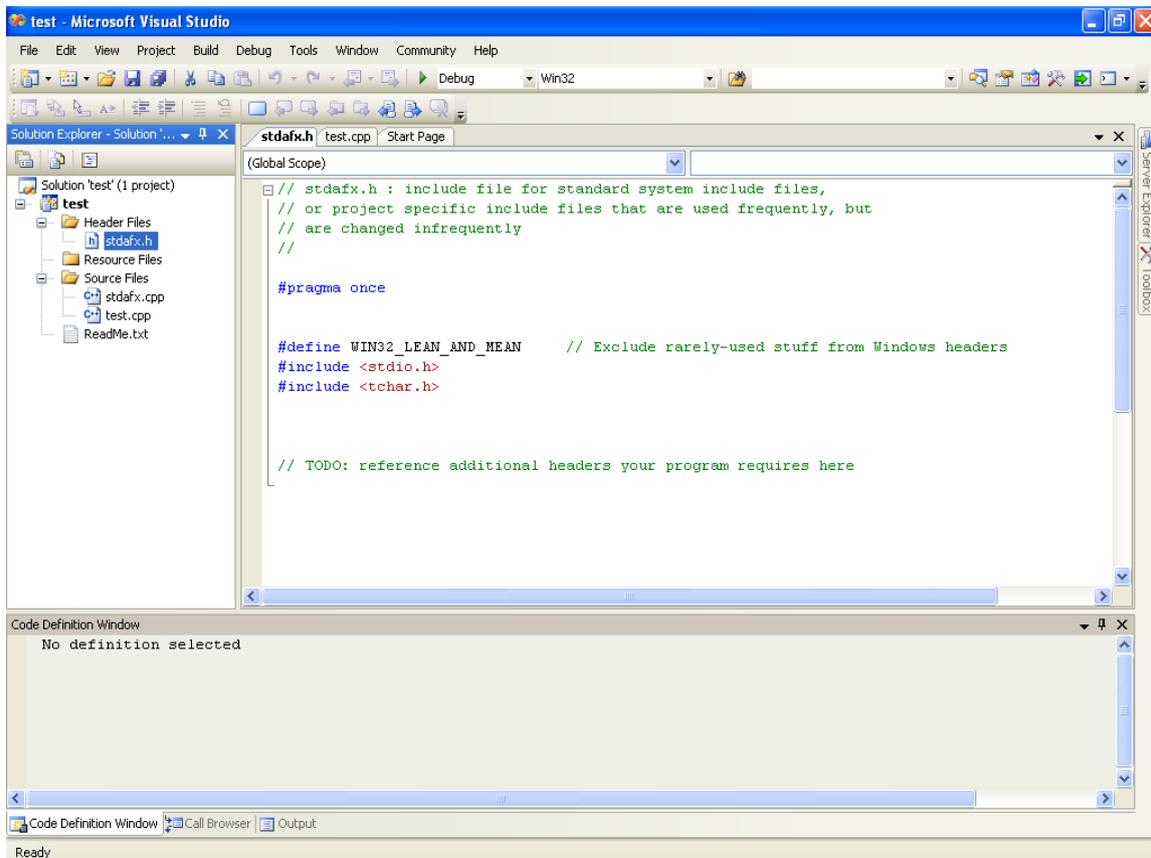


Рисунок 1.5 – Содержимое файла *stdafx.h*

7. Далее в окне *Solution Explorer* в папке *Source Files* необходимо открыть двойным щелчком левой кнопки мыши файл с расширением **.cpp* (*lab1_ivanov.cpp*). В этом файле пишется главная программа создаваемого проекта (рис. 1.6).

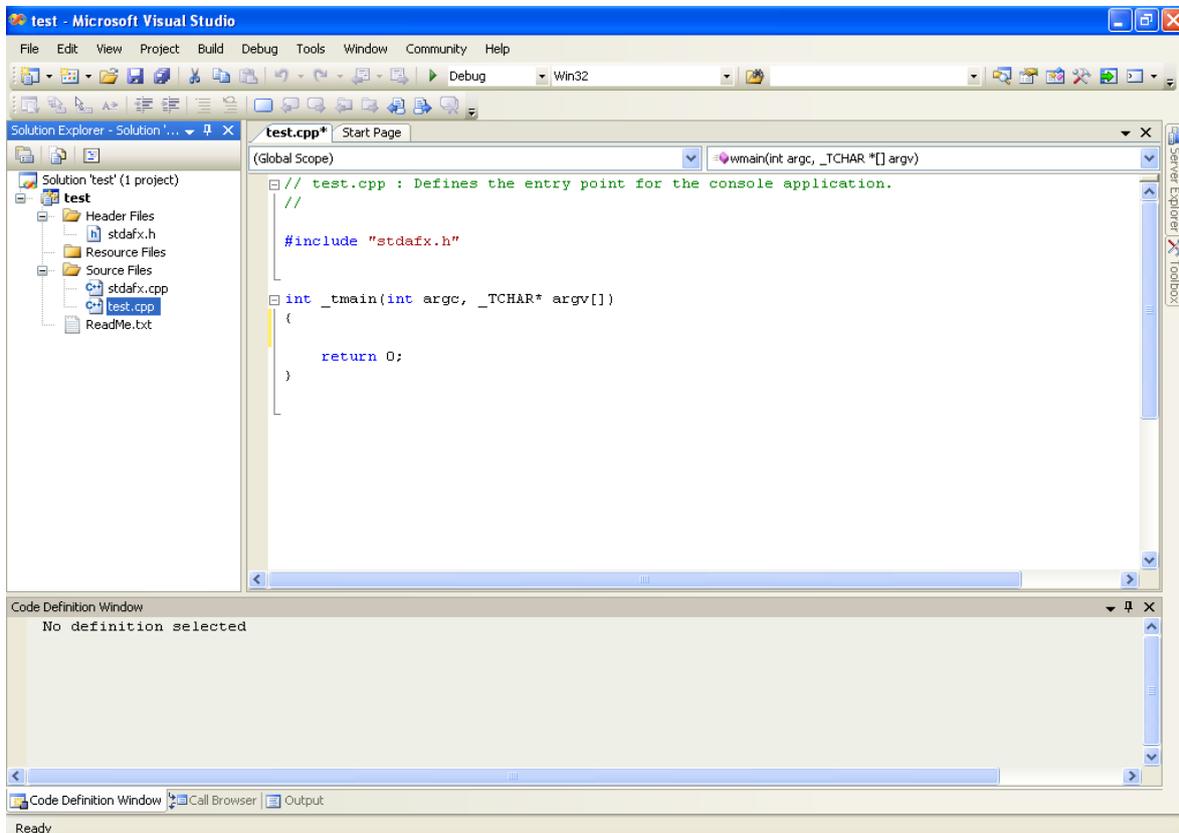


Рисунок 1.6 – Вид главного окна для ввода текста программы

Примечание:

Чтобы открыть проект, с которым вы работали ранее, необходимо выполнить следующие шаги:

1. Выбрать в строке меню команду *Project/OpenWorkspace*.
2. В открывшемся диалоговом окне найти нужный каталог, пользуясь полем *Files*, а в этом каталоге – нужный проектный файл *lab_family.dsw*
3. Открыть этот файл, щелкнув по нему мышью.

Компиляция, компоновка и выполнение проекта

Операции компиляция, компоновка и выполнение проекта могут быть выполнены через главное меню программы с помощью кнопок панели инструментов или с помощью комбинации клавиш. Для того чтобы

откомпилировать созданную программу, необходимо в меню выбрать *Build\BuildSolution*. *Build* – компоновка проекта. Компилируются все файлы, в которых произошли изменения с момента последней компоновки. После компиляции происходит сборка (*link*) всех объектных модулей, включая библиотечные в результирующий исполняемый файл. Сообщение об ошибках компоновки выводится в окно *Output*. О том, что компиляция прошла успешно, можно убедиться по следующей строке:

«1>lab1_ivanov - 0 error(s), 0 warning(s)
 ===== *Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped* =====»

Также для компиляции можно воспользоваться клавишей *F7* или кнопкой на панели инструментов (рис. 1.7).

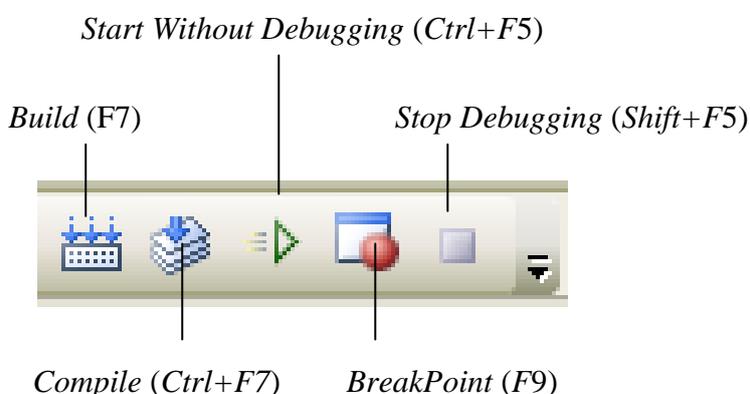


Рисунок 1.7 – Кнопки панели инструментов для выполнения проекта

Для того чтобы откомпилировать указанный файл, необходимо в главном меню выбрать *Build/Compile* или воспользоваться комбинацией клавиш *Ctrl+F7*, а также иконкой на панели инструментов (рис. 1.7).

Если компиляция программы успешно завершена без ошибок, можно запустить программу на выполнение. Для этого в меню необходимо выбрать *Debug/StartWithoutDebugging* или щелкнуть кнопку *StartWithoutDebugging* на панели инструментов (рис. 1.7), а также можно использовать комбинацию клавиш *Ctrl+F5*. После запуска на экране появится консольное приложение с результатами работы программы.

Работа с отладчиком

В интегрированной среде *Visual C++* выполнение программы можно осуществлять как всю целиком, так и построчно, т.е. программу можно

выполнять последовательно, строку за строкой – такой процесс называется пошаговым выполнением. Этот режим позволяет следить за тем, как изменяются значения различных переменных. Иногда он помогает понять, в чем заключается проблема: если обнаруживается, что переменная принимает неожиданное значение, то это может послужить отправной точкой для выявления ошибки. После обнаружения ошибки ее можно исправить и выполнить программу заново в отладочном режиме.

Также в отладочном режиме возможна установка в программе точки прерывания и выполнение программы до заданной точки. Когда во время выполнения встречается точка прерывания, программа останавливается, а на экране появляется отлаживаемый код. Это дает возможность детально выяснить, что происходит в программе.

- *Установка точки прерывания и выполнение программы до точки прерывания*

Точка прерывания позволяет остановить выполнение программы перед любой выполняемой инструкцией (оператором), с тем, чтобы продолжить выполнение программы либо в пошаговом режиме, либо в непрерывном режиме до следующей точки прерывания.

Чтобы задать точку прерывания перед некоторым оператором, необходимо установить перед ним текстовый курсор и нажать клавишу *F9* или щелкнуть на панели инструментов кнопку *BreakPoint* (рис.1.7), также можно воспользоваться главным меню: *Debug/ToggleBreakpoint*. Точка прерывания обозначается в виде красного кружка на левом поле окна редактирования. Повторный щелчок на указанной кнопке снимает точку прерывания. В программе может быть несколько точек прерываний.

