

Федеральное агентство связи
Уральский технический институт связи и информатики (филиал)
ФГБОУ ВО «Сибирский государственный университет
телекоммуникаций и информатики» в г. Екатеринбурге
(УрТИСИ СибГУТИ)



УТВЕРЖДАЮ
Директор УрТИСИ СибГУТИ
_____ Е.А. Минина
« ____ » _____ 20__ г.

Практикум программирования на языке C++

УЧЕБНАЯ ПРАКТИКА ПО МДК 01.01 "СИСТЕМНОЕ ПРОГРАММИРОВАНИЕ"

для специальности:

09.02.03 «Программирование в компьютерных системах»

Практическая работа №1

Екатеринбург
2020

Введение	3
Среда программирования Visual C++	3
1.1. Общий вид окна	3
1.2. Создание консольного приложения и работа с ним.....	4
1.3. Компиляция и запуск проекта	5
1.4. Отладка программы.....	6
1.5. Создание рабочего пространства для нескольких проектов	6
ЛПрактическая работа №1	7
Выполнение программы простой структуры. Вычисление выражений с использованием стандартных функций	7
1. Цель задания:	7
2. Теоретические сведения	7
2.1. Структура программы на C++	7
2.2. Элементы языка C/C++	9
2.3. Константы в C/C++.....	9
2.3. Типы данных в C++	10
2.4. Переменные.....	11
2.5. Операции	11
2.6. Выражения	13
2.7. Ввод и вывод данных	13
3. Постановка задачи	14
4. Варианты	15
5. Методические указания	20
6. Содержание отчета	21

Введение

Для того, чтобы научиться программировать, в первую очередь, надо научиться строить и записывать алгоритмы решаемых задач. Алгоритм – это точное предписание, определяющее вычислительный процесс, идущий от изменяемых начальных данных к конечному результату, т. е. это рецепт достижения какой-либо цели. Совокупность средств и правил для представления алгоритма в виде пригодном для выполнения вычислительной машиной называется языком программирования, алгоритм, записанный на этом языке – программой. Для записи алгоритмов существуют разные формы:

- 1) словесное описание (псевдокоды),
- 2) графическое описание (блок-схемы),
- 3) алгоритмические языки.

Для того чтобы составить программу желательно выполнить по порядку следующие этапы:

- 1) Определить исходные данные задачи и результаты, которые должны быть получены, а также формулы, связывающие исходные данные и результаты.
- 2) Составить алгоритм в виде блок-схемы, с помощью которого можно от исходных данных перейти к результатам.
- 3) Записать алгоритм на требуемом языке программирования (т. е. каждому блоку блок-схемы надо поставить в соответствие оператор языка программирования).
- 4) Выполнить программу, используя какую-то систему программирования.
- 5) Выполнить отладку и тестирование программы. При выполнении программы могут возникать ошибки трех типов:

Самыми опасными являются именно семантические ошибки, т. к. их достаточно сложно обнаружить. Программа будет работать, но неправильно, причем, ошибки в ее работе могут возникать не все время, а только при каких-то определенных наборах исходных данных. Для обнаружения таких ошибок выполняется тестирование программы. Набор исходных данных, для которых известен результат, называется тестом. Если результаты работы теста не совпадут с известным значением, значит, в программе имеется ошибка. Тест, выявивший ошибку, считается успешным. Отладка программы заканчивается, когда достаточное количество тестов будет выполнено неуспешно. Самым распространенным критерием для определения количества неуспешных тестов является тестирование ветвей: набор тестов в совокупности должен обеспечить прохождение каждой ветви не менее одного раза.

Начинающие программисты должны обязательно выполнять все указанные этапы. В дальнейшем этапы 2-3 можно объединить в один и сразу записывать программу на требуемом языке программирования.

В качестве изучаемого языка программирования выбран C++, т. к. этот язык позволяет выработать алгоритмическое мышление, стоит короткую программу, продемонстрировать основные приемы алгоритмизации.

Среда программирования Visual C++ 6.0

1.1. Общий вид окна

Проект (project) – это набор файлов, которые совместно используются для создания одной программы.

Рабочее пространство (workspace) может включать в себя несколько проектов.

После запуска VC++ 6.0 на экране появится окно (рис. 1).

