МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«ДАГЕСТАНСКИЙ ГОСУДАСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Учебно-методические указания

к выполнению лабораторных работ №1-№8

по дисциплине «Языки и методы программирования» для обучающихся по направлению подготовки бакалавров 01.03.02-«Прикладная математика и информатика».

Махачкала 2022

УДК 681.3.06(072)
Методические указания к выполнению лабораторных работ №1-№8
по дисциплине «Языки и методы программирования» для студентов направление подготовки бакалавров 01.03.02-«Прикладная математика и информатика».–Махачкала: ИПЦ ДГТУ, 2022.-34с.

Методические указания подготовлены в соответствии с Государственным образовательным стандартом высшего образования.

Указания содержат описания лабораторных работ по темам:

Каждая лабораторная работа содержит теоретический материал и практическую часть.

Лабораторная работа №1. Интегрированная среда разработки Dev-C++.

Лабораторная работа №2. Арифметические и логические операции в языке C++.

Лабораторная работа №3. Явное и неявное преобразование типов.

Лабораторная работа №4. Программирование алгоритмов разветвляющейся структуры.

Лабораторная работа №5. Программирование алгоритмов разветвляющейся структуры. Оператор switch и условный (тернарный) оператор ? :.

Лабораторная работа № 6-8. Программирование циклических алгоритмов.

Составители: ст. преподаватель кафедры «Прикладной математики и информатики» Алиосманова О.А.

Рецензент:
Зав. кафедрой «ПОВТ и АС» ДГТУ, к.э.н., доцент Т.Г. Айгумов.
Директор НИИ Региональных проблем информатизации ДГУНХ, профессор кафедры «ИТи ИБ», д.э.н. С.Э. Савзиханова.

Печатается согласно постановлению
Ученого совета Дагестанского государственного технического университета
от _____2022 г.

Оглавление

Лабораторная работа №1. Интегрированная среда разработки Dev-C++. 5
Индивидуальные задания 14
Контрольные вопросы 14
Лабораторная работа №2. Арифметические и логические операции в языке C++......15
Индивидуальные задания 18
Контрольные вопросы 19
Лабораторная работа №3. Явное и неявное преобразование типов. 19
Задания 22
Лабораторная работа №4. Программирование алгоритмов разветвляющейся структуры. 23
Индивидуальные задания 24
Контрольные вопросы 26
Лабораторная работа №5. Программирование алгоритмов разветвляющейся структуры. Оператор switch и условный (тернарный) оператор ? : 26
Индивидуальные задания 29
Контрольные вопросы 30
Лабораторная работа № 6-8. Программирование циклических алгоритмов......31
Индивидуальные задания 34
Контрольные вопросы 35
Литература 35

Лабораторная работа №1. Интегрированная среда разработки Dev-C++.

***Цель работы**: знакомство с интегрированной средой разработки Dev-C++, предназначенной для разработки программ на языке C++.*

Краткие теоретические сведения

Язык С, созданный Денисом Ритчи в начале 70-х годов в Bell Laboratory американской корпорации AT&T, является одним из универсальных языков программирования. Язык С считается языком системного программирования, хотя он удобен и для написания прикладных программ, С является языком высокого уровня , а также обладает набором низкоуровневых средств, обеспечивающих доступ к аппаратной части компьютера. Широкое распространение С привело ко многим вариациям языка, C++ это расширение языка С. C++ – это гибридный язык, который предоставляет возможность программировать и в стиле С, в объектно-ориентированном стиле, и в обоих стилях сразу, он разработан Бьерном Страуструпом в начале 80-х годов в Bell Laboratories.

Как правило, чтобы выполнить программу на C/C++ необходимо пройти через 6 этапов: редактирование исходного текста программы, препроцессорную (то есть предварительную) обработку, компиляцию, компоновку, загрузку и выполнение.

1. Первый этап представляет создание и редактирование файла с исходным текстом программы. Файл исходного текста программы на языке С обычно имеет расширение «*.c», на языке C++ – «*.cpp». Это обычный текстовый файл, в который записывают текст программы в любом текстовом редакторе, например в Блокноте.
2. На втором этапе компилятор начинает препроцессорную обработку текста программы прежде чем ее компилировать (перевести программу в машинный код).

Препроцессорная обработка – это поиск так называемых «директив компиляции» или «директив препроцессора», которые указывают, какие преобразования исходного текста программы нужно выполнить перед его компиляцией. Обычно это включение других текстовых файлов (заголовочных файлов) в файл, который подлежит компиляции.

3. Третий этап – это компиляция: перевод исходного текста программы в машинные коды и создание так называемого объектного файла с расширением «*.o» (несмотря на то, что в этом файле уже записан машинный код, объектный файл еще нельзя запускать на компьютере, потому что в нем не хватает стандартных функций (например, для ввода и вывода данных)).
4. Четвертый этап – компоновка (линкинг- связывание объектных файлов) . Компоновщик подключает стандартные функции, хранящиеся в библиотеках (в Dev C++ они имеют расширение «*.a»). В результате получается один файл с расширением *.exe, который и представляет собой готовую программу.

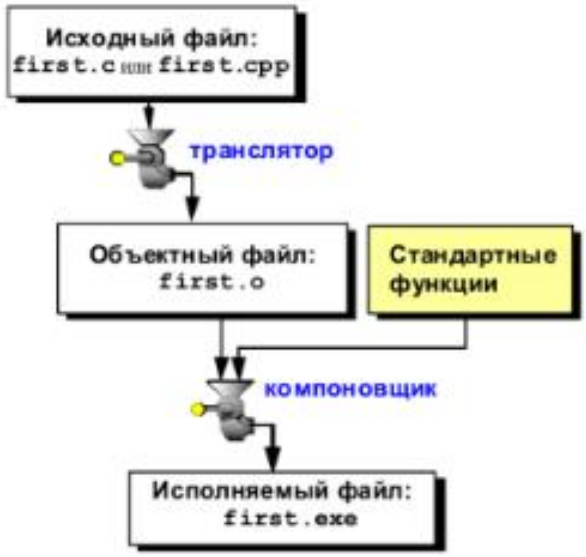


Рис.1 Этапы создания программы.

5. Пятый этап - загрузка. Перед выполнением программа должна быть размещена в памяти. Это делается с помощью загрузчика (входит в состав операционной системы), который забирает загрузочный модуль программы с диска и перемещает его в память.

6. Шестой этап - это выполнение. Каждый из названных выше этапов может заканчиваться ошибкой или неудачей из-за ошибки. Тогда программист должен вернуться к редактированию и исправлению исходного текста программы, предварительно его хорошо проанализировав. Затем снова пройти через все этапы работы с исходным текстом программы до получения работающего без ошибок загрузочного модуля.

Общие сведения об интегрированной среде разработки Dev-C++.

Для выполнения потребуется специальное программное обеспечение. Хотя вы можете использовать отдельные программы на каждом из этих этапов, но один пакет программного обеспечения ("IDE" от англ. "Integrated Development Environment") объединяет в себе все эти программы. Обычно с IDE вы получаете редактор кода с нумерацией строк и подсветкой синтаксиса, а также компилятор и линкер. А когда вам нужно будет провести отладку программы, вы сможете использовать встроенный отладчик. Кроме того, IDE объединяет и ряд других полезных возможностей: комплексная помощь, дополнение кода, в некоторых случаях еще и система контроля версий.

Интегрированная среда разработки – IDE Dev-C++ для языков C++ включает в себя следующие основные компоненты:

- специализированный текстовый редактор, в котором можно набирать и редактировать исходные программы;
- компилятор исходной программы в объектный код (IDE поставляется с компилятором Mingw, однако его можно настроить и на некоторые другие компиляторы);
- компоновщик для создания загрузочного модуля в различных целевых средах:(консольных программ в среде MSDOS, программ Windowsc графическим интерфейсом, файлов библиотек динамических связей (DLL), а также статических библиотек;
- средства запуска программ и вывода результатов в различных целевых средах;
- отладчики для устранения логических ошибок и ошибок при выполнении программ.

Эта IDE имеет набор меню, которые дают возможность именовать и сохранять файл исходного кода, а также компилировать, компоновать, выполнять и отлаживать программы, не покидая окно IDE. Если компилятор обнаруживает ошибки, выполняется возврат в программу редактора (при этом указываются ошибочные строки программы и соответствующие сообщения об ошибках). Если программа содержит несколько исходных модулей, в IDE создается проект, в котором указываются имена файлов, содержащих исходные модули. Это дает возможность не только компоновать все объектные модули проекта в единый загрузочный модуль, но и автоматически перекомпилировать только те модули проекта, в которых были сделаны изменения. Окно редактора Dev-C++ имеет следующий вид:

Рис.2 Окно редактора Dev-C++.

Панели инструментов можно добавлять или удалять с помощью контекстного меню, которое вызывается, если щелкнуть правой кнопкой мыши на свободном месте панелей инструментов.

Настройка Dev-C++

Перед работой с Dev-C++ должна быть выполнена настройка IDE.

Прежде всего, необходимо указать язык интерфейса для команд меню, кнопок панелей инструментов и выводимых сообщений (первоначально языком интерфейса является английский язык). Для изменения интерфейса в меню **Tools** (Сервис) надо выбрать команду **Environment Options** (Параметры среды) и во вкладке **Interface** (Интерфейс) диалогового окна этой команды из раскрывающегося списка поля **Language** (Язык) задать язык интерфейса: **Russian** (Русский).

Во вкладке в раскрывающемся списке **Тема** устанавливается также вид (тема) графических элементов окна IDE. Можно установить одну из трех тем:**New Look**,**Gnome** или **Blue** (в дальнейшем предполагается, что установлена тема **New Look**).

Шрифт и размер шрифта, используемые в редакторе, устанавливаются во вкладке **Вид** диалогового окна команды **Параметры редактора** меню **Сервис**. В раскрывающихся списках **Шрифт** областях **Шрифт редактора** и **Линейка** этой вкладки рекомендуется установить моноширинный шрифт **Courier New** и в раскрывающихся списках **Размер**– размер шрифта (рекомендуемый размер –**12**). Линейка – это область серого цвета слева от текста программы. Ширину этой области можно изменить с помощью поля **Ширина** в области **Линейка**. При включении переключателя **Номера строк** в области **Линейка** на линейке будут выводиться номера строк программы.

Цветовое оформление различных элементов программы: ключевых слов, комментариев к программе, директив препроцессора и т.д. можно установить с помощью полей вкладки **Синтаксис** диалогового окна **Параметры редактора** меню **Сервис**.

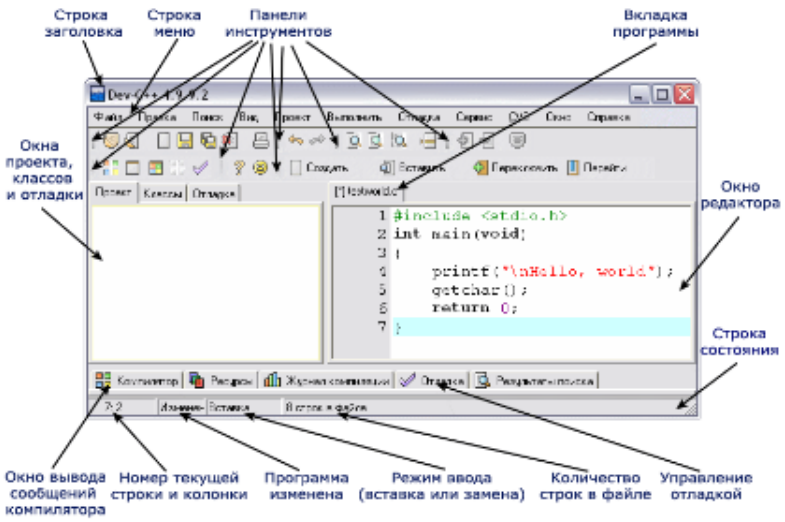
Создание и редактирование простых программ в Dev-C++

Создать простую программу, состоящую из одного модуля, в Dev-C++ можно одним из следующих способов:

- с помощью команды **Исходный файл** подменю команды **Создать** в меню**Файл;**
- нажав кнопку Исходный файл( )на панели инструментов;
- нажав клавиши **Ctrl**+**N**.