После установки точки прерывания программа запускается в отладочном режиме с помощью команды *Debug/StartDebugging* или нажатием клавиши *F5*. В результате код программы выполняется до строки, на которой установлена точка прерывания. Затем программа останавливается и отображает в окне *Editor* ту часть кода, где находится точка прерывания, причем желтая стрелка на левом поле указывает на строку, которая будет выполняться на следующем шаге отладки. Для того чтобы завершить работу с отладчиком, необходимо нажать комбинацию клавиш *Shift+F5*.

- *Пошаговое выполнение программы*

Для пошагового выполнения программы без захода в функции необходимо выбрать в главном меню *Debug/StepOver* или воспользоваться клавишей *F10*. Нажимая клавишу *F10*, можно выполнять один оператор за другим. Если необходимо выполнить пошаговое выполнение программы с кодом вызываемой функции, то надо нажать клавишу *F11* или выбрать в меню *Debug/Step Into*.

После запуска средств отладки изменяется внешний вид и состав окон интегрированной среды разработки. Соответствующие кнопки на панели инструментов позволяют расставлять точки прерывания и руководить пошаговым выполнением программ. Вместо точек прерывания можно воспользоваться функцией «Выполнять к курсору» (*Ctrl+F10*).

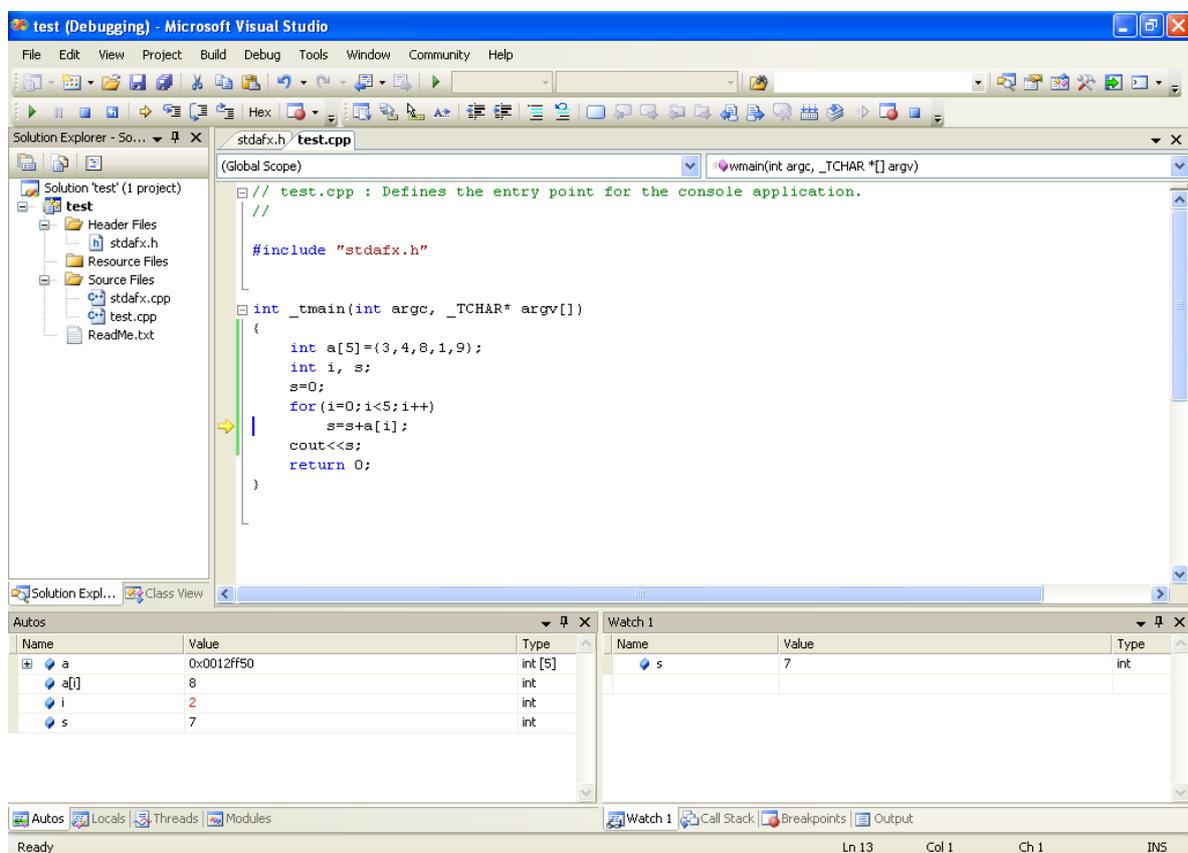


Рисунок 1.8 – Внешний вид окон в отладочном режиме

Во время отладки программы, для того чтобы узнать значение переменной, необходимо задержать над ней указатель мыши, и рядом с именем переменной на экране появится подсказка со значением этой переменной. Помимо экранной подсказки, переменная со своим значением отображается в окне *Auto*, расположенном в левом нижнем углу экрана. В этом окне приведены значения последних переменных, с которыми работал *Visual C++*. Кроме этого, в окне *Watch*, которое находится в правом

нижнем углу, можно задать имя переменной, за значениями которой вы хотите понаблюдать (рис.1.8).

Порядок выполнения лабораторной работы

1. Создать новый проект *File\New\Project*, ввести название проекта (*lab1_фамилия.cpp*).

2. Открыть окно менеджера проектов (*Solution Explorer*) *View\SolutionExplorer*, если оно закрыто.

3. Открыть двойным нажатием на панели *Solution Explorer* файл *stdafx.h*:

– дописать в конец файла следующие строки:

```
#include <fstream>  
#include <iostream>  
using namespace std;
```

4. Открыть двойным нажатием файл *lab1_фамилия.cpp*:

– дописать внутри основной программы (функция *_tmain*) следующие строки:

```
int x = 5, y = 15;  
int z = x + y;  
cout<< "z = " <<z<<endl;
```

Должен получиться следующий текст:

```
int _tmain(int argc, _TCHAR* argv[])  
{  
    int x = 5, y = 15;  
    int z = x + y;  
    cout <<"z = " <<z << endl;  
    return 0;  
}
```

5. Записать в отчет содержимое файлов *stdafx.h* и *lab1_фамилия.cpp*.

6. Откомпилировать получившуюся программу *Build\BuildSolution*.

7. Убедиться, что компиляция прошла успешно.

В окне вывода должны появиться строки

```
<<1>lab1 – 0 error(s), 0 warning(s)
```

```
===== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped=====
```

8. Установить точку прерывания на строке *<<int x = 5, y = 15;>*, нажав левой кнопкой мыши на серой границе слева от строки, или установив курсор на строке и нажав кнопку *F9* – должна появиться красная точка.

9. Запустить программу на выполнение в отладчике *Debug/StartDebugging*;

– выполнение программы остановится на строке с точкой прерывания;

– далее необходимо выполнить одну строку программы *Debug/StepOver*;

– наведите курсор мыши на переменных x и y и посмотрите, какие значения они приняли, запишите их в отчет;

– правой кнопкой мыши нажмите на переменную z и в появившемся меню выберите пункт *Add Watch*;

– выполните одну строку программы *Debug/StepOver*;

– значение переменной z изменилось (подсвечено красным цветом на панели *Watch*), запишите в отчет;

– выберите *Debug/Continue*, для того чтобы закончить выполнение программы.

10. Запустите программу без отладчика *Debug/StartWithoutDebugging*.

11. Запишите в отчет результаты работы программы, появившиеся на экране.

Контрольные вопросы

1. Технология создания программного обеспечения. Этапы создания.

2. Способы представления алгоритма. Назначение.

3. Оболочка *Microsoft Visual Studio*. Назначение. Внешний вид.

4. Создание программного проекта с использованием оболочки *Microsoft Visual Studio*.

5. Языки программирования высокого уровня. Назначения, преимущества и недостатки.

6. Компиляторы, назначение и виды компиляторов.

7. Языки программирования низкого уровня, назначение, преимущества и недостатки.

8. Язык программирования C/C++. Основные понятия, лексемы.

ЛАБОРАТОРНАЯ РАБОТА 2

РАЗРАБОТКА ЛИНЕЙНЫХ ПРОГРАММ НА ЯЗЫКЕ C++

Цель работы– приобретение и закрепление практических навыков при составлении простейших линейных программ на языке программирования C++.

Порядок выполнения работы

1. В соответствии с номером в журнале выберите индивидуальное задание.

2. Разработайте алгоритм решения задачи.

3. Составьте текст программы. При составлении текста программы придерживайтесь общей структуры программы, приведенной в приложении Б.

4. Создайте проект в интегрированной среде разработки *Microsoft Visual Studio*. (*lab2_фамилия.cpp*)

5. Введите текст программы.

6. Скомпилируйте программу. Если в программе есть ошибки, исправьте их. Если ошибок нет, то появится сообщение об успешной компиляции.

7. Запустите программу на выполнение, проанализируйте результаты работы выполнения программы. Убедитесь в правильности решения задачи.

8. Выполните отчет по лабораторной работе (приложение А), который должен содержать:

- титульный лист;
- цель работы;
- индивидуальное задание;
- алгоритм работы программы;
- текст программы;
- результаты работы программы;
- выводы.

ПРИМЕРЫ РЕШЕНИЯ ЗАДАНИЙ

Задание 2.1

Вычислить значение функции $y = 24 \cdot e^x \sqrt{\sin x^2 / \ln(2+x) + 4,5} + \cos^2 x$, где x – вещественное число из интервала от 0 до 1, введенное с клавиатуры. Найти модуль $|y|$ и округлить полученное значение y к ближайшему целому числу. Определить четность полученного числа, а также принадлежность округленного числа интервалу $[2..10]$ или $[12...20]$. Результат вывести на экран.

I. Выбор метода

Для решения задачи используем ряд функций (см. приложение В, табл. В.4):

doublefabs(doublex) – возвращает абсолютное значение аргумента;

doublefmod (doublex,doubley) – функция возвращает остаток от деления значений аргументов x/y ;

doublefloor(doublex) – возвращает наибольшее целое (представленное в виде значения с плавающей запятой), которое меньше значения аргумента или равно ему. Например: если $x = 1,02$, функция *floor* () вернет значение 1,0, а при $x = -1,02$, вернет значение: $-2,0$.

doubleceil(doublex) – возвращает наименьшее целое (представленное в виде значения с плавающей запятой), которое больше значения аргумента или равно ему. Например: если $x = 1,02$, функция *ceil* () вернет значение 2,0, а при $x = -1,02$, вернет значение: $-1,0$.

II. Описание решения задания на псевдокоде

1. Начало.
2. Ввести x .
3. Вычислить y .
4. Вычислить абсолютное значение $|y|$.
5. Округлить z .
6. Определить четность числа z .
7. Определить принадлежность интервалу $[2...10]$ или $[12...20]$.
8. Вывести результаты на экран.
9. Конец.

III. Схема алгоритма программы

Блок-схема алгоритма программы представлена на рисунке 2.1.