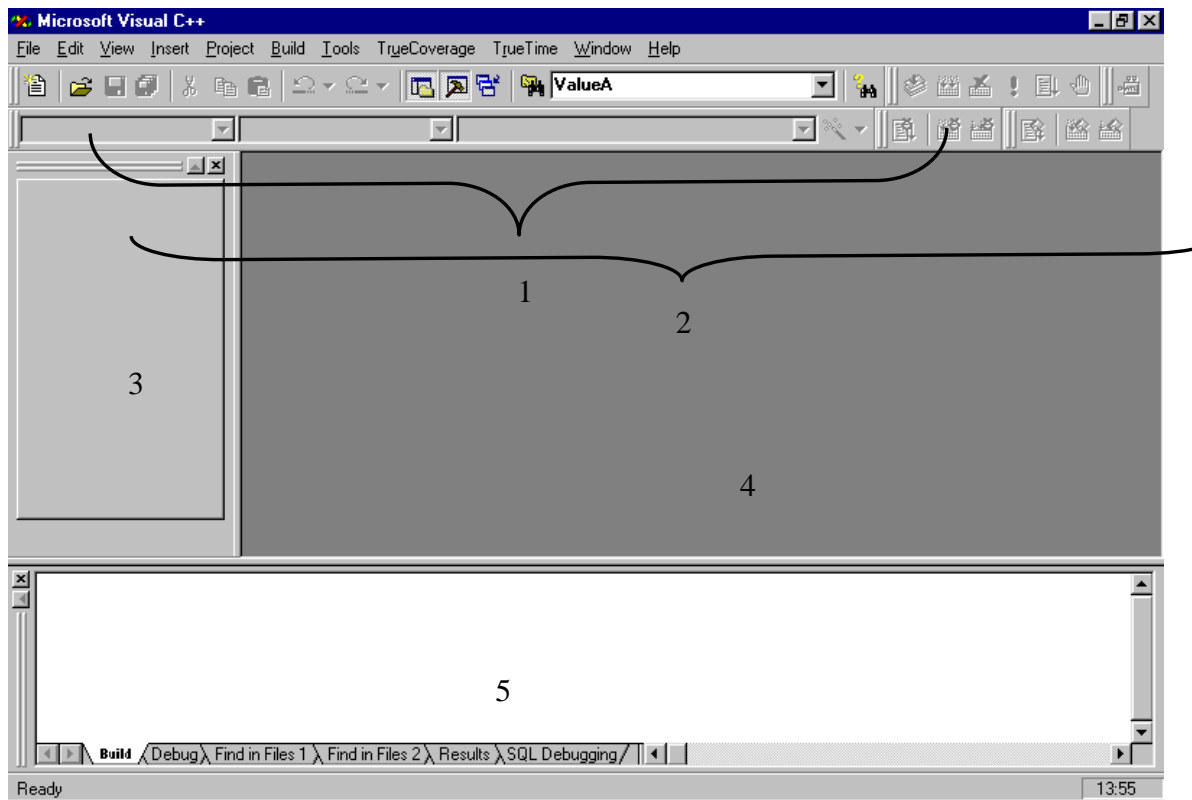


Рис. 1. Окно VC++ 6.0.

Окно содержит:

- Главное меню (1) – список основных команд VC++;
- Панель инструментов (2) - панель с кнопками команд Visual C++;
- Панель рабочего пространства Workspace (3) - содержит две вкладки:
 - ClassView – отображает список классов в проекте,
 - FileView – отображает список файлов, входящих в проект.
- Окно для редактирования кодов (4) – окно с текстом программы;
- Выходную панель результатов компиляции (5) - окно для вывода сообщений в процессе компиляции или отладки, показывает текущую стадию компиляции, список ошибок и предупреждений и их количество.

1.2. Создание консольного приложения и работа с ним

Консольное приложение – это приложение, которое с точки зрения программиста является программой DOS, но может использовать всю доступную оперативную память (если каждый элемент данных программы не будет превышать 1 Мб). Этот тип приложения запускается в особом окне, которое называется “Окно MS-DOS”. На примере консольных приложений прослеживаются этапы развития VC++ при переходе от одной версии к другой.

Каждое приложение, разрабатываемое как отдельный проект в среде VC++6.0, нуждается в том, чтобы ему соответствовало свое собственное рабочее пространство. Рабочее пространство включает в себя те папки, в которых будут храниться файлы, содержащие информацию о конфигурации проекта. Для того чтобы создать новое пространство для проекта, надо выполнить следующие действия:

1. В линейке меню нажать на меню **File**.
2. Выбрать пункт **New** или нажать **Ctrl+N**.
3. Появится окно **New**. В нем содержится четыре вкладки: Files, Projects, Workspaces, Other Documents. Выбрать вкладку Projects.