При этом в области редактора появляется новая вкладка с пустым содержимым – Безымянный1, Безымянный2 и т.д.

Набор и редактирование текста программы в Dev-C++ выполняется так же, как и в приложении Блокнот. Для вырезания, копирования и вставки фрагментов программы можно использовать команды **Вырезать**, **Копировать** и **Вставить** меню **Правка**, либо клавиши



Ctrl+**X**, **Ctrl**+**C Ctrl**+**V**. Если текст программы был изменен, но изменения не сохранены, перед именем файла в названии вкладки выводятся символы "**[*]**".

При сохранении вновь созданного файла (с помощью команды **Сохранить** как менюФайл) рекомендуется задавать имя файла латинскими буквами. При сохранении файла в окне **Сохранить** файл следует в раскрывающемся списке **Тип файла** задать тип **(*.cpp)**.

Повторно сохранить отредактированный файл можно с помощью одной из следующих операций:

- выполнить команды **Сохранить** меню *Файл;*
- нажать кнопку **Сохранить** на панели инструментов;
- нажать клавиши **Ctrl**+**S**.

Открытие файла с программой выполняется одним из следующих способов:

- с помощью команды **Открыть** проект или файл меню **Файл**;
- при нажатии кнопки **Открыть** проект или файл на панели инструментов;
- при нажатии клавиш **Ctrl**+**O**.

Закрыть файл с программой можно:

- с помощью команды **Закрыть** меню **Файл**;
- нажатием кнопки **Закрыть** на панели инструментов;
- нажатием клавиш **Ctrl**+**F4**.

Выполнение простых программ в Dev-C++

Простую одномодульную программу в Dev-C++ можно сразу **откомпилировать, скомпоновать и выполнить** одним из следующих способов:

- с помощью команды Скомпилировать и выполнитьменю Выполнить;
- нажав кнопку Скомпилировать и выполнить ( ) на панели инструментов;
- нажав клавишу **F9**.

Отдельно **компиляция программы** выполняется:

- с помощью команды Скомпилировать меню Выполнить;
- при нажатии кнопки Скомпилировать ( ) на панели инструментов;
- при нажатии клавиш **Ctrl**+**F9**,

а **выполнение программы**:

- с помощью команды Выполнить меню Выполнить;
- при нажатии кнопки Выполнить ( ) на панели инструментов;
- при нажатии клавиш **Ctrl**+**F10**.

Если при компиляции обнаружены ошибки, в окне Компилятор выводятся соответствующие сообщения:

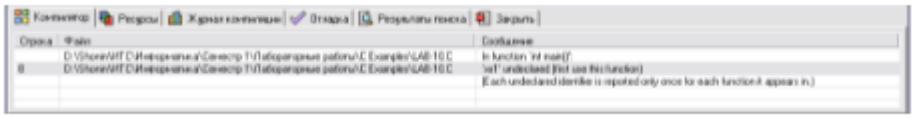


Рис.3.

Если же ошибок не обнаружено, выводится следующее диалоговое окно:

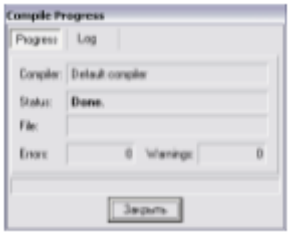


Рис.4.

Если при компиляции и компоновке не обнаружено ошибок, в Dev-C++ открывается окно выполнения программы как окно приложения MSDOS:

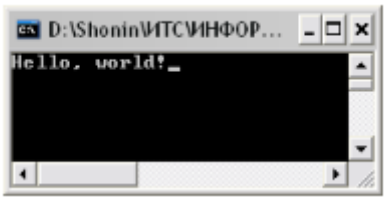


Рис.5.

Следует отметить, что при запуске программы в окне в Windows после выполнения оператора **return** окно выполнения программы автоматически закрывается и часто результат выполнения программы увидеть нельзя. Чтобы этого не происходило, обычно перед оператором **return** ставят оператор **getchar();**, который вызывает функцию ввода символа.

Копирование результатов выполнения программы в другое приложение, например в Блокнот, выполняется с помощью команд Пометить и Копировать или команды Выделить все в подменю команды Изменить системного меню приложения MSDOS.

Следует отметить, что в редакторе Dev-C++ используется кодировка ANSI(Windows-1251), а в окне приложения MSDOS– кодировка OEM. Поэтому буквы кириллицы в строках ввода или в выводимых значениях строк будут передаваться неправильно. Чтобы этого избежать, следует использовать в строках только латинские буквы. Это не распространяется на комментарии в программе.

Отладка программ в Dev-C++

Отсутствие в программе синтаксических ошибок не является гарантией того, что программа будет работать правильно, т. е. выдавать верные результаты. Обычно причиной неправильной работы программы является либо неверный алгоритм, либо неверная реализация правильного алгоритма, либо нарушение правил языка C++ (например, правил по преобразованию типов переменных). Чтобы выявить ошибки при работе программы использовать один из следующих способов:

- вставить в «подозрительных» местах программы отладочные оператор вывода
- запустить программу в режиме отладки.

Для запуска программы в режиме отладки необходимо выполнить одну из следующих операций:

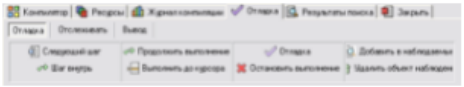
- выполнить команду Отладк ав меню Отладка;
- нажать кнопку Отладка на панели инструментов;
- нажать клавишу **F8**.

При первом запуске будет выведено сообщение:

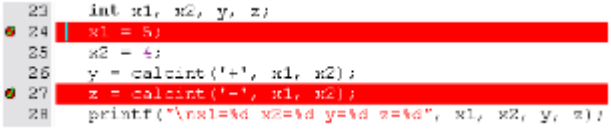
Ваш проект не содержит отладочной информации, хотите разрешить отладку и перестроить проект? Если в процессе отладки были внесены изменения в программу, будет выведено сообщение: Исходный файл новее исполняемого модуля. Хотите пересобрать исполняемый модуль?

На эти вопросы необходимо ответить Yes и повторно запустить отладку.

Управление выполнением программы в режиме отладки производится с помощью команд меню Отладка, либо аналогичных им кнопок на панели инструментов. Однако более удобным является использование вкладки Отладка в нижней части окна Dev-C++. В этой вкладке находятся кнопки всех необходимых для отладки инструментов

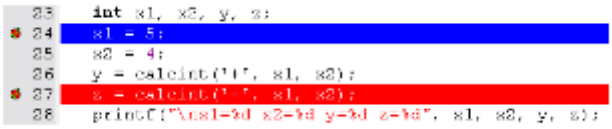


При работе в режиме отладки можно задать в программе **точки прерывания**– операторы, перед выполнением которых работа программы будет приостановлена. Чтобы задать такую точку или точки, необходимо щелкнуть мышью слева от этих операторов, и операторы будут выделены красным цветом:



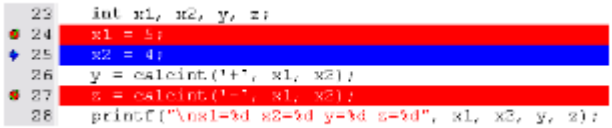
При повторном щелчке мышью слева от оператора точка прерывания для этого оператора будет удалена.

Чтобы запустить программу и остановить выполнение в точке прерывания, надо установить курсор в первой точке прерывания и нажать кнопку Выполнить до курсора. При этом цвет выделения первой точки прерывания измениться с красного на синий:

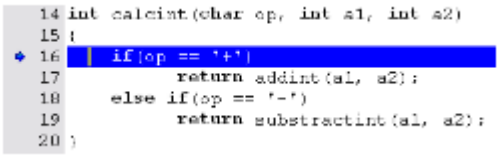


Чтобы остановить выполнение программы в следующей точке прерывания, надо повторно нажать кнопку Выполнить до курсора. Кнопка Продолжить выполнение выполняет программу либо до следующей точки прерывания, либо (после последней точки прерывания) до конца. Для прекращения выполнения программы в режиме отладки надо нажать кнопку Остановить выполнение.

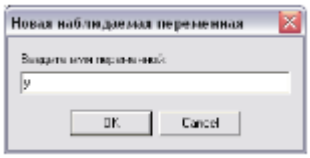
С помощью кнопки Следующий шаг( ) можно выполнять программу по одному оператору (шагу). При этом оператор, который будет выполняться следующим, подсвечивается синим цветом со стрелкой слева:



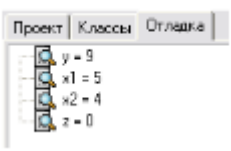
При использовании кнопки Следующий шаг( ) для пошагового выполнения вызов функций пользователя в операторе рассматривается как один шаг. Чтобы перейти к пошаговому выполнению операторов вызываемой функции (в данном случае функции**calcint()**) надо использовать кнопку Шаг внутрь( ):



Чтобы проверить правильность работы программы, часто требуется отслеживать изменение значения переменных при выполнении программы в режиме отладки. В качестве переменной могут выступать простые переменные, а также массивы, структуры и объединения (в этом случае будут отслеживаться все элементы массивов, структур или объединений). Можно также отслеживать отдельные элементы массивов, структур и объединений. Для того, чтобы отследить значения какой-либо переменной или ее элемента при выполнении программы надо (после достижения точки прерывания или выполнения очередного шага программы) нажать на кнопку Добавить в наблюдаемые и ввести имя переменной (или имя элемента переменной, например **a[0]**) в диалоговом окне:



После этого имена и текущие значения переменной будут выводиться в левом окне IDE(как это показано на рисунке для второй точки прерывания):



Значение переменной (или всех ее элементов) можно также узнать, если просто указать на ее имя в тексте программы курсором мыши (для этого во вкладке Параметры среды меню Сервис должна быть включена опция Следить за переменной под мышью).

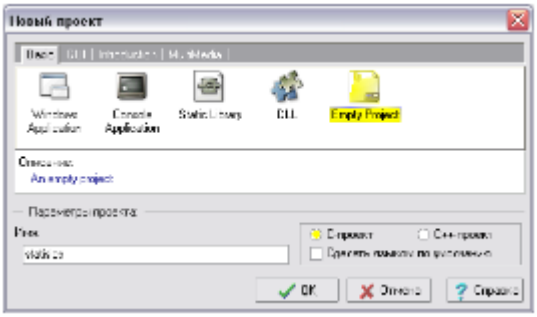
Чтобы убрать переменную из списка наблюдаемых, надо выделить строку для этой переменной в левом окне IDE и нажать на кнопку Удалить объект наблюдения.