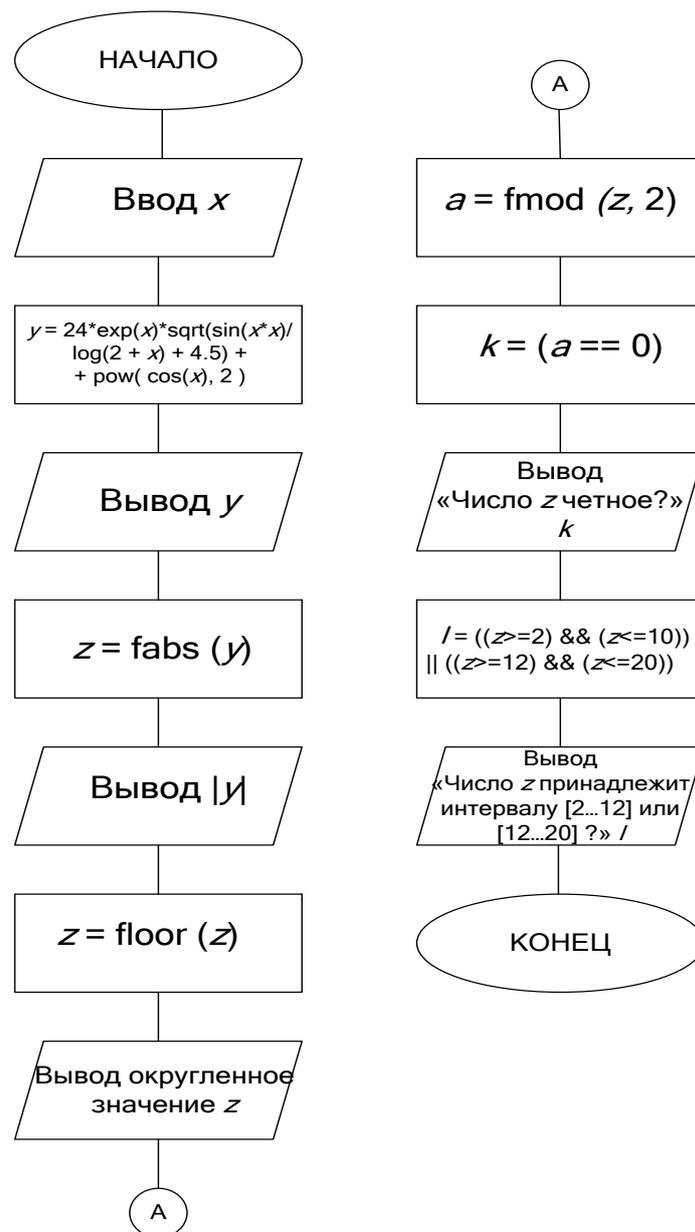


Рисунок 2.1 – Блок-схема алгоритма программы задания 2.1

IV. Разработка текста программы

1. Подключаем в файле *stdafx.h* необходимые для работы программы библиотеки:

#include <iostream> – для работы операторов ввода/вывода.

#include <math.h> – для использования математических функций.

using namespace std;

2. Разработка раздела описания переменных

float x, y, z, a;

boolk, l;

x – вещественная переменная, вводимая с клавиатуры;

y – вещественная переменная для вычисления значения $y(x)$;

z – вещественная переменная для вычисления значения $|y|$;

a – вещественная переменная для нахождения остатка от деления;

k – логическая переменная проверки равен ли остаток $a = 0$;

l – логическая переменная проверка принадлежности *z* интервалу.

3. Разработка тела программы

Исходные данные вводим с клавиатуры. Для подсказки, что вводить, используем оператор вывода ***cout***<< с соответствующим текстом. Затем используем оператор ввода ***cin***>> с указанием необходимых переменных. Например, фрагмент диалогового ввода переменной *x*, значения которой могут находиться в интервале от 0 до 1, имеет вид:

```
cout<<" Введите значение x из интервала [0..1]"<<"\n";  
cin>>x;
```

'\n' – элемент перевода строки, следующее значение будет выводиться на консольном приложении с новой строки.

Синтаксис программы

```
#include<iostream>  
#include<math.h>  
using namespacestd;  
  
int main( )  
{  
floatx, y, z, a, b; // описание переменных вещественного типа  
boolk, l; // описание переменных логического типа  
cout<<" Введите значение x из интервала [0..1]"<<"\n";  
cin>>x; // ввод значения переменной x  
y=24 * exp ( x ) * sqrt (sin (x*x) / log (2 + x)+4.5)+pow( cos ( x ),2); //  
вычисление значения y  
cout<<"y="<<y<<"\n"; //вывод значения y  
z=fabs(y); //вычисление абсолютного значения y  
cout<<" Модуль числа  $|y| =$ "<<z<<"\n"; //вывод абсолютного  
значения y  
z=floor(z); //округление значения z
```

```

    cout<<"Округленное значение z="<< z <<"\n"; //вывод округленного
значения z
    a=fmod(z, 2); // вычисление остатка от деления z на 2
    k=(a==0); // проверка, равен ли остаток от деления 0
    cout<<"Число z четное?"<<k<<"\n"; // вывод четности числа
    l=((z>=2)&&(z<=10))//      ((z>=12)&&(z<=20)); //проверка
принадлежности числа z интервалу
    cout<<"Число z принадлежит интервалу [2...12] или
[12...20]?"<<l<<"\n"; //вывод принадлежности числа z
заданному интервалу
    return 0;
}

```

4. Отладка и запуск программы

Для запуска программы используем комбинацию клавиш *Ctrl+F5*.

Ниже приведены результаты работы программы.

Введите значение x из интервала $[0..1]$:

0.1

$y=57.3404$

Модуль числа $|y|=57.3404$

Округленное значение $z=57$

Число z четное? *FALSE*

Число z принадлежит интервалу $[2...12]$ или $[12...20]$? *FALSE*

Задание 2.2

Написать программу, которая выводит сообщение, есть ли среди цифр трёхзначного целого числа, вводимого с клавиатуры, одинаковые.

I. Выбор метода

Для нахождения одинаковых цифр в трёхзначном целом числе необходимо воспользоваться арифметическими операторами: $\%$ —нахождение остатка от деления двух целых чисел, $/$ — целочисленное деление двух целых чисел.

Например: целое трёхзначное число $n = 962$.

$a = n / 100;$ $a=9$ —первая цифра числа n .

$d = n \% 100;$ $d=62;$

$b = d / 10;$ $b=6$ — вторая цифра числа n .

$c = d \% 10;$ $c=2$ — третья цифра числа n .

II. Описание решения задачи на псевдокоде

1. Начало.
2. Ввести n .
3. Определить первую цифру числа a .
4. Определить вторую цифру числа b .
5. Определить третью цифру числа c .
6. Сравнить три цифры a , b и c .
7. Вывести результат на экран.
8. Конец.

III. Схема алгоритма программы

Блок-схема алгоритма программы представлена на рисунке 2.2.

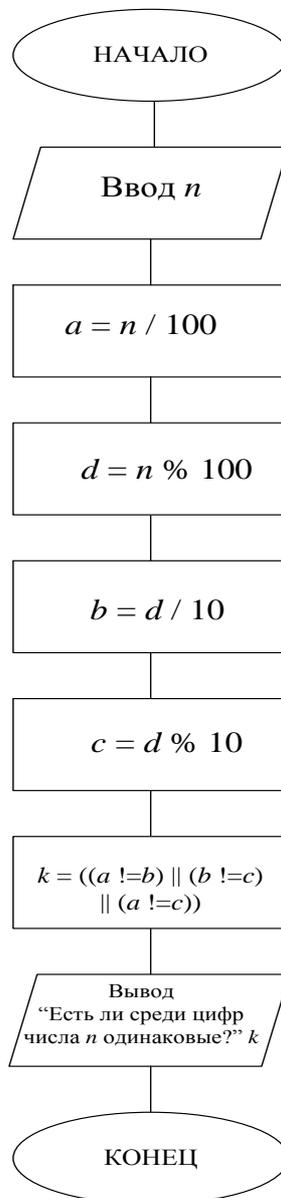


Рисунок 2.2 – Блок-схема алгоритма программы задания 2.2

IV. Разработка текста программы

1. Подключаем в файле *stdafx.h* необходимые для работы программы библиотеки:

```
#include<iostream>– для работы операторов ввода/вывода.  
using namespace std;
```

2. Разработка раздела описания переменных

```
int n, a, b, c, d;  
bool k;
```

n–трёхзначное целое число, вводимое с клавиатуры;

a, b, c– первая, вторая и третья цифра трёхзначного числа *n*;

d– вспомогательная целая переменная;

k– логическая переменная.

3. Разработка тела программы

Исходные данные вводим с клавиатуры. Для подсказки, что пользователь должен ввести целое трёхзначное число, используем оператор вывода *cout<<* с соответствующим текстом. Затем используем оператор ввода *cin>>* с указанием необходимых переменных.

```
cout<< “ Введите целое трехзначное число\n”;  
cin>>n; //ввод трехзначного целого числа с клавиатуры
```

Синтаксис программы

```
#include<iostream>  
using namespace std;
```

```
int main( )
```

```
{
```

```
int n, a, b, c, d;
```

```
bool k;
```

```
cout<< “ Введите целое трехзначное число\n”; //Вывод текста
```

```
cin>>n; //Ввод значения переменной n
```

```
a=n / 100; // Вычисление первой цифры числа n
```

```
d=n % 100; //Вычисление остатка от деления числа n на 100
```

```
b=d / 10; //Вычисление второй цифры числа n
```

```
c=d % 10; //Вычисление третьей цифры числа n
```

```
k=((a != b) || (b != c) || (a != c)); //проверка есть ли среди цифр
```

одинаковые

```

cout<<"Есть ли среди цифр числа"<<n<<" одинаковые ?"<<k;
return 0;
}

```

4. Отладка и запуск программы

Для отладки программы используем клавишу *F7*, убеждаемся в отсутствии ошибок и запускаем программу на выполнение с помощью комбинации клавиш *Ctrl+F5*. Ниже приведены результаты работы программы.

1-й запуск

Введите целое трехзначное число

235

Есть ли среди цифр числа 235 одинаковые ? FALSE

2-й запуск

Введите целое трёхзначное число

565

Есть ли среди цифр числа 565 одинаковые ? TRUE

1. Индивидуальные задания

1. Вычислить значение функции $f(x) = 3,5x^3 + \cos x / \sin 2x + e^x$, где $x=0,1$. Округлить значение $f(x)$ к ближайшему целому числу. Найти остаток от деления целой части на 5 и вывести результат на экран.

2. Вычислить значение функции $f(x) = \cos(x+1) * \operatorname{tg} x / (\ln x + 2,5)$, где x – любое число из интервала $[0...1]$, вводимое с клавиатуры в диалоговом режиме. Определить принадлежность $f(x)$ отрезку $[3...9]$.

3. Ввести три целых двухзначных числа. Найти сумму первых цифр данных чисел. Вывести сообщение о четности последних цифр данных чисел.

4. Ввести три вещественных числа a, b и c . Найти частное от деления $a+b/c$ и округлить его к ближайшему целому. Вывести сообщение о четности полученного числа.

5. Ввести два вещественных числа c, d . Найти число x , соответствующее целому от деления c на d . Определить куб числа x и вывести результат на экран.

6. Ввести два целых числа a и b . Найти остаток от деления a на b . Определить принадлежность остатка интервалу $[0...4]$.

7. Вычислить значение функции $f(x) = |x^3| + x/3,8 \operatorname{arctg} x$ и округлить его до ближайшего целого. Определить остаток от деления $f(x)$ на 10. Число x – любое число из интервала $[0 \dots 1]$, вводимое с клавиатуры.

8. Определить квадрат четырехзначного целого числа, полученного выписыванием в обратном порядке.

9. Определить куб трехзначного числа, полученного выписыванием в обратном порядке целой части вещественного числа.

10. Вычислить целую часть среднего арифметического заданных четырех положительных чисел.

11. Вычислить значение $f(x) = \sin x + \arccos x$, где x – любое число из диапазона $[0 \dots 1]$. Вывести на печать целую часть значения $f(x)$.

12. Определить, четные ли первая и последняя цифры заданного трехзначного числа.