4. Из списка возможных проектов выбрать **Win32 Console Application** для создания приложения DOS.
5. В поле **Project name** ввести имя проекта.
6. В поле **Location** ввести путь для размещения каталога проекта, или, нажав на кнопку справа [...], выбрать нужную директорию.
7. Должен быть установлен флажок **Create New Workspace**. Тогда будет создано новое рабочее окно. Нажать кнопку **ОК**
8. Установить один из флажков:
 - **An empty project** – создается пустой проект, не содержащий заготовок для файлов;
 - **A simple application** – создается простейшая заготовка, состоящая из заголовочного файла StdAfx.h, файла StdAfx.cpp и файла реализации;
 - **A “Hello World” application** и **An application that supports MFC** являются демонстрационными и разными способами демонстрируют вывод на экран строки символов.

Нажать кнопку **Finish**. Появится информация о созданном проекте содержащая: тип проекта, некоторые особенности и директорию.

После создания проекта в него необходимо записать код программы. При этом можно создать новый файл или добавить в проект существующий файл.

Для создания нового файла надо выполнить следующие действия:

1. Выбрать меню **File > New** или **Project > Add to Project > New**.
2. Открыть вкладку **Files**.
3. Выбрать **C++ Source File**.
4. Чтобы создаваемый файл был автоматически присоединен к проекту, необходимо установить флаг **Add to project**.
5. В поле **Filename** ввести имя файла.
6. В поле **Location** указать путь для создания файла.
7. Нажать **ОК**.

Для добавления существующего файла надо:

1. Выбрать в меню **File > Add to Project > Files**
2. Указать полное имя файла, который нужно присоединить

Для открытия существующего проекта надо:

1. Выбрать меню **File > Open Workspace**
2. Указать файл с расширением **.dsw**

Для сохранения текущего проекта надо выбрать в главном меню **File > Save Workspace**.

Для закрытия текущего проекта надо выбрать в главном меню **File > Close Workspace**.

После создания или открытия проекта в окне **Workspace** появится или список классов, или список файлов входящих в проект. В зависимости от типа проекта, он будет или пустой, или содержать изначально некоторые файлы, присущие данному типу. Проект приложения для DOS изначально пустой. В него можно добавить новые файлы или присоединить уже существующие.

1.3. Компиляция и запуск проекта

Для компиляции проекта надо выбрать в главном меню **Build > Build <имя проекта>** или нажать клавишу F7.

Visual C++ 6.0 откомпилирует исходные файлы и создаст соответствующие файлы с расширением **.obj**. Затем эти файлы соединяются в исполняемый файл. Весь процесс компиляции и создания исполняемого файла отображается в окне Output, вкладка Build. После компиляции файла его можно запустить.

Для запуска исполняемого файла надо выбрать в главном меню **Build > Execute <имя файла>.exe** или нажмите клавиши **Ctrl+F5** . Если файл был создан, то он запустится. Для повторного запуска файла не нужно его снова компилировать. Но если в программу были внесены изменения, то перед запуском необходимо выполнить компиляцию. Выполняется именно файл с расширением .exe, а не текущий проект, т.е. в процессе запуска компиляции не происходит.

1.4. Отладка программы

Для отладки программы используется команда главного меню **Build>Start Debug> Step Into** – отладка с заходом в функции, которая начинается с первой строки функции `main` или **Build>Start Debug> Run to Cursor** – выполнение программы до курсора, т. е. отладка начинается с той строки, в которой установлен курсор. После выполнения этой команды выполнение программы происходит в режиме отладчика. Переход к следующей строке программы можно выполнять с помощью команды **Step Into (F11)** (с заходом во все вызываемые функции) или с помощью команды **Step over (F10)** (без захода в вызываемые функции). Выход из функции нижнего уровня выполняется командой **Step Out (Shift+F11)**. Текущие значения переменных можно просматривать:

- 1) в специальных окнах **Watch** (отображает значения всех используемых переменных) и **Value** (отображает значения заданных пользователем переменных);
- 2) при наведении курсора мышки на переменную отображается текущее значение этой переменной.

1.5. Создание рабочего пространства для нескольких проектов

Несколько проектов можно объединить в одно рабочее пространство с помощью команды **Project/Insert Project into Workspace**. Активный проект, т. е. тот, который будет выполняться, устанавливается с помощью команды **Project/Set Active Project**. Активный проект надо отметить галочкой.

Практическая работа №1

Выполнение программы простой структуры. Вычисление выражений с использованием стандартных функций

1. Цель задания:

- 1) Выполнение простой программы в системе программирования VC++6.0
- 2) Приобретение навыков в записи выражений на языке C++ и использование стандартных функций.