Работа с проектами в Dev-C++

Большие и сложные программы обычно разбиваются на программные модули. В C++ программные модули – это функции. В ходе выполнения программы одни функции вызывают другие, и, после выполнения функции, управление возвращается в вызывающую функцию. Выполнение программы начинается с выполнения функции с именем **main()**. Функции небольшой программы можно собрать в одном файле, однако если программа большая, лучше разбить ее на несколько файлов, в каждом из которых содержатся одна или несколько функций. В этом случае при модификации программы достаточно перекомпилировать только те файлы, в которых произведены изменения, а затем повторно выполнить компоновку и запуск программы. Этот процесс можно выполнять вручную, однако, многие IDE позволяют автоматизировать процесс создания и модификации больших программ с помощью проектов.

Проект– это набор файлов, содержащих программные модули, которые рассматриваются как одна программа. Создать новый проект можно либо с помощью команды Проект под меню команды Создать меню Файл, либо нажав кнопку Проект на панели инструментов.

После этого открывается диалоговое окно нового проекта:



Для создания многомодульных программ с выводом в окно MSDOS лучше выбрать вид Empty Project. Затем в области Параметры проекта задается имя проекта (латинскими буквами) и тип проекта (C-проект). После нажатия кнопки OK выбирается папка, в которой будет сохранен файл с данными о проекте (с расширением**.dev**).

После этого в левом окне IDE во вкладке Проект появляется имя проекта. С помощью контекстного меню для проекта можно выполнить следующие операции:

- создать новый файл (команда Создать файл);
- добавить существующий файл программного модуля к проекту (команда Добавить к проекту);
- удалить файл из проекта – файл при этом не уничтожается (команда Удалить из проекта);
- добавить папку в проект (команда Добавить папку);
- настроить параметры проекта (команда Параметры проекта).

В проекте должен быть хотя бы один файл. После создания или добавления всех модулей проекта:

можно запускать проект на компиляцию и выполнение, используя те же команды и кнопки, что и для одномодульных программ. При этом будут повторно компилироваться только те файлы проекта, в которых проведены изменения. Загрузочный файл для проекта имеет то же имя, что и имя проекта, но с расширением **.exe**.

Примечания:

1. Имя файла проекта должно содержать только латинские буквы и цифры.
2. Файл проекта должен быть сохранен в той же папке, что и файлы проекта.
3. Абсолютный путь к файлу проекта не должен содержать пробелов.

Справочная система Dev-C++

Справочная система вызывается с помощью команд меню Справка. С помощью команды Справка Dev-C++можно получить сведения о работе в среде IDE(основных выполняемых операциях и командах меню) в разделе Dev-C++ 5, а в разделе An Introduction to C Programming содержатся краткие сведения о языке C++.

Можно также добавить дополнительные справочные данные в меню Справка. Так, папку **Docs**, содержащую папку **cpp_manpages** с тремя файлами: **c++ man.chm**, **cpp.chm** и **manpages.chm** надо скопировать в папку **Dev-Cpp**. Чтобы сделать доступными эти файлы при работе cDev-C++, надо выполнить следующие действия:

- выполнить команду Настроить меню справки в меню Справка;
- в диалоговом окне Редактор меню справки нажать кнопку Обзор, а затем найти и открыть папку **cpp_manpages**;
- нажать кнопку Добавить и выбрать файл **c++ man.chm**;
- выбрать в раскрывающемся списке Значок в области Параметры меню справки значок для справки и нажать кнопку OK.

В результате в меню Справка появится новая команда, которая выводит на экран справочник поC/C++ (**cpprefrrence.com**) и справочные страницы дляLinux(**manpages.com**).

+Контактная информация – адреса сайтов разработчиков Dev-C++, компилятора Mingw, а также дополнительных ресурсов содержится в диалоговом меню команды ODev-C++меню Справка.

Понятие об алгоритмах

Существует несколько определений понятия алгоритма. Приведем два самых распространенных.

Алгоритм – последовательность чётко определенных действий, выполнение которых ведёт к решению задачи. Алгоритм, записанный на языке машины, есть программа решения задачи.

Алгоритм – это совокупность действий, приводящих к достижению результата за конечное число шагов.

Вообще говоря, первое определение не передает полноты смысла понятия алгоритм. Используемое слово "последовательность" сужает данное понятие, т.к. действия не обязательно должны следовать друг за другом – они могут повторяться или содержать условие.

Свойства алгоритмов:

1. Дискретность (от лат. discretus — разделенный, прерывистый) – это разбиение алгоритма на ряд отдельных законченных действий (шагов).
2. Детерминированность (от лат. determinate — определенность, точность) - любое действие алгоритма должно быть строго и недвусмысленно определено в каждом случае. Например, алгоритм проезда к другу, если к остановке подходят автобусы разных маршрутов, то в алгоритме должен быть указан конкретный номер маршрута 5. Кроме того, необходимо указать точное количество остановок, которое надо проехать, скажем, три.
3. Конечность – каждое действие в отдельности и алгоритм в целом должны иметь возможность завершения.
4. Массовость – один и тот же алгоритм можно использовать с разными исходными данными.
5. Результативность – алгоритм должен приводить к достоверному решению.

Существует несколько способов записи алгоритмов. На практике наиболее распространены следующие формы представления алгоритмов:

-словесная (запись на естественном языке);

-псевдокоды (полуформализованные описания алгоритмов на условном алгоритмическом языке, включающие в себя как элементы языка программирования, так и фразы естественного языка, общепринятые математические обозначения и др.);

-графическая (изображения из графических символов – блок-схема);

-программная (тексты на языках программирования – код программы).

Различают три основных вида алгоритмов:

1. линейный алгоритм,
2. разветвляющийся алгоритм,
3. циклический алгоритм.

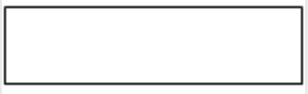
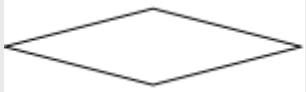
Линейный алгоритм – это алгоритм, в котором действия выполняются однократно и строго последовательно.

Разветвляющийся алгоритм – это алгоритм, в котором в зависимости от условия выполняется либо одна, либо другая последовательность действий.

Циклический алгоритм – это алгоритм, команды которого повторяются некое количество раз подряд.

Блок-схема представляет собой удобный и наглядный способ записи алгоритма.

Блок-схема состоит из функциональных блоков разной формы, связанных между собой стрелками. В каждом блоке описывается одно или несколько действий.

Форма блока	Назначение блока
	начало и конец блок-схемы
	блок ввода данных
	блок выполнения действия
	блок условия
	блок вывода данных

Любая команда алгоритма записывается в блок-схеме в виде графического элемента – блока, и дополняется словесным описанием. Блоки в блок-схемах соединяются линиями потока информации. Направление потока информации указывается стрелкой. В случае потока информации сверху вниз и слева направо стрелку ставить не обязательно. Блоки в блок-схеме имеют только один вход и один выход (за исключением логического блока – блока с условием).

Структура программы

Запускать программу мы научились, теперь можно приступать к изучению языка С++. Прежде чем использовать объекты необходимо подключить файл iostream с помощью директивы include: #include <iostream>

Обратите внимание на то, что название файла указывается внутри угловых скобок без расширения h, так как файл iostream входит в состав стандартной библиотеки С++. После подключения файла все объекты будут доступны через пространство имен std, поэтому при обращении к объекту необходимо указать название пространства имен и два символа

двоеточия перед названием объекта. Для ввод и вывода данных в языке С++ предназначен объект cin>> и cout<<, объявленные в файле iostream. Например, вывести строку "Hello, world!" можно так:

std::cout<<"Hello, world!";

Если каждый раз указывать название пространства имен лень, то можно импортировать все идентификаторы в глобальное пространство имен с помощью инструкции:

usinq namespace std;

В этом случае название пространства имен указывать не нужно: cout<<"Hello, world!";

Структура обычной программы выглядит так:

```
#include <iostream> // Подключение заголовочных файлов
using namespace std; // Пространства имен
int main()              // Главная функция
{ cout<< "Welcome to C++!";  // Инструкции – тело функции
return 0;                          // успешное завершение
}
```

Индивидуальные задания

Задание 1. Знакомство с интерфейсом DevC++

Запустите интегрированную среду DevC++.

Ознакомьтесь с интегрированной средой, описание которой приведено в теоретической части работы.

Задание 2. Составление алгоритма решения задачи.

1. Набрать программу в окне редактирования DevC++, которая вычислят длину окружности, площадь круга и объем шара, имеющих радиус R (любое целое число).
2. Для каждой строки сделать комментарии характеризующую ее. Сохраните программу под своим именем, откомпилировать и выполнить.
3. Для данной задачи построить блок-схему в вашем отчете. Отчет будет содержать титульный лист, само задание, блок- схему, код программы и скрин окна вывода.

```
#include < iostream>
 using namespace std;
int main(){
int R;
float L, S, V;
const float Pi = 3.14;
cout <<" Vvedite R=";
cin >> R;
L= 2 * P i * R;
S= Pi*R*R;
V=4/3* Pi*R*R*R;
 cout <<"L="<<L<<"S="<<S<<"V="<<V;
return 0;}
```

Контрольные вопросы

1. Какое имя носит главная исполняемая функция Си++? Что такое редактирование и компиляция программы?
2. Дайте определение понятия «переменная» и «идентификатор».
3. Сколько переменных требуется описать в программе, если необходимо решить следующую задачу – «С клавиатуры вводятся три числа, необходимо вывести на экран значение минимального из этих трех чисел»?
4. Какие инструкция используется в Си++ для ввода и вывода информации?
5. Для чего надо подключать директиву iostream.
6. Чем отличается комментарий // от комментария /*, */?
7. Основные команды меню интегрированной среды DevC++.

8. С какой целью в начале кода пишется строка using namespace std; и что надо делать, если эта строка отсутствует?

9. Раскрыть понятие алгоритма: виды, свойства, способы записи.

10. Этапы прохождения программ.

Лабораторная работа №2. Арифметические и логические операции в языке С++.