13. Ввести любое вещественное число. Округлить его к ближайшему целому. Вывести сообщение о принадлежности полученного числа отрезку $[0 \dots 50]$.

14. Определить, равна ли сумма двух первых цифр заданного четырехзначного числа сумме двух его последних цифр.

15. Найти произведение цифр заданного четырехзначного числа.

16. Определить число, полученное выписыванием в обратном порядке цифр заданного трехзначного числа.

17. Вычислить значение $f(x) = \ln x / \sin x + e^x$, где $x = 0,7$. Найти остаток от деления целой части $f(x)$ на $y(x)$, где $y(x) = \arcsin x + \arccos x + |x|$.

18. Вычислить длину окружности, площадь круга и объем шара одного и того же заданного радиуса.

19. Вычислить периметр и площадь прямоугольного треугольника по длинам двух катетов.

20. Найти произведение цифр заданного четырехзначного числа.

2. Индивидуальные задания

Напишите программу расчета по двум формулам (таблица 2.1). Результаты вычислений по первой и второй формуле должны совпадать. Для использования математических функций в программе необходимо подключить библиотеку `<math.h>`

Таблица 2.1 – Задания к лабораторной работе №2

<p>Вариант 1</p> $z_1 = 2 \sin^2(3\pi - 2\alpha) \cos^2(5\pi + 2\alpha)$ $z_2 = \frac{1}{4} - \frac{1}{4} \sin\left(\frac{5}{2}\pi - 8\alpha\right)$	<p>Вариант 2</p> $z_1 = \cos \alpha + \sin \alpha + \cos 3\alpha + \sin 3\alpha$ $z_2 = 2\sqrt{2} \cos \alpha \cdot \sin\left(\frac{\pi}{4} + 2\alpha\right)$
<p>Вариант 3</p> $z_1 = \frac{\sin 2\alpha + \sin 5\alpha - \sin 3\alpha}{\cos \alpha + 1 - 2 \sin^2 2\alpha}$ $z_2 = 2 \sin \alpha$	<p>Вариант 4</p> $z_1 = \frac{\sin 2\alpha + \sin 5\alpha - \sin 3\alpha}{\cos \alpha - \cos 3\alpha + \cos 5\alpha}$ $z_2 = \operatorname{tg} 3\alpha$
<p>Вариант 5</p> $z_1 = 1 - \frac{1}{4} \sin^2 2\alpha + \cos 2\alpha$ $z_2 = \cos^2 \alpha + \cos^4 \alpha$	<p>Вариант 6</p> $z_1 = \cos \alpha + \cos 2\alpha + \cos 6\alpha + \cos 7\alpha$ $z_2 = 4 \cos \frac{\alpha}{2} \cdot \cos \frac{5}{2} \alpha \cdot \cos 4\alpha$
<p>Вариант 7</p> $z_1 = \cos^2\left(\frac{3}{8}\pi - \frac{\alpha}{4}\right) - \cos^2\left(\frac{11}{8}\pi + \frac{\alpha}{4}\right)$ $z_2 = \frac{\sqrt{2}}{2} \sin \frac{\alpha}{2}$	<p>Вариант 8</p> $z_1 = \cos^4 x + \sin^2 y + \frac{1}{4} \sin^2 2x - 1$ $z_2 = \sin(y+x) \cdot \sin(y-x)$
<p>Вариант 9</p> $z_1 = (\cos \alpha - \cos \beta)^2 - (\sin \alpha - \sin \beta)^2$ $z_2 = -4 \sin^2 \frac{\alpha - \beta}{2} \cdot \cos(\alpha + \beta)$	<p>Вариант 10</p> $z_1 = \frac{\sin\left(\frac{\pi}{2} + 3\alpha\right)}{1 - \sin(3\alpha - \pi)}$ $z_2 = \operatorname{ctg}\left(\frac{5}{4}\pi + \frac{3}{2}\alpha\right)$
<p>Вариант 11</p> $z_1 = \frac{1 - 2 \sin^2 \alpha}{1 + \sin 2\alpha}$ $z_2 = \frac{1 - \operatorname{tg} \alpha}{1 + \operatorname{tg} \alpha}$	<p>Вариант 12</p> $z_1 = \frac{\sin 4\alpha}{1 + \cos 4\alpha} \cdot \frac{\cos 2\alpha}{1 + \cos 2\alpha}$ $z_2 = \operatorname{ctg}\left(\frac{3}{2}\pi - \alpha\right)$
<p>Вариант 13</p> $z_1 = \frac{\sin \alpha + \cos(2\beta - \alpha)}{\cos \alpha - \sin(2\beta - \alpha)}$ $z_2 = \frac{1 + \sin 2\beta}{\cos 2\beta}$	<p>Вариант 14</p> $z_1 = \frac{\cos \alpha + \sin \alpha}{\cos \alpha - \sin \alpha}$ $z_2 = \operatorname{tg} 2\alpha + \sec 2\alpha$
<p>Вариант 15</p> $z_1 = \frac{\sqrt{2b + 2\sqrt{b^2 - 4}}}{\sqrt{b^2 - 4} + b + 2}$ $z_2 = \frac{1}{\sqrt{b + 2}}$	<p>Вариант 16</p> $z_1 = \frac{x^2 + 2x - 3 + (x+1)\sqrt{x^2 - 9}}{x^2 - 2x - 3 + (x-1)\sqrt{x^2 - 9}}$ $z_2 = \sqrt{\frac{x+3}{x-3}}$

Продолжение таблицы 2.1

<p>Вариант 17</p> $z_1 = \frac{\sqrt{(3m+2)^2 - 24m}}{3\sqrt{m} - \frac{2}{\sqrt{2}}}$ $z_2 = -\sqrt{m}$	<p>Вариант 18</p> $z_1 = \left(\frac{a+2}{\sqrt{2a}} - \frac{a}{\sqrt{2a}+2} + \frac{2}{a-\sqrt{2a}} \right) \cdot \frac{\sqrt{a}-\sqrt{2}}{a+2}$ $z_2 = \frac{1}{\sqrt{a}+\sqrt{2}}$
<p>Вариант 19</p> $z_1 = \left(\frac{1+a+a^2}{2a+a^2} + 2 - \frac{1-a+a^2}{2a-a^2} \right)^{-1} (5-2a^2)$ $z_2 = \frac{4-a^2}{2}$	<p>Вариант 20</p> $z_1 = \frac{(m-1)\sqrt{m} - (n-1)\sqrt{n}}{\sqrt{m^3n + nm + m^2 - m}}$ $z_2 = \frac{\sqrt{m} - \sqrt{n}}{m}$

Контрольные вопросы

1. Языки программирования низкого, среднего и высокого уровня.
2. Что такое компилятор, интерпретатор.
3. Основные лексемы языка программирования C++ (идентификатор, ключевое слово, оператор).
4. Описание переменных, физический смысл переменной.
5. Целые типы данных, назначение, примеры.
6. Стандартные типы данных.
7. Операторы ввода и вывода на языке C++. Назначение, синтаксис.
8. Математические и логические операции.
9. Функции *floor ()* и *ceil ()*.
10. Приоритеты операций в выражении. Изменение приоритетов операций. Примеры.
11. Автоматическое преобразование типов в выражении.
12. Оператор присваивания на C++. Примеры.

ЛАБОРАТОРНАЯ РАБОТА 3

ИСПОЛЬЗОВАНИЕ УСЛОВНОГО ОПЕРАТОРА *if* ПРИ СОЗДАНИИ ПРОГРАММ НА ЯЗЫКЕ C++

Цель работы– приобретение и закрепление практических навыков при написании разветвляющихся программ на языке программирования C++ с использованием условного оператора *if*.

Условный оператор *if*

Блок-схема условного оператора *if* представлена в приложении Г. Параметр *условие* представляет собой выражение логического типа, которое может включать в себя следующие операции отношения: $<$, $>$, $==$, $>=$, $<=$, $!=$, $\&\&$, $\|$ (приложение В, табл. В.3). Логическое выражение (*условие*) формирует результат типа *bool*, равный *true* или *false*. Если логическое выражение принимает значение *true*, то выполняется *оператор1*, иначе (*else*)– логическое выражение равно *false*, выполняется *оператор2* (приложение Г, рис. Г.1). Если в операторе *if* на результат логического выражения *false* отсутствует оператор (ветки *else* нет), то осуществляется переход к оператору, следующему за условным (приложение Г, рис. Г.2). Если в условном операторе *if ... else* после вычисления значения выражения в условии должно быть выполнено несколько операторов, то используется *составной оператор* (приложение Г, рис. Г.3).

Порядок выполнения работы

1. В соответствии с номером по журналу выберите индивидуальное задание.
2. Разработайте алгоритм решения задачи.
3. Составьте текст программы. При составлении текста программы придерживайтесь общей структуры программы, приведенной в приложении Б.
4. Создайте проект в интегрированной среде разработки *Microsoft Visual Studio*. (*lab3_фамилия.cpp*)
5. Введите текст программы.

6. Скомпилируйте программу. Если в программе есть ошибки, исправьте их. Если ошибок нет, то появится сообщение об успешной компиляции.

7. Запустите программу на выполнение, проанализируйте результаты работы выполнения программы. Убедитесь в правильности решения задачи.

8. Напишите отчет по лабораторной работе (приложение А), который должен содержать:

- титульный лист;
- цель работы;
- индивидуальное задание;
- алгоритм работы программы;
- текст программы;
- результаты работы программы;
- выводы.

ПРИМЕРЫ РЕШЕНИЯ ЗАДАНИЙ

Задание 3.1

Ввести три целых числа и расположить их в порядке убывания: $x > y > z$. Вычислить значение функции $f(z) = 3,5 + \sin z - 0,47z$. Если полученный результат $f(z) < 0$, то вычислить модуль $|f(z)|$, иначе найти остаток от деления чисел x и y .

I. Выбор метода

Для написания программы используется условный оператор *if*, также для вычисления модуля $|f(z)|$ – функция *fabs(x)* (приложение В, табл. В.4). Для нахождения остатка от деления числа x на y используется оператор *%*.

II. Описание решения задачи на псевдокоде

1. Начало.
2. Ввести x , y и z .
3. Расположить в порядке убывания числа x , y и z .
4. Вычислить значение функции $f(z)$.
5. Вычислить модуль $|f(z)|$, если $f(z) < 0$.
6. Найти остаток от деления x на y , если $f(z) > 0$.
7. Вывести результаты на экран.
8. Конец.

III. Схема алгоритма программы

Блок-схема алгоритма программы представлена на рисунке 3.1.