2. Теоретические сведения

2.1. Структура программы на C++

Программа на языке Си имеет следующую структуру:

#директивы препроцессора

.....

#директивы препроцессора

функция а ()

операторы

функция в ()

операторы

void main () //функция, с которой начинается выполнение программы

операторы

описания

присваивания

функция

пустой оператор

составной

выбора

циклов

перехода

Директивы препроцессора управляют преобразованием текста программы до ее компиляции. Исходная программа, подготовленная на C++ в виде текстового файла, проходит 3 этапа обработки:

- 1) препроцессорное преобразование текста;
- 2) компиляция;
- 3) компоновка (редактирование связей или сборка).

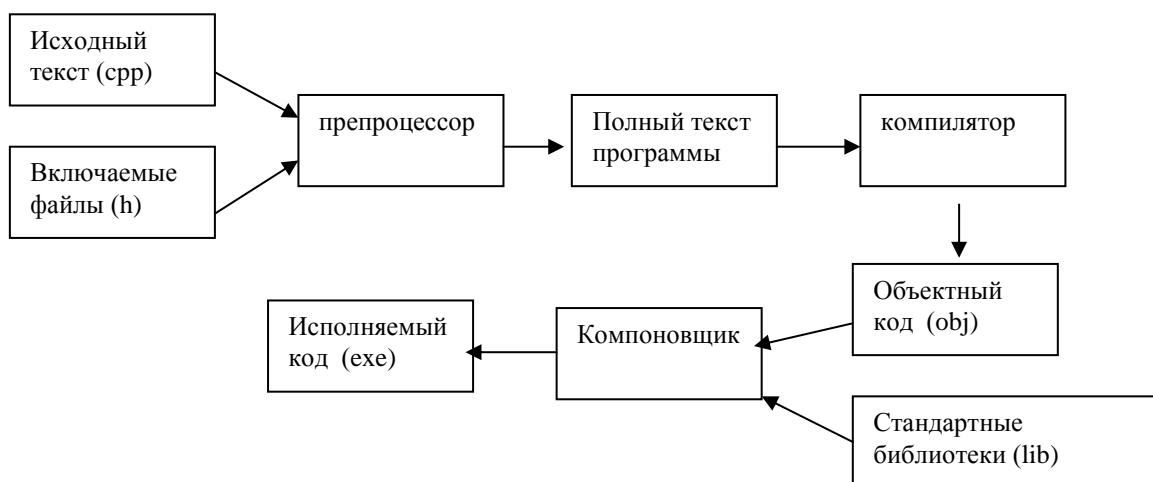


Рис. 2. Обработка C++ программы

После этих трех этапов формируется исполняемый код программы. Задача препроцессора – преобразование текста программы до ее компиляции. Правила препроцессорной обработки определяет программист с помощью директив препроцессора. Директива начинается с #.

#define – указывает правила замены в тексте.

#include<имя заголовочного файла> – директива предназначена для включения в текст программы текста из каталога заголовочных файлов, поставляемых вместе со стандартными библиотеками. Каждая библиотечная функция C имеет соответствующее описание в одном из заголовочных файлов. Список заголовочных файлов определен стандартом языка. Употребление директивы include не подключает соответствующую стандартную библиотеку, а только позволяют вставить в текст программы описания из указанного заголовочного файла. Если используется заголовочный файл из стандартной библиотеки, то его имя заключают в угловые скобки. Если используется заголовочный файл, который находится в текущем каталоге проекта (он может быть создан разработчиком программы), то его имя заключается в кавычки. Подключение кодов библиотеки осуществляется на этапе компоновки, т. е. после компиляции. Хотя в заголовочных файлах содержатся все описания стандартных функций, в код программы включаются только те функции, которые используются в программе.

После выполнения препроцессорной обработки в тексте программы не остается ни одной препроцессорной директивы.

Программа представляет собой набор описаний и определений, и состоит из набора функций. Среди этих функций всегда должна быть функция с именем main. Без нее программа не может быть выполнена. Перед именем функции помещаются сведения о типе возвращаемого функцией значения (тип результата). Если функция ничего не возвращает, то указывается тип void: void main(). Каждая функция, в том числе и main, должна иметь список параметров. Список может быть пустым, тогда он указывается как (void) (слово void может быть опущено: ()).

За заголовком функции размещается тело функции. Тело функции – это последовательность определений, описаний и исполняемых операторов, заключенных в фигурные скобки. Каждое определение, описание или оператор заканчивается точкой с запятой.