***Цель работы**: Изучить типы данных языка С++, правила записи арифметических и логических выражений, некоторых операторов специального. Научиться составлять линейные программы.*

Типы данных

В языке С++ имеются следующие встроенные-простые типы и спецификаторы которые расширяют возможности этих типов, они представлены в следующей таблице

Название	Обозначение	Диапазон значений
Байт	`char`	от -128 до +127
без знака	`unsigned char`	от 0 до 255
Короткое целое число	`short`	от -32768 до +32767
Короткое целое число без знака	`unsigned short`	от 0 до 65535
Целое число	`int`	от – 2147483648 до + 2147483647
Целое число без знака	`unsigned int` (или просто `unsigned`)	от 0 до 4294967295
Длинное целое число	`long`	от – 2147483648 до + 2147483647
Длинное целое число без знака	`unsigned long`	от 0 до 4294967295
Вещественное число одинарной точности	`float`	от ±3.4e-38 до ±3.4e+38 (7 значащих цифр)
Вещественное число двойной точности	`double`	от ±1.7e-308 до ±1.7e+308 (15 значащих цифр)
Вещественное число увеличенной точности	`long double`	от ±1.2e-4932 до ±1.2e+4932
Логическое значение	`bool`	значения true(истина) или false (ложь)

Операции языка С++

Операции - это комбинации символов, определяющие действия по преобразованию значений, они делятся на:

1. **Арифметические операции**

+ сложение
- вычитание
* умножение
/ деление
% остаток
++ увеличить на единицу, префиксная и постфиксная формы
-- уменьшить на единицу, префиксная и постфиксная формы

2. **Операции сравнения**

== равно
!= не равно
< меньше
> больше
<= меньше или равно
>= больше или равно

3. Логические операции

&& логическое И
|| логическое ИЛИ
! логическое НЕ
Таблицы истинности логических операций приведены в следующих таблицах:

x	*y*	*x* and *y*
0	0	0
0	1	0
1	0	0
1	1	1

x	*y*	*x* or *y*
0	0	0
0	1	1
1	0	1
1	1	1

x	not *x*
0	1
1	0

x	*y*	*x* xor *y*
0	0	0
0	1	1
1	0	1
1	1	0

4. Битовые операции

& битовое И
| битовое ИЛИ
^ битовое ИСКЛЮЧАЮЩЕЕ ИЛИ
~ битовое НЕ
<< сдвиг влево
>> сдвиг вправо

5. Операции присваивания

= присваивание

И особняком стоит операция **sizeof**. Эта операция позволяет определить, сколько памяти занимает то или иное значение, она имеет две формы записи. Например:

```
sizeof(long);    // сколько байтов занимает тип long
sizeof (b);      // сколько байтов занимает переменная b
```

Каждая *операция* имеет *приоритет*. Если в *выражении* несколько операций, то первой будет выполнена операция с более высоким *приоритетом*. Если же *операции* одного и того же *приоритета*, они выполняются слева направо

№	Операция	Порядок выполнения
1	[ ] . () (вызов метода)	Слева направо
2	! ~ ++ -- +(унарный) -(унарный) () (приведение) new	Справа налево
3	* / %	Слева направо
4	+ -	Слева направо
5	<< >> >>>	Слева направо
6	< <= > >= instanceof	Слева направо
7	== !=	Слева направо
8	&	Слева направо
9	^	Слева направо
10	|	Слева направо
11	&&	Слева направо
12	||	Слева направо
13	?:	Слева направо
14	= += -= *= /= %= |= ^= <<= >>= >>>=	Справа налево

Функции библиотеки math.h

Функции для расчета математических выражений находятся в библиотеке math.lib (подключение библиотеки: **#include math.h**). Все аргументы в тригонометрических функциях задаются в радианах. Параметры и аргументы всех остальных функций имеют тип **double** (кроме abs(*x*)).

Математическая функция	Программная запись	Описание
$\|x\|$	`fabs(x)`	Модуль числа.
$\sin x$	`sin(x)`	Синус числа, аргумент в радианах.
$\cos x$	`cos(x)`	Косинус числа, аргумент в радианах.
$\mathrm{tg}\, x$	`tan(x)`	Тангенс числа, аргумент в радианах.
e^x	`exp(x)`	Экспонента числа.
$\ln x$	`log(x)`	Натуральный логарифм числа.
$\lg x$	`log10(x)`	Десятичный логарифм числа.
x^y	`pow(x, y)`	x в степени y.
10^x	`pow10(x)`	Степень десяти.
$\sqrt{x}$	`sqrt(x)`	Квадратный корень из числа.
$\arcsin x$	`asin(x)`	Арксинус числа, в радианах.
$\arccos x$	`acos(x)`	Арккосинус числа, в радианах.
$\mathrm{arctg}\, x$	`atan(x)`	Арктангенс числа, в радианах.
π	`M_PI`	Число π =3.141593

Пример 1. Написать программу для вычисления линейного арифметического выражения где нужно вычислить z:

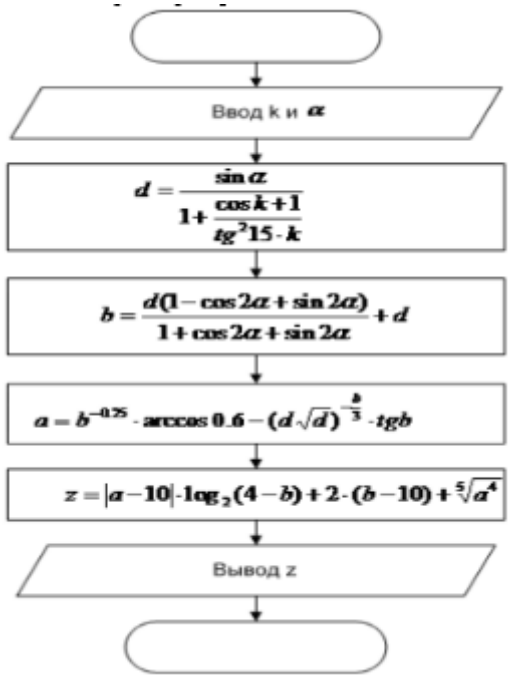
$$z = |a-10| \cdot \log_2(4-b) + 2(b-10) + \sqrt[5]{a^4}, \text{ где } a = b^{-0.25} \cdot \arccos 0.6 - (d\sqrt{d})^{-\frac{b}{3}} \cdot tgb,$$

$$b = \frac{d(1-\cos 2\alpha + \sin 2\alpha)}{1+\cos 2\alpha + \sin 2\alpha} + d, \; d = \frac{\sin\alpha}{1+\frac{\cos k + 1}{tg^2 15 \cdot k}}.$$

Для произвольных k и α. Исходные данные: k – вещественный тип, α будет обозначаться alf – вещественный тип. Результат: z – вещественный тип. При вычислении надо учитывать, что при вычислении выражения, все переменные, участвующие в этом выражении должны быть известны. Это диктует последовательность вычислений.

Тестовый пример: при k=1 и α=1 z=-0,48473. (Проверка результатов выполнялась в Excel)

```
#include <iostream>
#include <math.h>
#include <stdlib.h>
using namespace std;  int main () {
double k, alfa, a, b, d, z; cout << "Vvedite k: "; cin >> k;
cout << "Vvedite afa: "; cin >> alfa;
d=sin(alfa)/(1+(cos(k)+1)/(pow(tan(15),2)*k));
b=d*(1-cos(2*alfa)+sin(2*alfa))/(1+cos(2*alfa)+sin(2*alfa))+d;
a=pow(b,(-0.25))*acos(0.6)-pow(d*sqrt(d),(-b/3))*tan(b);
z=fabs(a-10)*log(4-b)/log(2.0)+2*(b-10)+pow(pow(a,4),(1,0/5));
cout<<"z= "<<z<<endl;
return(0);}
```



Пример 2. Написать программу для вычисления линейного арифметического выражения

h=$\frac{x^{2y}+e^{y-1}}{1+x\,|y-tgz|}+10\sqrt[3]{x}$ -ln(z)

При x = 2.45, y = – 0.423*10^{-2}, z = 1.232*10^{3} ответ *h* = 3,4655.

Текст программы:

```
#include <iostream>
#include <math.h>
using namespace std;
int main () {
double x,y,z,a,b,c,h; cout << "Vvedite x: ";
cin >> x;
cout << "Vvedite y: "; cin >> y;
cout << "Vvedite z: "; cin >> z;
a = pow(x,2*y)+exp(y-1); b = 1+x*fabs(y-tan(z));
c = 10*pow(x,1/3.)-log(z); h = a/b+c;
cout << "Result h= " << h << endl;
return 0;
}
```

Индивидуальные задания

Задание 1. Написать программу согласно номеру своего варианта для вычисления линейного арифметического выражения, а так же сделать блок схему.

1. Даны x, y, z. Вычислить a, b, если

$$a = \frac{\sqrt{|x-1|}-\sqrt{|y|}}{1+x^2/2+x^2/4},\ b = x(arctg(z)+e^{-(x+3)})$$

;

2. Даны x, y, z. Вычислить a, b, если

$$a = \frac{3+e^{y-1}}{1+x^2|y-tg(z)|},\ b = 1+|y-x|+\frac{(y-x)^2}{2}+\frac{|y-x|^3}{3}$$

;

3. Даны x, y, z. Вычислить a, b, если

$$a = (1+y)\frac{x + y/(x^2+4)}{e^{-x-2} + 1/(x^2+4)},\ b = \frac{1+\cos(y-2)}{x^4/2 + \sin^2 z};$$

4. Даны x, y, z. Вычислить a, b, если

$$a = y + \frac{x}{y^2 + \left|\frac{x^2}{y + x^3/3}\right|},\ b = \left(1 + tg^2\frac{z}{2}\right);$$

5. Даны x, y, z. Вычислить a, b, если

$$a = \frac{2\cos(x - \pi/6)}{1/2 + \sin^2 y},\ b = 1 + \frac{z^2}{3 + z^2/5};$$

6. Даны x, y, z. Вычислить a, b, если

$$a = \frac{1 + \sin^2(x+y)}{2 + \left|x - 2x/(1 + x^2y^2)\right|} + x,\ b = \cos^2\left(arctg\frac{1}{z}\right);$$

7. Даны x, y, z. Вычислить a, b, если

$$a = \ln\left|\left(y - \sqrt{|x|}\right)\left(x - \frac{y}{z + x^2/4}\right)\right|,\ b = x - \frac{x^2}{3!} + \frac{x^3}{4!}$$

8. Даны x, y, z. Вычислить a, b, если

$$a = (1+y)\frac{x + y/(x^2+4)}{e^{-x-2} + 1/(x^2+4)},\ b = \frac{1+\cos(y-2)}{x^4/2 + \sin^2 z}$$

9. Вычислить s, при x = 16.55*10*3 ; y = –2.75; z = 0.15 ответ s = –40.6307

$$s = \sqrt{10\left(\sqrt[3]{x} + x^{y+2}\right)}\left(\arcsin^2 z - |x - y|\right).$$

10. Вычислить s при x = –4.5; y = 0.75*10^{-4} ; z = –0.845*10^2 ответ s = –3.23765.

$$s = \frac{\sqrt[3]{9 + (x-y)^2}}{x^2 + y^2 + 2} - e^{|x-y|}\mathrm{tg}^3 z.$$

Задание 2. С помощью операции sizeof составить программу вывода размера (в байтах) всех типов данных.

Задание 3. Написать программу демонстрирующую работу всех операций.

Контрольные вопросы

1. Какие математические функции находятся в заголовочном файле math.h
2. Какие типы данных существуют в языке Си++?
3. Чем отличается спецификаторы от типов данных и назовите их.
4. Какие операции относят к логическим, приведите примеры использования.
5. Какие операции относят к арифметическим, приведите примеры использования.
6. Какие операции относят к побитовым, приведите примеры использования.
7. Какие операции относят к операциям сдвига, приведите примеры.
8. Перечислите основные операции языка С++ в порядке убывания приоритетов.

Лабораторная работа №3. Явное и неявное преобразование типов.

Цель работы: *изучение работы компилятора неявного преобразования типов.*

Теоретические сведения

В С++ существует два типа преобразования:

1. Неявное преобразование.

2. Явное преобразование (также известное, как приведение типов).

Неявное преобразование.

Неявное преобразование типов данных выполняет компилятор C++. Результат любого вычисления будет преобразовываться к наиболее точному типу данных, из тех типов данных, которые участвуют в вычислении. Рассмотрим таблицу с преобразованиями типов данных на примере операции деления.