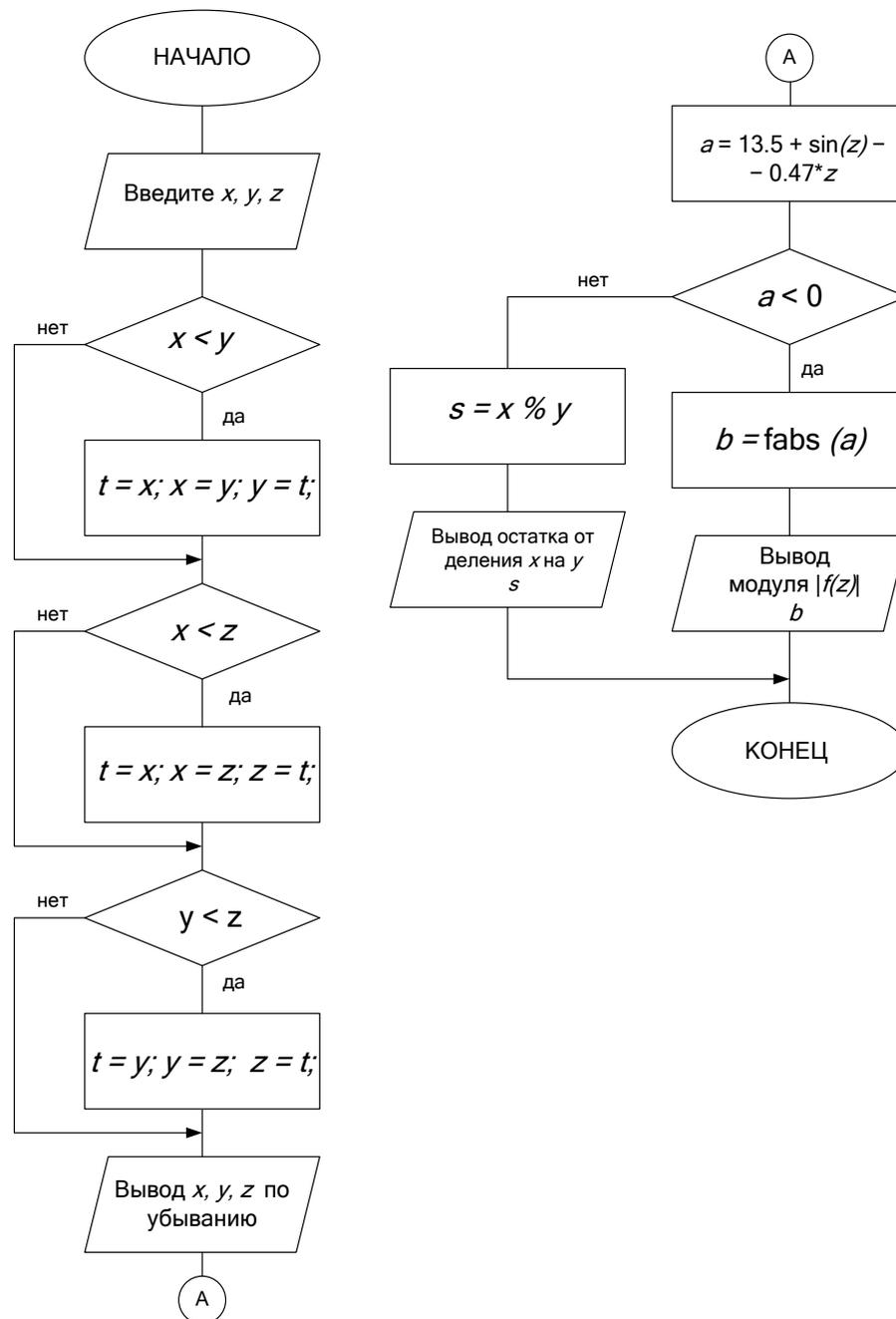


Рисунок 3.1 – Блок-схема алгоритма программы задания 3.1

IV. Разработка текста программы

1. Подключаем в файле *stdafx.h* необходимые для работы программы библиотеки:

#include <iostream> – для работы операторов ввода/вывода.

#include<math.h>– для использования математических функций.

usingnamespacestd;

2. Разработка раздела описания переменных

int x, y, z, t, s;

float a, b;

x, y, z– три целых числа, вводимых с клавиатуры;

s – целая переменная, остаток от деления *x* на *y*;

t–дополнительная переменная;

a – вещественная переменная для вычисления значения $f(z)$;

b – вещественная переменная для округления значения $f(z)$;

3. Разработка тела программы

При вводе исходных данных с клавиатуры для подсказки, что вводить, используем оператор вывода на экран **cout<<** с соответствующим текстом. Затем используем оператор ввода **cin>>** с указанием необходимых переменных. Например, фрагмент диалогового ввода переменных *x, y, z* имеет вид:

cout<<" Введите три числа x, y и z \n";

cin>>x>>y>>z;

Чтобы вывести на экран числа *x, y, z* в порядке убывания необходимо расположить в переменной *x* самое большое значение, а в переменной *z* самое маленькое значение. Для этого нужно сравнить сначала значения переменных *x* и *y*. Если $x < y$, то необходимо поменять значения переменных: в *x* записать значение *y*, а в *y* записать значение *x*. Эта процедура осуществляется с помощью дополнительной переменной *t*:

if (x<y)// сравнение значений переменных

{

t=x;// присвоить переменной *t* значение переменной *x*

x = y;// присвоить переменной *x* значение переменной *y*

y = t;// присвоить переменной *y* значение переменной *t*

}

Если $x > y$, то менять местами значения не нужно, т.к. такое расположение значений переменных удовлетворяет заданному условию (в переменной *x* должно быть большее число).

Далее необходимо сравнить значения переменных *x* и *z*. Если $x < z$, то необходимо поменять значения переменных, если нет, то программа переходит на следующий условный оператор сравнения **if (y<z)**.

```

if (x<z)// сравнение значений переменных
{
  t = x;// присвоить переменной t значение переменной x
  x = z;// присвоить переменной x значение переменной z
  z = t;// присвоить переменной z значение переменной t
}
if (y<z)// сравнение значений переменных
{
  t = y;// присвоить переменной t значение переменной y
  y = z; // присвоить переменной y значение переменной z
  z = t;// присвоить переменной z значение переменной t
}

```

Синтаксис программы

```

#include<iostream>
#include<math.h>
using namespace std;

int main ( )
{
  int x, y, z, t, s;
  float a, b;
  cout<<" Введите три целых числа x, y и z \n";
  cin>>x>>y>>z;
  // расположить числа x, y, z в порядке убывания
  if (x<y)// сравнение значений переменных x и y
  {
    t=x;// присвоить переменной t значение переменной x
    x=y;// присвоить переменной x значение переменной y
    y=t;// присвоить переменной y значение переменной t
  }
  if(x<z)// сравнение значений переменных x и z
  {
    t=x;// присвоить переменной t значение переменной x
    x=z;// присвоить переменной x значение переменной z
    z=t;// присвоить переменной z значение переменной t
  }
}

```

```

if(y<z)// сравнение значений переменных u и z
{
t=y;// присвоить переменной t значение переменной y
y=z;// присвоить переменной y значение переменной z
z=t;// присвоить переменной z значение переменной t
}
cout<<"Вывод значений x, y и z по убыванию"<<"\n";
cout<<x<<" "<<y<<" "<<z<<"\n";
a=13.5+sin (z) - 0.47*z;// вычислить значение f (z)
cout<<"Значение f (z)="<<a<<"\n";
if (a<0)
{
b=fabs(a);// нахождение модуля |f(z)|
cout<<"|f(z)|="<<b<<"\n";
}
else
{
s=x % y;// нахождение остатка от деления x на y
cout<<"Остаток от деления"<<x<<" на "<<y<<" = "<<s<<"\n";
}
return 0;
}

```

4. Отладка и запуск программы

Для отладки программы использован пошаговый режим с помощью клавиш F10. Ниже приведены результаты работы программы.

1-й запуск

Введите три целых двузначных числа x, y и z

13 78 34

Вывод значений x, y и z по убыванию

78 34 13

Значение $f(z) = -1.93$

$|f(z)| = 1.93$

2-й запуск

Введите три целых двузначных числа x, y и z

48 56 12

Вывод значений x, y и z по убыванию

56 48 12

Значение $f(z) = 8.06$

Остаток от деления 56 на 48 = 8.

Задание 3.2

Дана заштрихованная область (рис.3.2) и точка с координатами x и y , вводимые с клавиатуры. Написать программу, определяющую, попадает ли точка в заштрихованную область. Результат вывести в виде текстового сообщения.

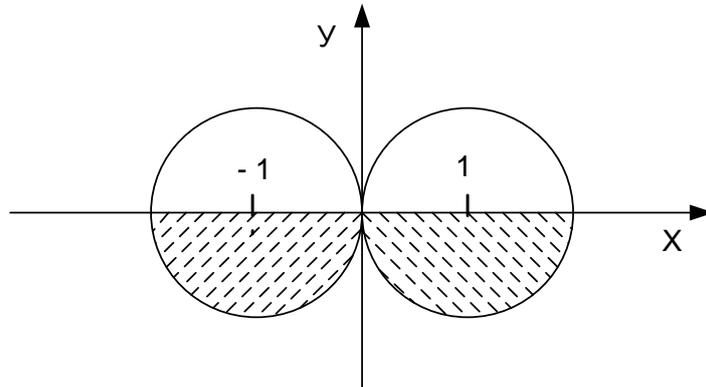


Рисунок 3.2 –Графически заданная область для задания 3.2

I. Выбор метода

Запишем условия попадания точки в область в виде формулы. Точка может попасть в правый полукруг либо в левый полукруг, в обоих случаях значение y должно быть отрицательным. Для того чтобы операция ИЛИ была выполнена раньше, чем операция И, необходимы круглые скобки.

$$y < 0 \text{ И } ((x-1)^2 + y^2 \leq 1 \text{ ИЛИ } (x+1)^2 + y^2 \leq 1)$$

II. Описание решения задачи на псевдокоде

1. Ввести значение аргументов x и y .
2. Определить, принадлежит ли точка заштрихованной области.
3. Вывести результат в виде сообщения на экран.

III. Схема алгоритма программы

Блок-схема алгоритма программы представлена на рисунке 3.3.

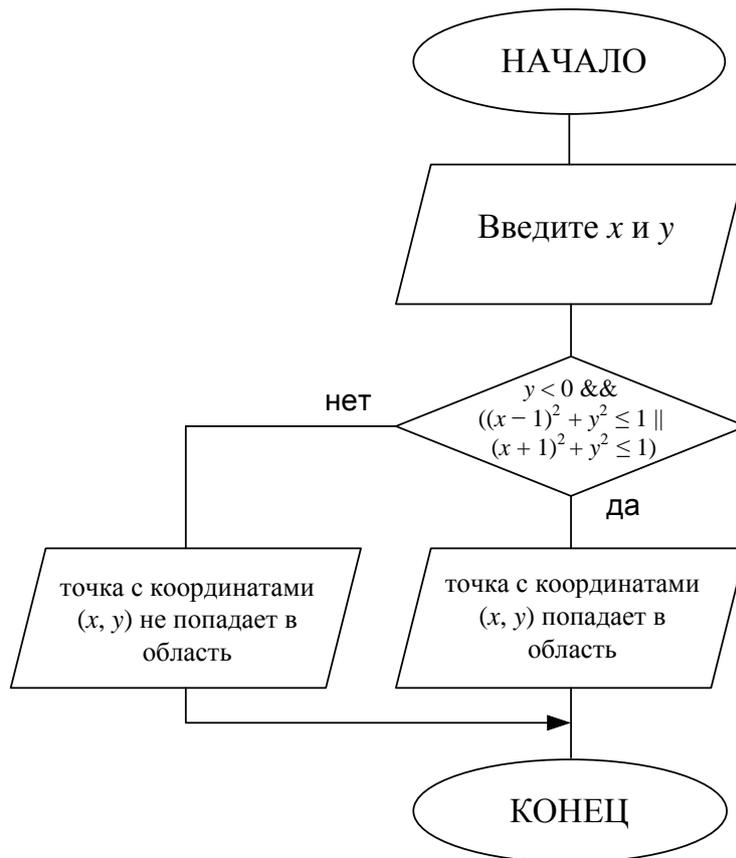


Рисунок 3.3 – Блок-схема алгоритма программы задания 3.2

IV. Разработка текста программы

1. Подключаем в файле *stdafx.h* необходимые для работы программы библиотеки:

```
#include<iostream>– для работы операторов ввода/вывода.  
usingnamespacestd;
```

2. Разработка раздела описания переменных

```
floatx, y;
```

x, y– вещественные переменные, вводимые с клавиатуры координаты.