Таблица 1 — Неявное преобразование типов данных в C++

x	y	Результат деления	Пример
делимое	делитель	частное	x = 15 y = 2
int	int	int	15/2=7
int	float	float	15/2=7.5
float	int	float	15/2=7.5

Из таблицы видно, что меняя переменные различных типов данных местами, результат остается тот же (делимое и делитель). Существуют правила неявного преобразования:

1. тип char приводится к short;
2. short - к int;
3. signed - к unsigned;
4. все целые типы преобразуются к long, а long - к unsigned long;
5. Целое без знака преобразуется к более длинному целому со знаком путём расширения нулём.
6. Целое со знаком преобразуется к более длинному целому со знаком путём размножения знаков.
7. Целое со знаком преобразуется к более короткому целому со знаком путём усечения старших бит.
8. При преобразовании целого со знаком к целому без знака целое со знаком преобразуется к размеру целого без знака и результат интерпретируется как целое без знака.
9. Преобразование целого со знаком к плавающему типу происходит без потери информации, за исключением случая преобразования типа long к float, тогда точность может быть частично потеряна.
10. Величина типа float преобразуется к double без изменения значений. Величины double, преобразованные к float, представляются точно, если это возможно.
11. При преобразовании величины с плавающей точкой к целым типам они сначала преобразуются к типу long, при этом дробная часть отбрасывается, а затем величины типа long преобразуются к требуемому типу. Если значение слишком велико для long, то результат будет не определён (отбрасывается младшая часть мантиссы).

Явное преобразование

Явное преобразование данных выполняет сам программист. Что же касается явного преобразования, то оно необходимо для того чтобы выполнять некоторые манипуляции, тем самым меняя результат вычисления. Самый простой способ явного преобразования типа следующий:

```
int x = 5;
double y = 15.3;
    x = (int) y;
    y = (double) x;
```

Еще один пример явного преобразования типов данных:

float(15) / 2 // результат равен 7.5, число 15 преобразуется в вещественный тип данных float.
double(15) / 2 // результат равен 7.5 – тоже самое!!!

В C++ также предусмотрена унарная операция приведения типа:

static_cast</*тип данных*/>(/*переменная или число*/)

Пример: cout<< static_cast< float>(15)/2; // результат равен 7.5

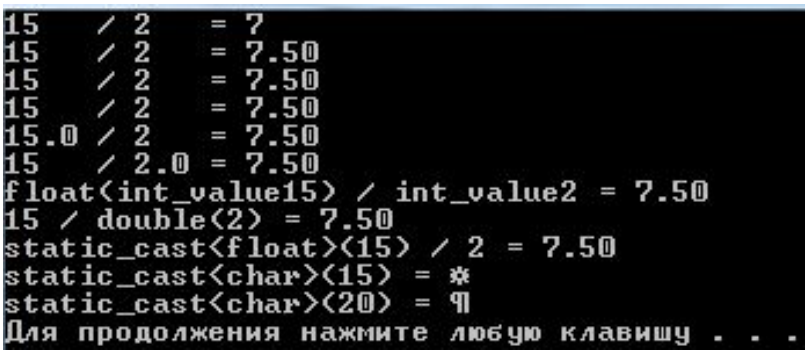
Пример: с переменной:

```
int ret=15;
static_cast<float>(ret)/2 //результат равен 7.5
```

В случае с переменной надо понимать, что переменная ret не преобразуется в тип данных float, а всего лишь создается временная копия переменной ret с типом данных float.

Рассмотрим на практике все способы явного и неявного преобразования типов данных.

```
1.
2. #include <iostream>
3. #include <iomanip.h>
4. using namespace std;
5. int  main() {
6. int int_value15 = 15, int_value2 = 2; // объявляем две переменные типа int
7.  float float_value15 = 15, float_value2 = 2; // объявляем две переменные типа float
8.  cout << fixed << setprecision(2); // определяем, два знака после запятой
9.  cout<< "15  / 2 = " << int_value15 / int_value2 << endl;//неявное преобраз. типов данных
10. cout << "15 / 2  = " << int_value15/ float_value2 << endl;
11. cout << "15   / 2   = " << float_value15 / int_value2   << endl;
12. cout << "15   / 2   = " << float_value15 / float_value2 << endl;
13. cout << "15.0 / 2   = " << 15.0 / 2 << endl  ;// явное преобразование типа данных, число 15.0
        -     число с плавающей точкой
14. cout << "15/ 2.0 = " <<15/2.0 << endl; //явное преобразование типа данных, число 2.0  -
        число с плавающей точкой
15.cout << "float(int_value15) / int_value2 = " << float(int_value15) / int_value2 << endl ;// явное
        преобразование типа данных
16. cout << "15 / double(2) = "<< 15/double(2) << endl;//используя приводимый тип как
        функцию
17.cout << "static_cast<float>(15) / 2 = " << static_cast<float>(15) / 2 << endl; // унарная
        операция приведения типа
18. cout << "static_cast<char>(15) = " << static_cast<char>(15) << endl;  // можно печатать
        различные символы из таблицы ASCII,
19.  cout << "static_cast<char>(20) = " << static_cast<char>(20) << endl; // в скобочках
        прописываем код символа, который находим в таблице ASCII
20. system("pause");
21.return 0;}
```



В **строке 3** подключена библиотека манипуляций ввода/вывода <iomanip.h>, эта библиотека нужна для использования различных манипуляторов, в нашем случае — fixed setprecision(). В **строке 6** специально объявили 2 переменные типа int, аналогично две переменные типа float в **строке 7**, эти переменные нужны для преобразования их значений в другие типы данных. В **строке 8** после оператора cout и операции сдвига в поток вывода << стоят два манипулятора fixed и setprecision(). Манипулятор fixed — это не параметризированный манипулятор, так как никаких параметров не принимает, пишется без круглых скобок. Данный манипулятор применяется в паре с параметризированным манипулятором setprecision() и выполняет фиксированное отображение разрядов после запятой. А манипулятор setprecision() отображает количество знаков после запятой, причём то, которое указано в скобочках. В **строках 9, 10, 11, 12** показаны примеры неявного преобразования типов данных, эти примеры взяты из **таблицы 1**. В **строках 13, 14** показан один из способов явного преобразования данных. Суть такого способа заключается в том, что нужно дописать запятую и нуль к целому числу. В **строках 15, 16** явное преобразование выполняется посредством использования приводимых типов как функций, внутри скобочек которых, нужно указать значение или переменную, которую необходимо преобразовать. В **строках 17, 18, 19** выполняется явное преобразование типов данных с помощью унарной операции преобразования данных. В круглых скобочках указывается, переменная или значение, которое нужно преобразовать, а в обрамлении знаков < > тип данных, к которому нужно преобразовать.

Задания

1. Введите следующий код:

```
int x = 12;
int y = 7;
double z = x/y;    // Чему в этом случае будет равно z, почему ?
int x = 12;
int y = 7;
double z = (double)x/y;     // Чему будет равно z в этом случае, почему?
```

Какое из приведённого выше списка правило, привело к полученному результату?

2. Заполните таблицу

Неявное преобразование типов данных в С++

	x=15	**y=2**	**z=x/y**	Результат
№	делимое	делитель	частное	
1	int	int	int	
2	int	int	float	
3	int	float	float	
4	float	int	float	

3. Даны определения переменных:

```
char cval; int ival;
float fval; double dva1;
unsigned int ui;
```

Какие неявные преобразования типов будут выполнены?

a) cva1 = 'a' + 3;
b) fval = ui - ival * 1.0;
c) dva1 = ui * fval;
d) cva1 = ival + fvat + dva1;

4. Каковы значения i и d после каждого задания?
int i; double d;
a) d=i=3.5;
b) i=d=3.5;

5. Каков результат каждого из этих выражений?
long unsigned ul1 = 3, ul2 = 7;
a) ul1 & ul2 b) ul1 | ul2
c) ul1 && ul2 d) ul1 || ul2

6.Даны следующие определения, char cval; int ival; unsigned int ui; float fval; double dval; определите неявные преобразования типов, если таковые имеются:
a) cval=’a’+3; b) fval=ui-ival*1.0;
c) dval = ui * fval; d) cval = ival + fval + dval;

Лабораторная работа №4. Программирование алгоритмов разветвляющейся структуры.

Цель работы: *выработать навыки разработки алгоритмов ветвления, научиться правильно использовать конструкции языка Си++ при разработке программ, содержащих условие.*

Для программирования ветвящихся алгоритмов в языке Си++ имеется несколько различных средств, основных 4: ***if, if-else, ? и switch.*** Рассмотрим первые 2 оператора.

Оператор условной передачи управления if

Оператор if используется для условного выполнения фрагмента программы. Формат оператора: **if (выражение) оператор**;

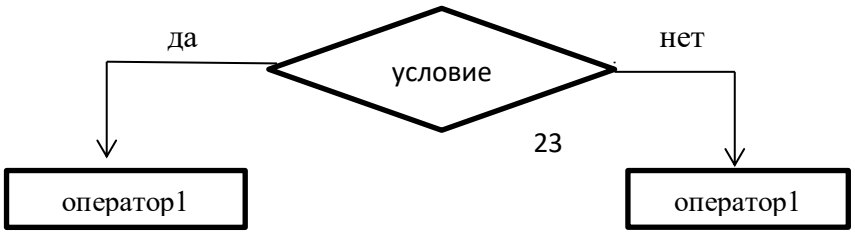
Выражение должно заключаться в круглые скобки и в результате выполнения должно давать значение: истина или ложь. Если выражение истинно, выполняется оператор, следующий за выражением. Если выражение ложно, то оператор не выполняется. Оператор if, связанный с блоком выполняемых операторов, выгладит следующим образом:

```
if (выражение)
{оператор1;
оператор2;}
```

Если необходимо, чтобы программа выполняла два разных действия в зависимости от истинности некоторого выражения, то используется оператор ***if-else***. Формат оператора:

```
if (выражение)   оператор1;
else   оператор2;
```

В этом операторе, если выражение истинно, то выполняется оператор1, если же выражение ложно, выполняется оператор2. И оператор1, и оператор2 может быть блоком операторов, заключенным в фигурные скобки. Блок схема условного оператора, *полной формы* следующая:



Можно использовать и вложенные операторы if. При этом else связывается с ближним if, еще не имеющим части else.

Для выполнения многочисленных последовательных сравнений часто используется комбинация операторов if-else-if. В общем виде они выглядят следующим образом:

```
if (выражение1)
   оператор1;
else  if(выражение2)
  оператор2;
else if(выражение3)
   оператор3;
```

Каждое действие может быть составным блоком в фигурных скобках. Такая управляющая логическая структура вычисляет каждое выражение до тех пор, пока не найдет истинное. Когда это происходит, все оставшиеся проверочные условия опускаются.

Пример 1. Вычислить: $Y = \begin{cases} x^2 + 4, если & X \le 5 \\ 7x, если & X > 5 \end{cases}$

```
#include <iostream>
using namespace std;
main()
{ double y,x;
cout<<”введите значение  x  \n”;
cin>>x;
if (x<=5) y=pow(x,2);
else if (x>5) y=7*x;
cout<<” значение y=”<<y;
return(0);}
```

Пример 2. Программа вводит с клавиатуры целое число. Если это число больше десяти, то программа должна выполнить одно действие, иначе — другое.

```
#include <iostream.h>
int main()
{  double num;
   cout << "Введите произвольное число: ";
   cin >> num;
   if (num < 10) { // Если введенное число меньше 10.
      cout << "Это число меньше 10." << endl;
   } else { // иначе
      cout << "Это число больше либо равно 10." << endl;
   }
   return 0;}
```

Индивидуальные задания

Задание 1. Написать программу согласно номеру своего варианта для вычисления системы уравнений , а так же сделать блок схему.