3. Разработка тела программы

Для ввода исходных данных с клавиатуры используем оператор ввода *cin>>*, а также оператор вывода *cout<<*, с соответствующим сообщением.

```
cout<<" Введите координаты точки x и y\n";  
cin>>x>>y;
```

Условный оператор для определения попадания точки в заштрихованную область имеет вид:

```
if (y < 0 && ((x - 1)*(x - 1) + y * y <= 1 // (x + 1)*(x + 1) + y*y <= 1))
    cout<<" Точка попадает в область\n";
else
    cout<<" Точка не попадает в область\n";
```

Синтаксис программы

```
#include<iostream>
using namespace std;

int main()
{
    float x, y;
    cout<<" Введите координаты точки x и y\n";
    cin>>x>>y;
    // проверка принадлежности точки заштрихованной области
    if (y < 0 && ((x - 1) * (x - 1) + y * y <= 1 // (x + 1)*(x + 1) + y * y <= 1))
        cout<<" Точка попадает в область\n";
    else
        cout<<" Точка не попадает в область\n";
    return 0;
}
```

4. Отладка и запуск программ

Для отладки программы используем клавишу *F7*, убеждаемся в отсутствии ошибок и запускаем программу на исполнение с помощью комбинации клавиш *Ctrl+F5*. Ниже приведены результаты работы программы.

1-й запуск

Введите координаты точки x и y

0,5; 0.2

Точка не попадает в область.

2-й запуск

Введите координаты точки x и y

1.2; -0.5

Точка попадает в область.

1. Индивидуальные задания

1. Вычислить значение $y(x) = \ln x / \log_{10}x$, где x – число, вводимое с клавиатуры. Если $y(x) < 0$, то вычислить $z(x) = x^4 \sin^2 x$, иначе, вычислить значение функции $z(x) = \sqrt{\cos^2(x-5) + 4} \cdot \ln x$.

2. Для заданного года определить значение столетия (например, 1900 год – 19 столетие, 1901 год – 20 столетие).

3. Для заданного числа x и a найти значение уравнения $f(x)$, где

$$f(x) = \begin{cases} 2ax + |a-1|, & \text{при } 0 < a < 2, \\ (1+x)/a, & \text{при } a \leq 0, \\ e^x - a, & \text{при } a \geq 2. \end{cases}$$

Округлить полученное значение $f(x)$ и вывести полученный результат на экран.

4. Найти максимальную цифру в записи трехзначного числа. Определить, является ли данная цифра чётным числом.

5. Ввести целое число. Определить принадлежность числа интервалам $[-100...0]$; $[23...90]$; $[145...158]$. Если число не принадлежит требуемым интервалам – вывести сообщение. Если число попадает в интервал $[23...90]$, то найти остаток от деления введенного числа на 38.

6. Даны отрицательные числа a, b и c . Найти наибольшее из трех чисел и вычислить его куб.

7. Вычислить $y(x)$:

$$y(x) = \begin{cases} \log_{10}(x+3), & \text{при } -1 \leq x \leq 1; \\ x^5 + \sqrt{x^3 + 1}, & \text{при } x > 15; \\ 1/(x-5), & \text{иначе} \end{cases}$$

Округлить полученное значение и вывести результат на экран.

8. Даны произвольные числа a, b, c . Если нельзя построить треугольник с такими длинами сторон, то напечатать 0, иначе напечатать 3,2 или 1 в зависимости от того, равносторонний этот треугольник, равнобедренный или какой-либо иной.

9. Вычислить значение функции:

$$y(x) = \begin{cases} x^3 + \operatorname{tg} x, & \text{при } x > 0,3 \\ \cos x + \arcsin x, & \text{при } x \leq 0,3 \end{cases}$$

10. Определить, попадает ли точка с координатами x, y в окружность радиусом r . Если не попадает, то вычислить радиус окружности, в которую она попадает.

11. Написать программу, которая проверяет, является ли целое число n , введенное с клавиатуры, кратным 5.

12. Определить, в каком квадрате находится точка с координатами x, y и вывести на печать номер квадрата.

13. Найти разность двух наименьших из трёх чисел.

14. Составить программу определения, является ли введенное с клавиатуры целое число n четным двузначным числом.

15. Составить программу проверки, является ли заданное трехзначное число палиндромом.

16. Для целого числа n найти сумму квадратов его цифр. Проверить, является ли полученное число четным.

17. Составить программу определения, пройдет ли шар радиуса r через прямоугольное отверстие со сторонами a и b .

18. Составить программу, которая проверяет, делится ли заданное трехзначное число на каждую из своих цифр.

19. Составить программу, которая определяет можно ли построить треугольник с заданными сторонами a, b, c .

20. Составить программу, которая переменной d присваивает наибольшее из трех чисел, а переменной s наименьшее из трех чисел.

2. Индивидуальные задания

Вычислить и вывести на экран значения функции F , где a, b, c, x – вещественные числа, вводимые с клавиатуры (таблица 3.1).

Таблица 3.1 – Задания к лабораторной работе №3

Вариант 1	Вариант 2
$F = \begin{cases} ax^2 + b & \text{при } x < 0 \text{ и } b \neq 0 \\ \frac{x-a}{x-c} & \text{при } x > 0 \text{ и } b = 0 \\ \frac{x}{c} & \text{в остальных случаях} \end{cases}$	$F = \begin{cases} \frac{1}{ax} - b & \text{при } x + 5 < 0 \text{ и } c = 0 \\ \frac{x-a}{a} & \text{при } x + 5 > 0 \text{ и } c \neq 0 \\ \frac{10x}{c-4} & \text{в остальных случаях} \end{cases}$

Продолжение таблицы 3.1

<p>Вариант 3</p> $F = \begin{cases} ax^2 + bx + c & \text{при } a < 0 \text{ и } c \neq 0 \\ \frac{-a}{x-c} & \text{при } a > 0 \text{ и } c = 0 \\ a(x+c) & \text{в остальных случаях} \end{cases}$	<p>Вариант 4</p> $F = \begin{cases} -ax - c & \text{при } c < 0 \text{ и } x \neq 0 \\ \frac{x-a}{-c} & \text{при } c > 0 \text{ и } x = 0 \\ \frac{bx}{c-a} & \text{в остальных случаях} \end{cases}$
<p>Вариант 5</p> $F = \begin{cases} a - \frac{x}{10+b} & \text{при } x < 0 \text{ и } b \neq 0 \\ \frac{x-a}{x-c} & \text{при } x > 0 \text{ и } b = 0 \\ 3x + \frac{2}{c} & \text{в остальных случаях} \end{cases}$	<p>Вариант 6</p> $F = \begin{cases} ax^2 + b^2x & \text{при } c < 0 \text{ и } b \neq 0 \\ \frac{x+a}{x+c} & \text{при } c > 0 \text{ и } b = 0 \\ \frac{x}{c} & \text{в остальных случаях} \end{cases}$
<p>Вариант 7</p> $F = \begin{cases} -ax^2 - b & \text{при } x < 5 \text{ и } c \neq 0 \\ \frac{x-a}{x} & \text{при } x > 5 \text{ и } c = 0 \\ \frac{-x}{c} & \text{в остальных случаях} \end{cases}$	<p>Вариант 8</p> $F = \begin{cases} -ax^2 & \text{при } c < 0 \text{ и } a \neq 0 \\ \frac{a-x}{cx} & \text{при } c > 0 \text{ и } a = 0 \\ \frac{x}{c} & \text{в остальных случаях} \end{cases}$
<p>Вариант 9</p> $F = \begin{cases} ax^2 + b^2x & \text{при } a < 0 \text{ и } x \neq 0 \\ x - \frac{a}{x-c} & \text{при } a > 0 \text{ и } x = 0 \\ 1 + \frac{x}{c} & \text{в остальных случаях} \end{cases}$	<p>Вариант 10</p> $F = \begin{cases} ax^2 - bx + c & \text{при } x < 3 \text{ и } b \neq 0 \\ \frac{x-a}{x-c} & \text{при } x > 3 \text{ и } b = 0 \\ \frac{x}{c} & \text{в остальных случаях} \end{cases}$
<p>Вариант 11</p> $F = \begin{cases} ax^2 + \frac{b}{c} & \text{при } x < 1 \text{ и } c \neq 0 \\ \frac{x-a}{(x-c)^2} & \text{при } x > 1 \text{ и } c = 0 \\ \frac{x^2}{c^2} & \text{в остальных случаях} \end{cases}$	<p>Вариант 12</p> $F = \begin{cases} ax^3 + b^2 + c & \text{при } x < 0.6 \text{ и } b + c \neq 0 \\ \frac{x-a}{x-c} & \text{при } x > 0.6 \text{ и } b + c = 0 \\ \frac{x}{c} + \frac{x}{a} & \text{в остальных случаях} \end{cases}$
<p>Вариант 13</p> $F = \begin{cases} ax^2 + b & \text{при } x-1 < 0 \text{ и } b-x \neq 0 \\ \frac{x-a}{x} & \text{при } x-1 > 0 \text{ и } b-x = 0 \\ \frac{x}{c} & \text{в остальных случаях} \end{cases}$	<p>Вариант 14</p> $F = \begin{cases} -ax^3 - b & \text{при } x+c < 0 \text{ и } a \neq 0 \\ \frac{x-a}{x-c} & \text{при } x+c > 0 \text{ и } a = 0 \\ \frac{x}{c} + \frac{c}{x} & \text{в остальных случаях} \end{cases}$

Продолжение таблицы 3.1

<p>Вариант 15</p> $F = \begin{cases} -ax^2 + b & \text{при } x < 0 \text{ и } b \neq 0 \\ \frac{x}{x-c} + 5.5 & \text{при } x > 0 \text{ и } b = 0 \\ \frac{x}{-c} & \text{в остальных случаях} \end{cases}$	<p>Вариант 16</p> $F = \begin{cases} a(x+c)^2 - b & \text{при } x = 0 \text{ и } b \neq 0 \\ \frac{x-a}{-c} & \text{при } x = 0 \text{ и } b = 0 \\ a + \frac{x}{c} & \text{в остальных случаях} \end{cases}$
<p>Вариант 17</p> $F = \begin{cases} ax^2 - cx + b & \text{при } x + 10 < 0 \text{ и } b \neq 0 \\ \frac{x-a}{x-c} & \text{при } x + 10 > 0 \text{ и } b = 0 \\ \frac{-x}{a-c} & \text{в остальных случаях} \end{cases}$	<p>Вариант 18</p> $F = \begin{cases} ax^3 + bx^2 & \text{при } x < 0 \text{ и } b \neq 0 \\ \frac{x-a}{x-c} & \text{при } x > 0 \text{ и } b = 0 \\ \frac{x+5}{c(x-10)} & \text{в остальных случаях} \end{cases}$
<p>Вариант 19</p> $F = \begin{cases} a(x+7)^2 - b & \text{при } x < 5 \text{ и } b \neq 0 \\ \frac{x-cd}{ax} & \text{при } x > 5 \text{ и } b = 0 \\ \frac{x}{c} & \text{в остальных случаях} \end{cases}$	<p>Вариант 20</p> $F = \begin{cases} -\frac{2x-c}{cx-a} & \text{при } x < 0 \text{ и } b \neq 0 \\ \frac{x-a}{x-c} & \text{при } x > 0 \text{ и } b = 0 \\ -\frac{x}{c} + \frac{-b}{2x} & \text{в остальных случаях} \end{cases}$

Контрольные вопросы

1. Составить программу поиска наибольшего из трёх чисел.
2. Составить программу, которая запрашивает время суток и выводит соответствующее приветствие.
3. Составить программу проверки для заданного числа –признак делимости на 5.
4. У наибольшего из чисел n и m найти цифру младшего разряда (единиц) и остаток от его деления на 3.
5. Составить программу определения, пройдет ли шар радиуса r в квадратное отверстие со стороной a .
6. Составить программу определения, пройдет ли куб с ребром a в круглое отверстие радиуса r .
7. Составить программу проверки, является ли год високосным.
8. Составить программу, которая вводит 2 числа и присваивает переменной z наибольшее из них, если первое число отрицательное, в противном случае z присваивает наименьшее.
9. Составить программу поиска максимального из 4-х чисел.