1. $y = \begin{cases} x^2, если\ 0 \le x \le 3; \\ 1 / x, если\ x < 0;\ x > 10; \\ 2 + x^2, если\ 3 < x \le 10. \end{cases}$

2. $y = \begin{cases} \ln|x|, если\ x > 1; \\ e^x, если\ 0 < x \le 1; \\ 2x^2, если\ x \le 0. \end{cases}$

3. $y = \begin{cases} 15, \text{ если } 0{,}1 \le x < 0{,}5; \\ 20, \text{ если } x > 10; \\ 4x, \text{ в остальных случаях.} \end{cases}$

4. $y = \begin{cases} \cos^2(x^2 - 1), \text{ если } x > 1; \\ 2x, \text{ если } 0 \le x \le 1; \\ x^4 + 0{,}5\sin x, \text{ если } x < 0. \end{cases}$

5. $y = \begin{cases} \sin 2x + 5, \text{ если } x \le 10; \\ \cos^2(x - 5), \text{ если } x > 12; \\ e^x, \text{ если } 10 < x \le 12. \end{cases}$

6. $y = \begin{cases} 1, \text{ если } x = 3; \\ 2, \text{ если } x = 6; \\ 3, \text{в остальных случаях.} \end{cases}$

7. $y = \begin{cases} e^x + 10, \text{ если } -10 < x < 0; \\ e^x - \sin x, \text{ если } x \ge 0; \\ \ln|x|, \text{ в остальных случаях.} \end{cases}$

8. $y = \begin{cases} 5x + 6, \text{ если } 6 < x < 7; \\ 2, \text{ если } 7 \le x < 10; \\ 9, \text{в остальных случаях.} \end{cases}$

9. $y = \begin{cases} x^2 + 4, \text{ если } x \ge 1; \\ x^3 - 2, \text{ если } 0 < x < 1; \\ x^4 + 2, \text{ если } x \le 0. \end{cases}$

10. $y = \begin{cases} \cos x \cdot e^x, \text{ если } 0 < x < 3; \\ \ln|x|, \text{ если } x \ge 3; \\ |x|, \text{если } x \le 0. \end{cases}$

Задание 2. Написать программу согласно номеру своего варианта, сделать блок схему.

1. Напишите программу-модель анализа пожарного датчика в помещении, которая выводит сообщение «Пожарная ситуация», если температура (ее значение вводится с клавиатуры) в комнате превысила 60ºС.

2. Ракета запускается с точки на экваторе и развивает скорость v км/с. Каков результат запуска? Замечание: если v<=7.8 км/с, то ракета упадет на Землю, если 7.8<v<11.2, то ракета станет спутником Земли, если 11.2□v□16.4, то ракета станет спутником Солнца, если v>16.4, то ракета покинет Солнечную Систему.

3. Рис расфасован в два пакета. Вес первого - m кг, второго – n кг. Составьте программу, определяющую: а) какой пакет тяжелее – первый или второй? б) вес более тяжелого пакета.

4. Написать программу, которая бы запрашивала возраст мужчины и сообщала, сколько лет ему осталось до пенсии, либо что он уже пенсионер.

5. Туристы вышли из леса на шоссе неподалеку от километрового столба с отметкой А км и решили пойти на ближайшую автобусную остановку. Посмотрев на план местности, руководитель группы сказал, что автобусные остановки расположены на километре В и на километре С. Куда следует пойти туристам?

6. Написать программу, которая бы запрашивала целое число и распечатывала любое его значение, кроме13. Если заданное число равно13, вместо него печатается число 77.

7. Валя и Вера на своем садовом участке собрали А кг клубники. Из них В кг собрала Вера. Кто из девочек собрал клубники больше и на сколько?

8. Даны длины трех отрезков a, b, c. Если можно построить треугольник по этим трем отрезкам, то вычислить его периметр и площадь.

9. Первая бригада маляров за t1 час покрасила А м2 стен, а вторая бригада за t2 часа покрасила В м2. У какой бригады производительность труда выше и на сколько?

10. Каждое утро майор Знаменский заходит в тир и делает 5 выстрелов через плечо. Если он набирает 50 очков, то вечером идет с Зиночкой в ресторан, а если меньше, то на тренировку в тир. Написать программу, которая распечатывает планы майора на вечер.

Задание 3. Написать программу согласно номеру своего варианта, сделать блок схему.

1. Заданы три числа. Определить, могут ли они представлять собой стороны одного треугольника. Найти наибольшее и присвоить его значение переменной D.

2. Определить, какому квадранту (т.е. четверти) принадлежит точка M(x, y). Значение x и y выбирают произвольно.

3. Определить, принадлежит ли точка N(3,2; 7,5) треугольнику, образованному осями OX, OY и прямой y = 7,6 - 3,5x

4 Выяснить, лежит ли точка (x, y) внутри круга радиуса R с центром в начале координат, или вне круга, или на окружности.

5. Заданы два отрезка на оси $A_1 = 3,2$; $B_1 = 7,8$; $A_2 = 1$; $B_2 = 5$. Если точка x принадлежит одновременно первому и второму отрезкам, то $P = tg\ x$.

6. Заданы две точки: $M_1(8,21; -8,1)$; $M_2(2,14; 15,81)$. Присвоить переменной W значение Q, если расстояние между точками равно нулю и значение $P = Q / 31$ в противном случае. Здесь $Q = 6,28 + \sin x – ab / (a + b)$; $a = 12,4$; $b = 3,62$; x выбирают произвольно.

7. Заданы длины трех сторон треугольника a, b, c. Определить, является ли треугольник равнобедренным, равносторонним или разносторонним.

8. Скорость движения объекта изменяется в пределах от V_0 до V_m по формуле $V_t = V_0 + a_t$. Составить алгоритм и программу для вычисления скорости, учитывая, что при $t < 0$ принять $V_t = V_0$, а при $t > m$ принять $V_t = V_m$. Здесь $a = 2,5$; $t = 40$; $V_0 = 25$; $V_m = 117$;

9. Прямоугольник задан в плоскости четырьмя точками $A(x_a; y_a)$; $B(x_b; y_b)$; $C(x_c; y_c)$; $D(x_d; y_d)$. Составить алгоритм и программу для определения, принадлежит ли данному прямоугольнику точка M(14; 0,5), если $x_a = x_b = 12$; $x_c = x_d = 20$; $y_a = y_d = 1$; $y_c = y_b = 3$;

10. Заданы длины трех сторон треугольника m, n, h. Определить, является ли треугольник прямоугольным? $m = 3$; $n = 5,1$; $h = 4$.

Контрольные вопросы

1. Напишите формат и пример условного оператора в полной форме.
2. Напишите формат и пример условного оператора в краткой форме.
3. Какие операции относят к логическим, приведите примеры использования.

Лабораторная работа №5. Программирование алгоритмов разветвляющейся структуры. Оператор switch и условный (тернарный) оператор ? :

Цель работы: *Научится программировать разветвляющиеся алгоритмы, используя оператор-переключатель.*

Теперь мы рассмотрим оставшиеся 2 оператора **switch** и **тернарый оператор ? :**.

Операция условия ? : . Это единственная операция, которая имеет три операнда. Формат операции:

выр1 ? выр 2 : выр 3;

Данная операция реализует алгоритмическую структуру ветвления. Алгоритм ее выполнения следующий:

1. первым вычисляется значение *выр 1*, которое обычно представляет собой некоторое условие.
2. Если оно истинно, т. е. не равно 0, то вычисляется *выр 2* и полученный результат становится результатом операции.
3. В противном случае в качестве результата берется значение *выр 3.*

Пример 1. Вычисление абсолютной величины переменной X можно организовать с помощью одной операции:

```
X<0 ? -X : X;
```

Пример 2. Выбор большего значения из двух переменных a и b:

```
max=(a<=b)?b:a;
```

Пример 3. Заменить большее значение из двух переменных a и b на единицу:

```
(a>b)?a:b=l;
```

Правила языка в данном случае позволяют ставить условную операцию слева от знака присваивания.

Пример. Вычислить: $Y = \begin{cases} x^2 + 4, если \quad X \le 5 \\ 7x, если \quad X > 5 \end{cases}$

```
#include <iostream.h>
main()
{ double y,x;
cout<<”введите значение  x  \n”;
cin>>x;
x<=5? y=pow(x,2): y=7*x;
cout<<” значение y=”<<y;
return(0);}
```

Часто необходимо сравнить некоторую переменную или выражение с несколькими значениями. Для этого можно использовать либо вложенные операторы if-else-if, либо оператор ***switch***. Формат оператора switch следующий:

```
switch (общее- выражение)
{case  константа1 : оператор1; break;
   case  константа2 : оператор2; break;
  default: операторы;}
```

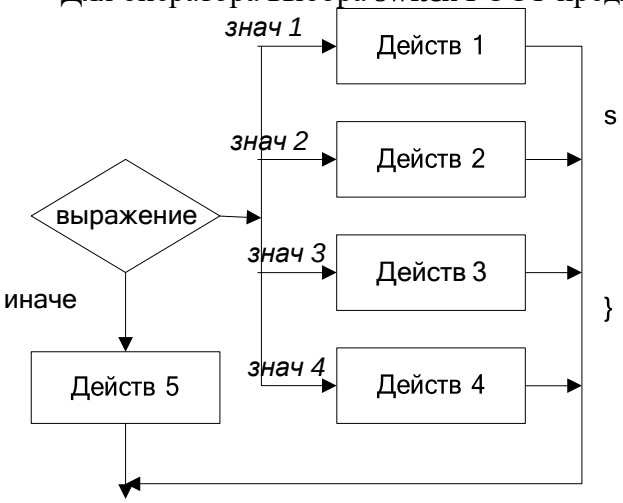
Сначала вычисляется общее- выражение и сравнивается с константой 1. Если результат вычисления выражения равен константе1, выполняется оператор1 и все последующие операторы. Если же требуется, чтобы при совпадении выражения с какой-либо константой выполнялись не все операторы, а только тот оператор, который непосредственно следует за соответствующим case, то нужно использовать ключевое слово ***break*** (прервать выполнение).

Если выражение не равно константе1, то процесс повторяется для константы2 и т.д. Если в конструкцию включен элемент “default", то в том случае, когда результат вычисления выражения не совпадает ни с одним из значений констант, будут выполняться операторы, записанные после "default".

Пример 1. Фрагмент программы, которая выводит названия чисел от 1 до 4

```
switch (n)
{case  1: cout << "один"; break;
  case  2: cout << "два"; break;
  case  3: cout << "три"; break;
  case  4: cout << "четыре"; break;
  default:
  cout << "неизвестное число"; }
```

Для оператора выбора switch ГОСТ предлагает следующую блок-схему:



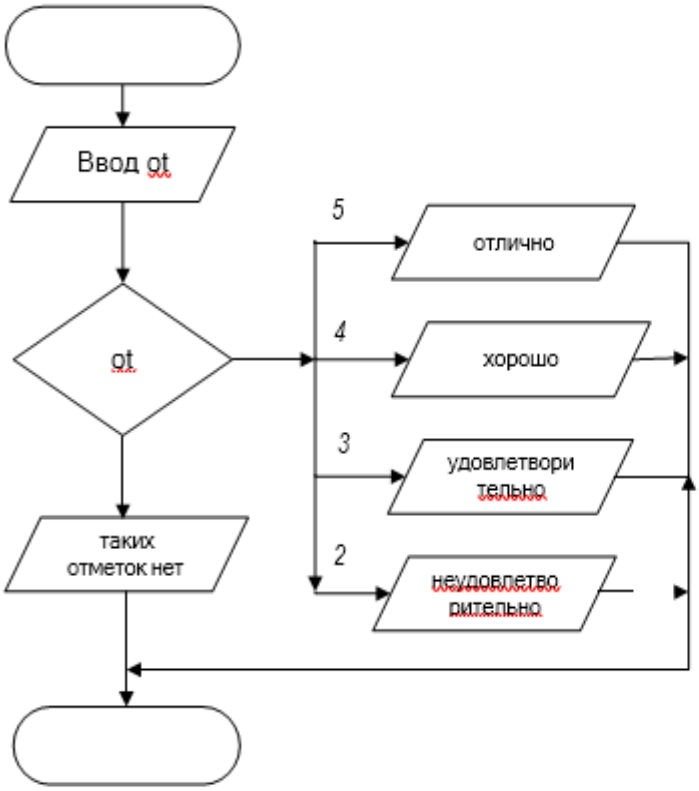
Пример 2. Ввести числовые оценки, вывести текстовые оценки.

Исходные данные: ot – целый тип числовая оценка.

Результатом является вывод оценки в виде текста.

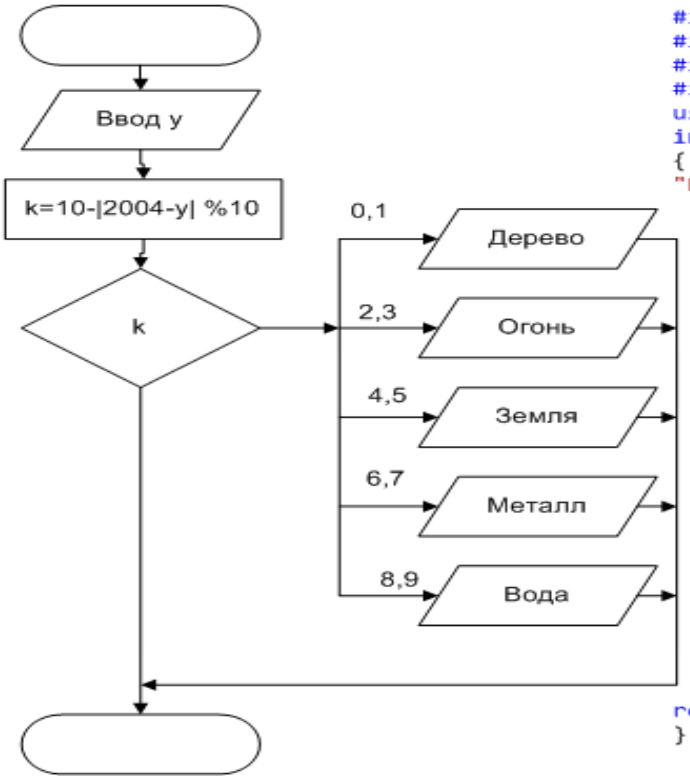
Тестовый пример:

a. при ot=4, вывод «хорошо»;
b. при ot=4, вывод «таких отметок нет»;
c. при ot=3, вывод «удовлетворительно».




```
#include <iostream>
#include <conio.h>
#include <clocale>
using namespace std;
int main()
{ setlocale(LC_ALL, "Russian_Russia.1251");
  int ot;
 cout<<"Введите отметку "; cin>>ot;
 switch (ot) {
   case 5: cout<<"отлично"<<endl; break;
   case 4: cout<<"хорошо"<<endl; break;
   case 3: cout<<"удовлетворительно"<<endl;
           break;
   case 2:
         cout<<"неудовлетворительно"<<endl;
           break  ;
  default: cout<<"таких отметок нет"<<endl;
}
   _getch();
 return 0;
}
```

Пример 2. В восточном календаре за шестидесятилетний календарный цикл каждый год является не только годом какого-нибудь животного, но и относится в какой-нибудь стихии. Каждая стихия охватывает два года подряд. Стихии следуют в следующем порядке: Дерево, Огонь, Земля, Металл, Вода. Составить программу, в которой по году определить, к какой стихии он относится. 2004 год – это первый год стихии дерева.




```
#include <iostream>
#include <conio.h>
#include <clocale>
#include <cmath>
using namespace std;
int main()
{   setlocale(LC_ALL,
"Russian_Russia.1251");
   int y,k;
   cout<<"Введите год"; cin>>y;
   k=10 - abs(2004-y)%10;
   switch (k) {
   case 0:
   case 1:
            cout<<"Дерево"<<endl; break;
   case 2:
   case 3:
            cout<<"Огонь"<<endl; break;
   case 4:
   case 5:
            cout<<"Земля"<<endl; break;
   case 6:
   case 7:
            cout<<"Металл"<<endl; break;
   case 8:
   case 9:
            cout<<"Вода"<<endl; break;
   }
   _getch();
return 0;
}
```

Исходные данные: y – год - целый тип;

Результатом является вывод соответствующей стихии.

Для определения стихии надо найти разность по модулю между введенным годом и 2004 годом. Затем найти остаток от деления на 10 от этой разности. Если остаток 0 или 1, т о это стихия Дерево, если 2 или 3 – Огонь, если 4 или 5 – Земля, если 6 или 7 – Металл, если 8 или 9 – Вода.

Тестовый пример: при y=1966, вывод «Огонь»;

0ператоры break и continue

Оператор break может использоваться для выхода из некоторого цикла до того, как условие проверки получит значение "ложь". Во многом оператор break похож на оператор goto. При выходе из цикла по команде break программа продолжается с оператора, следующего за циклом. В отличие от break , оператор continue приводит к игнорированию всех следующих за ним операторов, однако не препятствует изменению переменной управления циклом или выполнению проверки условия продолжения цикла, т.е. цикл повторяется. Для решения некоторых задач оператор break и continue можно использовать вместе.

exit()

В некоторых случаях программу следует закончить до того, как выполнились все операторы или условия.

Для таких особых случаев в C++ имеется библиотечная функция ***exit( ).*** Эта функция может иметь один целочисленный аргумент, называемый статусом. Операционная система MSDOS интерпретирует значение статуса, равное нулю, как успешное завершение программы, а все ненулевые значения статуса говорят о различного вида ошибках. Для того чтобы можно было бы использовать функцию exit( ), нужно включить в программу заголовочный файл ***stdlib.h.***

Индивидуальные задания

Задание 1. Написать программу с помощью тернарного оператора, согласно номеру своего варианта, а так же сделать блок схему.

1. Заданы три числа. Определить, могут ли они представлять собой стороны одного треугольника. Найти наибольшее и присвоить его значение переменной D.
2. Определить, какому квадранту (т.е. четверти) принадлежит точка M(x, y). Значение x и y выбирают произвольно.
3. Определить, принадлежит ли точка N(3,2; 7,5) треугольнику, образованному осями OX, OY и прямой y = 7,6 - 3,5x
4 Выяснить, лежит ли точка (x, y) внутри круга радиуса R с центром в начале координат, или вне круга, или на окружности.
5. Заданы два отрезка на оси $A_1 = 3,2$; $B_1 = 7,8$; $A_2 = 1$; $B_2 = 5$. Если точка x принадлежит одновременно первому и второму отрезкам, то P = tg x.
6. Заданы две точки: $M_1(8,21; -8,1)$; $M_2(2,14; 15,81)$. Присвоить переменной W значение Q, если расстояние между точками равно нулю и значение P = Q / 31 в противном случае. Здесь Q = 6,28 + sinx – ab / (a + b); a = 12,4; b = 3,62; x выбирают произвольно.
7. Заданы длины трех сторон треугольника a, b, c. Определить, является ли треугольник равнобедренным, равносторонним или разносторонним.
8. Скорость движения объекта изменяется в пределах от V_0 до V_m по формуле $V_t = V_0 + a_t$. Составить алгоритм и программу для вычисления скорости, учитывая, что при $t < 0$ принять $V_t = V_0$, а при $t > m$ принять $V_t = V_m$. Здесь a = 2,5; t = 40; $V_0 = 25$; $V_m = 117$;
9. Прямоугольник задан в плоскости четырьмя точками $A(x_a; y_a)$; $B(x_b; y_b)$; $C(x_c; y_c)$; $D(x_d; y_d)$. Составить алгоритм и программу для определения, принадлежит ли данному прямоугольнику точка M(14; 0,5), если $x_a = x_b = 12$; $x_c = x_d = 20$; $y_a = y_d = 1$; $y_c = y_b = 3$;
10. Заданы длины трех сторон треугольника m, n, h. Определить, является ли треугольник прямоугольным? m = 3; n = 5,1; h = 4.

Задание 2. Написать программу с помощью оператора множественного выбора switch, согласно номеру своего варианта, а так же сделать блок схему.

1. Некоторое предприятие ежесуточно расходует X Квт/час электроэнергии – зимой, Y Квт/час – летом и Z квт/час - весной и осенью. Составить программу, вычисляющую расход электроэнергии R для заданного месяца текущего года.

2. Написать программу, которая бы по введенному месяцу выдает все приходящиеся на этот месяц праздничные дни.

3. Написать программу, которая по введенному номеру месяца выводила название сезона, к которому этот месяц относится, и проверяла правильность введенного месяца.

4. Написать программу, которая бы по введенному номеру единицы измерения (1 – килограмм, 2 – миллиграмм, 3 - грамм, 4 – тонна, 5 – центнер) и массе m выдавала бы соответствующее значение массы в килограммах.

5. Ввести день, месяц и год, проверить правильность введенной даты и вывести дату следующего дня.

6. Дано натуральное число N. Если оно делится на 4, вывести на экран ответ N=4k (где k – соответствующее частное), если остаток от деления на 4 равен 1 – N=4k+1, если остаток от деления на 4 равен 2 – N=4k+2, если остаток от деления на 4 равен 3

– N=4k+3. Например: 12=4*3, 22=4*5+2.

7. Ввести два положительных числа меньшие 10. Сложить эти два числа и написать результат сложения этих чисел - числительным.

8. Вычислить номер дня в не високосном годы по заданным числу и месяцу.

9. Для введенного числа k от 1 до 99 вывести «копеек» учитывая значение k. Например: 20 копеек, но 32 копейки.

10. Вывести по введенному году, какому животному восточного календаря соответствует этот год. Годы внутри цикла носят названия: крыса, бык, тигр, заяц, дракон, змея, лошадь, овца, обезьяна, петух, собака, свинья. Известно, что 2000 год– это год дракона.

Контрольные вопросы

1.Какой тип должно иметь выражение в операторе switch.

2.Если переменная z в условии задачи может принимать 4 значения, в зависимости от того попадает ли x в интервалы меньше 0, от 0 до корня квадратного от 2, от корня квадратного от 2 до 10, больше 10, может ли эта задача быть решена с помощью оператора switch.

3.Что будет, если пропустить оператор break во всех ветках, в первой ветке, в последней ветке.

4. В чем отличие операторов break и continue.
5. Назначение функции exit( )

Лабораторная работа № 6-8. Программирование циклических алгоритмов.

Цель работы: *формирование знаний и умений по работе с операторами языка. Приобретение навыков написания программ с использованием операторов повтора.*

В Си++ существуют три типа операторов цикла: и цикл с параметром, цикл с предусловием и цикл с постусловием, на этой лабораторной работе рассмотрим цикла с параметром.

Формат оператора цикла с параметром:

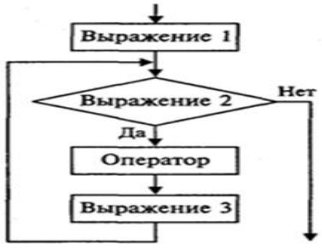
for (выр1; выр2; выр3) оператор;

где выр1- выполняется только один раз в начале цикла. Обычно оно определяет начальное значение параметра цикла (инициализирует параметр цикла).