ЛАБОРАТОРНАЯ РАБОТА 4

ИСПОЛЬЗОВАНИЕ ОПЕРАТОРА-ПЕРЕКЛЮЧАТЕЛЯ *switch* ПРИ СОЗДАНИИ ПРОГРАММ НА ЯЗЫКЕ C++

Цель работы– приобретение и закрепление практических навыков при написании разветвляющихся программ на языке программирования C++ с использованием оператора *switch*.

Оператор-переключатель *switch*

Оператор-переключатель *switch* является модификатором условного оператора и позволяет осуществлять многовариантный выбор, заменяя группу вложенных операторов *if-else*. Схема алгоритма, которая реализуется оператором, представлена на рисунке Г.4 в приложении Г. Параметр *<выражение>*– это выражение перечисляемого типа: целого, символьного, логического или пользовательского. Параметр *знач 1, знач 2 ... знач n*– *const* значения, которые будут сравниваться со значением выражения. При совпадении этих значений будет выполняться соответствующий оператор. Если таких совпадений не обнаружено, то будет выполняться *оператор m* в секции *default*. Значение *default* может отсутствовать, в этом случае произойдет переход к следующему оператору. Оператор *break*– служит для прекращения выполнения оператора-переключателя.

Порядок выполнения работы

1. В соответствии с номером по журналу выберите индивидуальное задание.
2. Разработайте алгоритм решения задачи.
3. Составьте текст программы. При составлении текста программы придерживайтесь общей структуры программы, приведенной в приложении Б.
4. Создайте проект в интегрированной среде разработки Microsoft VisualStudio. (*lab4_фамилия.cpp*)
5. Введите текст программы.

6. Скомпилируйте программу. Если в программе есть ошибки, исправьте их. Если ошибок нет, то появится сообщение об успешной компиляции.

7. Запустите программу на выполнение, проанализируйте результаты работы выполнения программы. Убедитесь в правильности решения задачи.

8. Напишите отчет по лабораторной работе (приложение А), который должен содержать:

- титульный лист;
- цель работы;
- индивидуальное задание;
- алгоритм работы программы;
- текст программы;
- результаты работы программы;
- выводы.

ПРИМЕРЫ РЕШЕНИЯ ЗАДАНИЙ

Задание 4.1

Написать программу, определяющую, какая из курсорных клавиш была нажата.

I. Выбор метода

Для определения, какая из курсорных клавиш нажата пользователем, воспользуемся функцией `_getwch()`, которая возвращает код нажатой клавиши. В случае нажатия функциональных или курсорных клавиш эта функция возвращает 0 либо 0xEO (в зависимости от компилятора), а её повторный вызов позволяет получить расширенный код клавиши.

II. Описание решения задачи на псевдокоде

1. Начало.
2. Определение нажатой клавиши курсора.
3. Вывод результата на экран.
4. Конец.

III. Схема алгоритма программы

Блок-схема алгоритма программы представлена на рисунке 4.1.

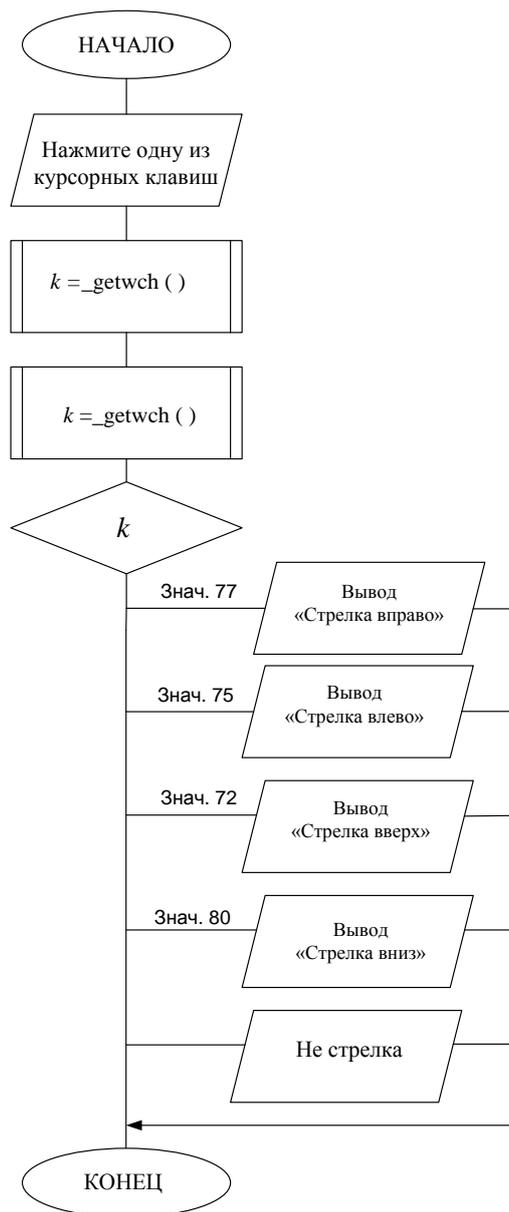


Рисунок 4.1 – Блок-схема алгоритма программы задания4.1

IV. Разработка текста программы

1. Подключаем в файле *stdafx.h* необходимые для работы программы библиотеки:

#include<iostream>– для работы операторов ввода/вывода.

#include<conio.h>– для работы функции ***_getch ()***.

using namespace std;

2. Разработка раздела описания переменных

intk;

k – переменная целого типа, определяющая код нажатой клавиши.

3. Разработка тела программы

Вывод на экран названия нажатой клавиши осуществляется с помощью оператора выбора *switch* (). Выражение, стоящее в скобках после ключевого слова *switch*, а также константные выражения в *case* должны быть целочисленного типа (они неявно приводятся к типу выражения в скобках).

Синтаксиспрограммы

```
#include<iostream>  
#include<conio.h>  
usingnamespacestd;  
intmain ( )  
{  
intk;  
cout<<" Нажмите одну из курсорных клавиш \n";//вывод на экран текста  
k = _getwch ();//вызов функции определяющей код нажатой клавиши  
k = _getwch();// повторный вызов функции  
switch (k)// оператор выбора  
{  
case77 : cout<<" Стрелка вправо\n ";  
break;  
case 75 : cout<<" Стрелка влево\n";  
break;  
case 72 : cout<<" Стрелкавверх\n ";  
break;  
case 80 : cout<<" Стрелкавниз\n";  
break;  
default : cout<<" Нестрелка\n";  
}  
return 0;  
}
```

4 Отладка и запуск программы

Для запуска программы на выполнение используем комбинацию клавиш *Ctrl+F5*. Результаты работы программы приведены ниже:

```
Нажмите одну из курсорных клавиш  
↑  
Стрелка вверх  
Нажмите одну из курсорных клавиш  
→  
Стрелка вправо
```

1. Индивидуальные задания

1. Если последняя цифра трехзначного числа 2, то найти разность цифр данного числа, иначе – напечатать название первой цифры трехзначного числа.

2. Вычислить $y = 1 + x / x^2$ для любого значения x . Округлить полученное значение к ближайшему целому. Напечатать название старшей цифры полученного числа.

3. Определить количество понедельников в году, приходящихся на 13-е число, считая, что год не високосный и 1 января приходится на понедельник.

4. Даны 3 десятичные цифры. Ввести на печать название меньшей из этих цифр и напечатать ее название.

5. Если первая цифра четырехзначного числа 1, то найти сумму цифр данного числа, если 2 – найти разность двух последних цифр числа, если первая цифра 8, то найти произведение двух последних цифр числа, иначе, вывести на печать название первой цифры числа.

6. Найти максимальную цифру в записи трехзначного числа и вывести на печать ее название.

7. Вывести на печать наименьшую цифру трехзначного числа и напечатать ее название. Определить, является ли полученная цифра четным числом.

8. Для натурального числа k напечатать фразу «мы нашли k грибов в лесу», согласовав окончание слова «гриб» с числом k .

9. Написать программу, которая запрашивает у пользователя номер месяца рождения, затем выводит название месяца или сообщение об ошибке.

10. Составить программу, которая для любого натурального n меньшего 100 дает наименование «год», «года», «лет» считая, что n возраст человека.

11. Найти минимальную цифру трёхзначного числа. Вывести на экран её название. Определить четность этого числа.

12. По дате определить день недели, считая, что год не високосный и 1 января приходится на понедельник.

13. Если последняя цифра трехзначного числа 5, то найти сумму цифр данного числа, если 3 – разность цифр данного числа, иначе – напечатать название второй цифры трехзначного числа.

14. Если первая цифра четырёхзначного числа 2, то найти разность двух последних цифр числа, если первая цифра больше 2 и меньше 8, то найти произведение двух последних цифр числа, иначе, вывести на печать название первой цифры числа.

15. Написать программу перевода оценок, полученных студентами на экзамене по информатике в болонскую систему.

16. Для вводимого с клавиатуры целого числа n напечатать фразу «мне n лет», учитывая при этом, что при некоторых значениях n слово «лет» надо заменить на слово «года» или «год».

17. Ввести с клавиатуры целое четырёхзначное число. Вывести на экран название второй цифры числа.

18. Написать программу, которая выводит на экран название максимальной цифры трёхзначного числа, введенного с клавиатуры.

19. Вычислить $y = (4 + x) / x^2$ для любого значения x . Округлить полученное значение к ближайшему целому. Напечатать название старшей цифры полученного числа.

20. По дате определить день недели, считая, что год не високосный и 1 января приходится на понедельник.

Контрольные вопросы

1. Оператор-переключатель *switch*. Синтаксис.
2. Оператор-переключатель *switch*. Блок-схема.
3. Написать программу, которая выводит на экран название максимальной цифры трёхзначного числа, введенного с клавиатуры.
4. Для натурального числа k напечатать фразу «мы нашли k грибов в лесу», согласовав окончание слова «гриб» с числом k .
5. Назначение функции `_getch()`.

СПИСОК ЛИТЕРАТУРЫ

1. Светозарова Г.И. Практикум по программированию на алгоритмических языках / Г.И. Светозарова, Е.В. Сигитов, А. В. Козловский. – Минск: «Наука», 1980. – 317 с.
2. Дейтел Х.М. Как программировать на С++ / Х.М. Дейтел, П.Дж. Дейтел. – М.: ЗАО «Издательство БИНОМ», 2000. – 1024 с.
3. СтрауструпБьерн. Язык программирования С++ / БьернСтрауструп. – М.: Бином, 2002. – 1098с.
4. ШилдтГерберт. Справочник программиста по С/С++: пер. с англ / Герберт Шилдт. –М.: Издательский дом «Вильямс», 2003. – 432 с.
5. Павловская Т.А. Структурное программирование С/С++. Практикум / Т.А. Павловская, Ю.А. Щупак. – Спб.: Питер, 2007. – 239 с.
6. Подбельский В.В. Язык Си+: учеб. пособие / В. В. Подбельский. – М.: БИНОМ, 1995. – 400 с.