выр 2 - условие выполнения цикла,

выр3- обычно определяет изменение параметра цикла, оператор -тело цикла, которое может быть простым или составным. В последнем случае используются фигурные скобки.

Алгоритм выполнения цикла for представлен на блок-схеме на рис.



Пример 1.Вычислить значение факториала N!

```
#include <iostream>
using namespace std;
main()
{long int f;
int i,n;
cout<<”введите факториал n=”;
cin>>n;
f=1;
for(i=l; i<=n; i++) f=f*i;
cout<<”\n ”<<n<<”!=”<<f;
return(0);}
```

С помощью цикла for нахождение N! можно организовать следующим образом:

```
f=l;
```

Используя операцию «запятая», можно в выр1 внести инициализацию значений сразу нескольких переменных:

```
for (f=l, i=l; i<=n; i++) f=f*i;
```

Некоторых элементов в операторе for может не быть, однако разделяющие их точки с запятой обязательно должны присутствовать. В следующем примере инициализирующая часть вынесена из оператора for:

```
f=l;  i=l;
for(;i<=f;i++) f=f*i;
```

В языке Си++ оператор for является достаточно универсальным средством для организации циклов в следующем фрагменте программа содержит два вложенных цикла for. В нем запрограммировано получение на экране таблицы умножения.

```
for(x=2; x<=9; x++)
for(y=2; y<=9; y++)
cout<<"\n"<<x<<"*"<<y<<"="<<x*y;
```

На экране будет получен следующий результат:

*2*2=4*
*2*3=6*
. . .
*9*8=72*
*9*9=81*

Оператор continue. Если выполнение очередного шага цикла требуется завершить до того, как будет достигнут конец тела цикла, используется оператор continue. Следующий фрагмент программы обеспечивает вывод на экран всех четных чисел в диапазоне от 1 до 100.

```
for(i=l; i<=100; i++)
{if(i%2) continue; cout<<"\t"<<i;}
```

Для нечетных значений переменной i остаток от деления на 2 будет равен единице, этот результат воспринимается как значение «истина» в условии ветвления, и выполняется оператор continue. Он завершит очередной шаг цикла, выполнение цикла перейдет к следующему шагу.

Оператор goto. Оператор безусловного перехода goto существует в языке Си, как и во всех других языках программирования высокого уровня. Однако с точки зрения структурного подхода к программированию его использование рекомендуется ограничить.

Формат оператора: **goto метка;**

Метка представляет собой идентификатор с последующим двоеточием, ставится перед помечаемым оператором.

Одна из ситуаций, в которых использование goto является оправданным — это необходимость «досрочного» выхода из вложенного цикла. Вот пример такой ситуации:

```
for(...)
{ while (...)
{ for(...)
{... goto exit ...}
}}
exit: cout<<"Выход из цикла";
```

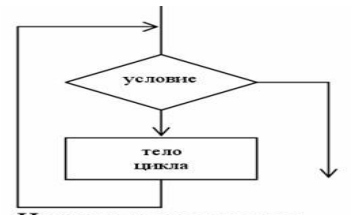
При использовании оператора безусловного перехода необходимо учитывать следующие ограничения:

- нельзя входить внутрь блока извне;
- нельзя входить внутрь условного оператора (if ...else...);
- нельзя входить внутрь переключателя;
- нельзя входить внутрь цикла.

Цикл с предусловием. Формат оператора цикла с предусловием:

while (выражение) оператор;

Цикл повторяет свое выполнение, пока значение выражения отлично от нуля, т. е. заключенное в нем условие цикла истинно. Блок-схема в общем виде выглядит так:



В качестве примера использования оператора цикла рассмотрим программу вычисления факториала целого положительного числа N!.

Пример 2.Вычислить значение факториала N!

```
#include <iostream.h>
using namespace std;
main()
{long int f;
int i,n;
cout<<”введите факториал n=”;
cin>>n;
f=i=1;
while(i<=n)
{f=f*i;
i++;}
cout<<”\n ”<<n<<”!=”<<f;
return(0);}
```

Рассмотрим еще один пример использования оператора цикла while.

Пример 3. Вычислить сумму гармонического ряда: 1+1/2+1/3+... с заданной точностью ε

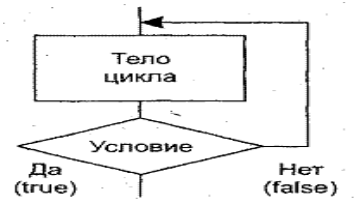
```
#include <iostream.h>
#include < limits.h>
 using namespace std;
main()
{int n=1;
double s=0,eps;
cout<<”введите точность =”;
cin>> eps;
while(1.0/n>eps && n<INT_MAX)
s+=1.0/n++;
cout<<”s=”<<s;
return(0);}
```

Файл limits.h, подключаемый препроцессором, содержит определения предельных констант для целых типов данных. В частности, константа с именем INT_MAX равна максимальному значению типа int в данной реализации компилятора. Если для типа int используется двухбайтовое представление, то INT_MAX=32767. В этом же заголовочном файле определены и другие константы: INT_MIN=-32768; LONG_MAX=2147483647 и т.д.

Цикл с постусловием. Формат оператора цикла с постусловием:

do оператор while (выражение);

Цикл выполняется до тех пор, пока выражение отлично от нуля, т.е. заключенное в нем условие цикла истинно. Выход из цикла происходит после того, как значение выражения станет ложным, иными словами равным нулю. В качестве примера рассмотрим программу. Блок-схема в общем виде выглядит так:



Пример 4. Вычислить значение факториала N!

```
#include <iostream.h>
using namespace std;
main()
{long int f;
int i,n;
cout<<”введите факториал n=”;
cin>>n;
f=i=1;
do
{f=f*i;
i++;}
while(i<=n);
cout<<”\n ”<<n<<”!=”<<f;
return(0);}
```

Индивидуальные задания

Задание 1. Написать программу согласно номеру своего варианта, сделать блок схему.

1. Найти произведение всех целых чисел от a до 20 (значение a вводится с клавы; a < 20).
2. Найти произведение всех целых чисел от a до b (значения a и b вводятся с клавиатуры; b > a).
3. Напечатать таблицу перевода 1, 2, ... 20 долларов США в рубли по текущему курсу (значение курса вводится с клавиатуры).
4. Напечатать таблицу перевода расстояний в дюймах в сантиметры для значений 10, 11, ... 22 дюйма (1 дюйм = 25.4 мм).
5. Напечатать таблицу стоимости 100, 200, 300, …, 2000 г конфет (стоимость 1 кг конфет вводится с клавиатуры).
6. Напечатать таблицу стоимости 50, 100, 150, ... 1000 г сыра (стоимость 1 кг сыра вводится с клавиатуры).
7. Найти произведение целых положительных четных чисел от 1 до n (значение n вводится с клавы).
8. Найти произведение целых положительных чисел от 1 до n, кратных 5 (значение n вводится с клавиатуры).
9. Найти среднее арифметическое всех целых чисел от 100 до b (значение b вводится с клавиатуры; b > 100).
10. Найти среднее арифметическое целых положительных нечетных чисел от 1 до n (значение n вводится с клавиатуры).
11. Найти среднее арифметическое целых положительных чисел от a до b и кратных 3 (значения a и b вводятся с клавиатуры; b > a).
12. Найти сумму всех целых чисел a до 500 (значение a вводится с клавиатуры; a < 500).
13. Найти сумму квадратов всех целых чисел от 1 до n (значение n вводится с клавиатуры; 0 < n < 100).
14. Найти сумму целых положительных нечетных чисел от 1 до n (значение n вводится с клавы).

Задание 2. Написать программу согласно номеру своего варианта для вычисления системы уравнений, сделать блок схему.

2. Вычислить функции $Y=X^2+Z$ для $0\le X\le 4$ с шагом 0,5 и $1\le Z\le 10$ с шагом 2.
3. Вычислить функции $Y=X^2+Z$ для $0\le X\le 5{,}6$ с шагом 0,2 и $1\le Z\le 7{,}5$ с шагом 1,5.
4. Известен начальный банковский вклад X и годовой процент P. Выяснить через сколько лет L вклад достигнет величины Y.
5. Вычислить значения функции $Y=X^2+Z$ для $0\le X\le 4$ и $0\le Z\le 10$ с шагом 1. В этой функции два аргумента. Решение здесь очень простое. Строятся два цикла – внешний (по X) и внутренний (по Z). В данном случае безразлично, какой параметр поместить снаружи, а какой внутри. Здесь на одно изменение переменной X произойдет 11 изменений Z.
6. В ведомости указана зарплата, выплаченная каждому из сотрудников фирмы за месяц. Определить общую сумму выплаченных по ведомости денег. Количество сотрудников

фирмы вводиться с клавиатуры.
7. Напечатать таблицу умножения на число n (значение n вводится с клавиатуры; 1 < n < 9).
8. Напечатать третьи степени всех целых чисел от a до 50 (значение a вводится с клавы; a < 50).
9. Напечатать таблицу соответствия между весом в фунтах и весом в килограммах для значений 1, 2, ..., 10 фунтов (1 фунт = 453 г).
10. Одна штука некоторого товара стоит 20,4 руб. Напечатать таблицу стоимости 2, 3, …, 20 штук этого товара.
11. Дана последовательность ненулевых целых чисел. Определить, сколько раз в этой последовательности меняется знак. Например, в последовательности 10, -4, 12, 56, -4 знак меняется 3 раза.

Задание 3. Написать программу вывода на экран таблицы значений функции y(x) для x, изменяющегося от a = 0,1 до b = 1,2 с шагом h = 0,1, согласно номеру своего варианта, сделать блок схему.

1. $y = \sum_{n=1}^{20} \frac{x^{n-1}}{2n+1}$.

2. $y = \sum_{n=0}^{20} \frac{(2x)^n}{n+1}$.

3. $y = \sum_{n=1}^{20} \frac{x^{n-1}}{\sin(nx)}$.

4. $y = \sum_{n=1}^{20} \frac{n^2+1}{n} \left(\frac{x}{2}\right)^n$.

5. $y = \sum_{n=1}^{20} \frac{\cos\left(n \cdot \frac{\pi}{4}\right)}{n+1} x^n$.

6. $y = \sum_{n=1}^{20} \frac{x^{2n-2}}{2n+1}$.

7. $y = \sum_{n=0}^{20} \frac{x^{2n}}{\cos(nx)}$.

8. $y = \sum_{n=1}^{20} \frac{2n^2+1}{2n} x^{2n-2}$.

9. $y = \sum_{n=1}^{20} \frac{2n+1}{\sin(nx)} x^{n-1}$.

10. $y = \sum_{n=0}^{20} \frac{\cos^n(x)}{2n+1}$.

Контрольные вопросы

1. Как можно выйти досрочно из цикла.
2. Если начальной значение счетчика окажется меньше конечного значения, будет ли выполняться тело цикла хотя бы один раз.
3. Формат оператора for.
4. Формат операторов цикла while и do-while.

Литература

1. Культин Н.Б. C/C++ в задачах и примерах. – СПб.: БХВ – Петербург, 2018.
2. Павловская Т.А. C/C++. Программирование на языке высокого уровня: Учебник для вузов. – СПБ.: Питер,2014.
3. https://metanit.com/cpp/tutorial/