ПРИЛОЖЕНИЯ

ПРИЛОЖЕНИЕ А

Образец оформления отчета по лабораторной работе

Страница 1

Страница 2

Министерство образования и науки
Украины

Цель работы:
Задание:

Национальный технический университет
«Харьковский политехнический институт»

Информатика

Кафедра «.....»

Лабораторная работа № ...
Тема

Выполнил
Ст гр.

Ф.И.О.
Проверил
Ф.И.О.

Харьков-20..

Страница 3

Блок-схема программы

Страница 4

Текст программы

Страница 5

Результаты работы программы

Страница 6

Выводы

ПРИЛОЖЕНИЕ Б
Структура программы. Типы данных

```

{область директив препроцессора (include, define);
{ описание пользовательских типов данных;
{неполные объявления функция;
{описание глобальных переменных;
{объявление функции 1;
{объявление функции 2;
.....
{объявление функции main();

```

Таблица Б.1 – Стандартные типы данных языка программирования C++

№	Тип	Размер	Диапазон	Знако- вый
1	<i>int</i>	4 байта (для 32-разрядных ОС)	-2147483648..2147483647 ($-2^{31}..2^{31}-1$)	+
2	<i>short</i>	2 байта	-32768...32767 ($-2^{15}..2^{15}-1$)	+
3	<i>long</i>	4 байта	-2147483647..2147483647 ($-2^{31}..2^{31}-1$)	+
4	<i>unsignedint</i>	4 байта (для 32-разрядных ОС)	0...4294967297 ($0..2^{32}-1$)	-
5	<i>unsignedshort</i>	2 байта	0...65535 ($0..2^{15}-1$)	-
6	<i>unsignedlong</i>	4 байта	0...4294967297 ($0..2^{32}-1$)	-
7	<i>float</i>	4 байта	$1,0 \cdot 10^{-38} \dots 1,0 \cdot 10^{38}$	+
8	<i>double</i>	8байта	$1,0 \cdot 10^{-308} \dots 1,0 \cdot 10^{308}$	+
9	<i>char</i>	1 байт	256 символов	-
10	<i>bool</i>	1 байт	true, false	-

ПРИЛОЖЕНИЕ В

Арифметические и логические операторы

Таблица В.1 – Арифметические операторы

<i>Оператор</i>	<i>Действие</i>
-	Вычитание
+	Сложение
*	Умножение
/	Деление
%	Остаток от деления
--	Декремент
++	Инкремент

Таблица В.2 – Приоритеты арифметических операторов

<i>Приоритет</i>	<i>Операторы</i>
Наивысший	++-- * /%
Низший	+-

Таблица В.3 – Операторы отношений и логические операторы

<i>Оператор</i>	<i>Значение</i>
>	Больше
>=	Больше или равно
<	Меньше
<=	Меньше или равно
==	Равно
!=	Не равно
&&	И
	ИЛИ
!	НЕ

Таблица В.4 – Основные функции библиотеки *math.h*

№	Имя функции	Возвращаемое значение	Тип аргумента	Тип результата <i>a</i>
1	$\text{acos}(x)$	Функция $\arccos x$ в радианах	Вещественное значение x	<i>double</i>
2	$\text{asin}(x)$	Функция $\arcsin x$ в радианах	Вещественное значение x	<i>double</i>
3	$\text{atan}(x)$	Функция $\text{arctg } x$ в радианах	Вещественное значение x	<i>double</i>
4	$\text{atan2}(x, y)$	Функция $\text{arctg}(x/y)$ в радианах	Вещественные значения x, y	<i>double</i>
5	$\text{cos}(x)$	Функция $\cos x$	Вещественное значение x	<i>double</i>
6	$\text{sin}(x)$	Функция $\sin x$	Вещественное значение x	<i>double</i>
7	$\text{tan}(x)$	Функция $\text{tg } x$	Вещественное значение x	<i>double</i>
8	$\text{exp}(x)$	Функция e^x	Вещественное значение x	<i>double</i>
9	$\text{abs}(x)$	Абсолютное значение (модуль) аргумента $ x $	Целое значение x	<i>int</i>
10	$\text{fabs}(x)$	Абсолютное значение (модуль) аргумента $ x $	Вещественное значение x	<i>double</i>
11	$\text{labs}(x)$	Абсолютное значение (модуль) аргумента $ x $	Длинное целое значение x	<i>long</i>
12	$\text{log}(x)$	Натуральный логарифм аргумента $\ln x$	Вещественное значение x	<i>double</i>
13	$\text{log10}(x)$	Десятичный логарифм аргумента $\log_{10} x$	Вещественное значение x	<i>double</i>
14	$\text{pow}(x, y)$	Вычисление x^y	Вещественные значения x, y	<i>double</i>
15	$\text{sqrt}(x)$	Квадратный корень аргумента \sqrt{x}	Вещественное значение x	<i>double</i>

ПРИЛОЖЕНИЕ Г

Операторы ветвления *if* и *switch*

Условный оператор *if ... else*

```
if(условие)  
    оператор1;  
else  
    оператор 2;
```

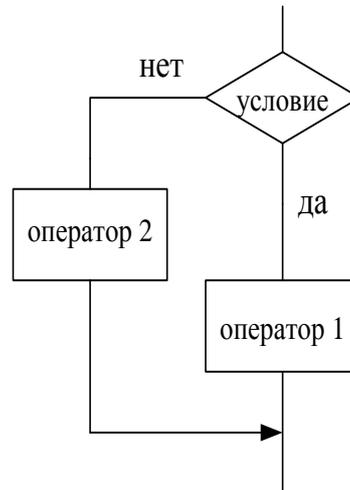


Рисунок Г.1 – Блок-схема алгоритма
условного оператора *if ... else*

Условный оператор *if*

```
if(условие)  
    оператор1;  
    оператор2;
```

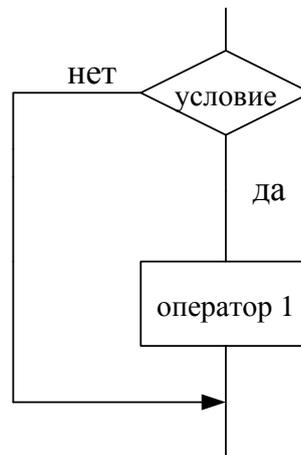


Рисунок Г.2 – Блок-схема алгоритма
условного оператора *if*

Продолжение приложения Г
Составной условный оператор *if ... else*

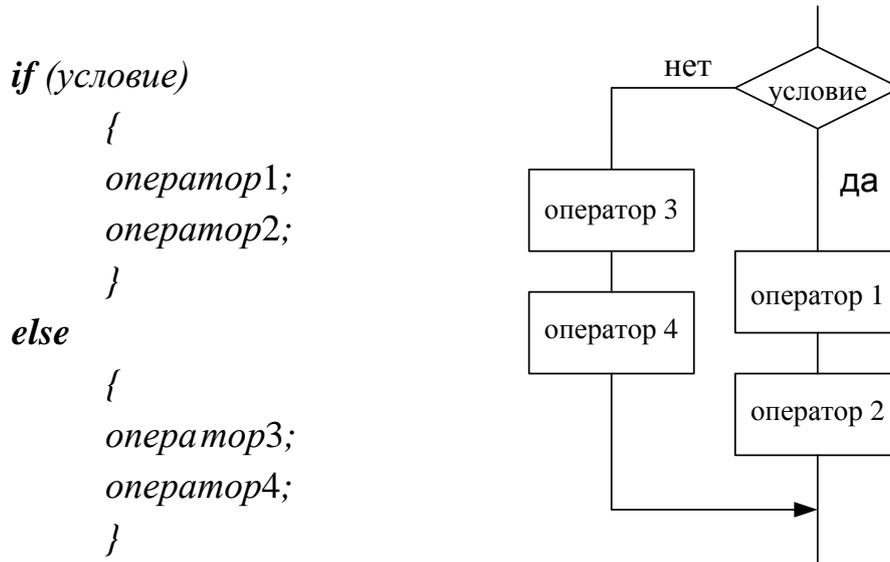


Рисунок Г.3 – Блок-схема алгоритма
 составного условного оператора *if ... else*

Оператор-переключатель *switch*

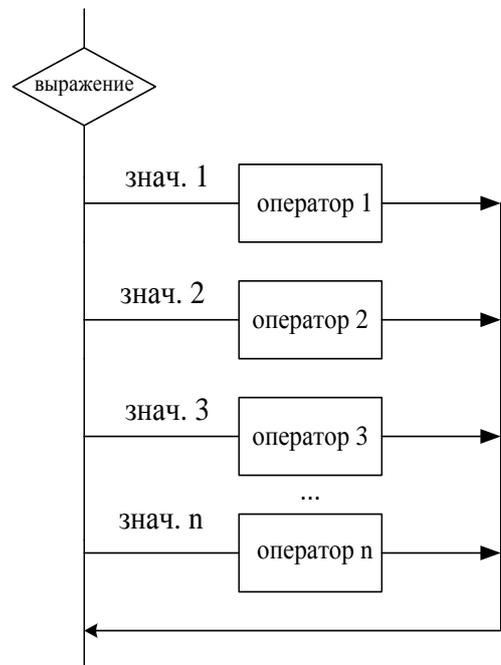
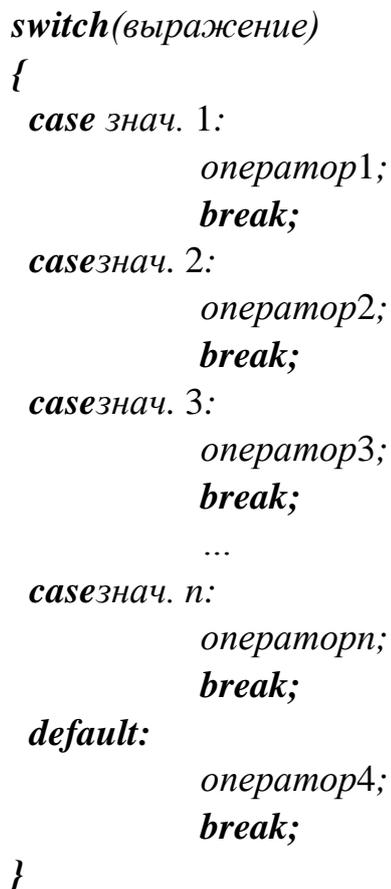


Рисунок Г.4 – Блок-схема алгоритма
 оператора-переключателя *switch*

СОДЕРЖАНИЕ

Вступление.....	3
Лабораторная работа1. Знакомство с интегрированной средой разработки <i>Microsoft Visual Studio</i>	4
Лабораторная работа2. Разработка линейных программ на языке C++	15
Лабораторная работа 3.Использование условного оператора <i>if</i> при создании программ на языке C++	26
Лабораторная работа 4. Использование оператора-переключателя <i>switch</i> при создании программ на языке C++... ..	39
Список литературы	45
Приложение А. Образец оформления отчета.....	46
Приложения Б. Структура программы. Типы данных	47
Приложение В. Арифметические и логические операторы	48
Приложение Г. Операторы ветвления <i>if</i> и <i>switch</i>	50