Определения – вводят объекты (объект – это именованная область памяти, частный случай объекта – переменная), необходимые для представления в программе обрабатываемых данных. Примерами являются

```
const int y = 10 ; //именованная константа
```


`float x ; //переменная`

Описания – уведомляют компилятор о свойствах и именах объектов и функций, описанных в других частях программы.

Операторы – определяют действия программы на каждом шаге ее исполнения.

2.2. Элементы языка C/C++

- 1) Алфавит языка который включает
 - прописные и строчные латинские буквы и знак подчеркивания;
 - арабские цифры от 0 до 9;
 - специальные знаки `{ } , | [] () + - / % * . \ ' : ; & ? < > = ! # ^`
 - пробельные символы (пробел, символ табуляции, символы перехода на новую строку).
- 2) Из символов формируются лексемы языка:
 - *Идентификаторы* – имена объектов C/C++-программ. В идентификаторе могут быть использованы латинские буквы, цифры и знак подчеркивания. Прописные и строчные буквы различаются, например, `PROG1`, `prog1` и `Prog1` – три различных идентификатора. Первым символом должна быть буква или знак подчеркивания (но не цифра). Пробелы в идентификаторах не допускаются.
 - *Ключевые (зарезервированные) слова* – это слова, которые имеют специальное значение для компилятора. Их нельзя использовать в качестве идентификаторов.
 - *Знаки операций* – это один или несколько символов, определяющих действие над операндами. Операции делятся на унарные, бинарные и тернарную по количеству участвующих в этой операции операндов.
 - *Константы* – это неизменяемые величины. Существуют целые, вещественные, символьные и строковые константы. Компилятор выделяет константу в качестве лексемы (элементарной конструкции) и относит ее к одному из типов по ее внешнему виду.
 - *Разделители* – скобки, точка, запятая пробельные символы.

2.3. Константы в C/C++

Константа – это лексема, представляющая изображение фиксированного числового, строкового или символьного значения. Константы делятся на 5 групп:

- целые;
- вещественные (с плавающей точкой);
- перечислимые;
- символьные;
- строковые.

Компилятор выделяет лексему и относит ее к той или другой группе, а затем внутри группы к определенному типу по ее форме записи в тексте программы и по числовому значению.

Целые константы могут быть десятичными, восьмеричными и шестнадцатеричными.

Название	Определение	Примеры
Десятичная константа	Последовательность десятичных цифр, начинающаяся не с 0, если это число не 0	8, 0, 192345
Восьмеричная константа	Последовательность восьмеричных цифр,	026, 034, 017

	которым предшествует 0.	
Шестнадцатеричная константа	Последовательность шестнадцатеричных цифр, которым предшествуют символы 0x или 0X	0xA, 0X00F, 0x123

Вещественные константы могут иметь две формы представления: с фиксированной точкой и с плавающей точкой.

Название	Вид	Примеры
Константы с фиксированной точкой	[цифры].[цифры]	5.7, .0001, 41.
Константа с плавающей точкой	[цифры][.][цифры]E[e[+ -] [цифры]	0.5e5, .11e-5, 5E3

Перечислимые константы вводятся с помощью ключевого слова `enum`. Это обычные целые константы, которым приписаны уникальные и удобные для использования обозначения.

```
enum {one=1, two=2, three=3, four=4};
enum {zero, one, two, three};
enum {ten=10, three=3, four, five, six};
enum {Sunday, Monday, Tuesday, Wednesday, Thursday, Friday, Saturday};
```

Символьные константы – это один или два символа, заключенные в апострофы. Символьные константы, состоящие из одного символа, имеют тип `char` и занимают в памяти один байт, символьные константы, состоящие из двух символов, имеют тип `int` и занимают два байта. Последовательности, начинающиеся со знака `\`, называются управляющими, они используются:

- для представления символов, не имеющих графического отображения, например:
 - `\a` – звуковой сигнал,
 - `\b` – возврат на один шаг,
 - `\n` – перевод строки,
 - `\t` – горизонтальная табуляция;
- для представления символов: `\, ', ?, " (\, \', \?, \")`;
- для представления символов с помощью шестнадцатеричных или восьмеричных кодов (`\073, \0xF5`);

Строчковая константа – это последовательность символов, заключенная в кавычки. Внутри строк также могут использоваться управляющие символы. Например:

```
"\nНовая строка",
"\n"Алгоритмические языки программирования\"".
```

2.3. Типы данных в C++

Типы C++ можно разделить на простые и составные. К простым типам относят типы, которые характеризуются одним значением. В языке C++ определено 6 простых типов данных:

<pre>int (целый) char (символьный) wchar_t (расширенный символьный) (C++) bool (логический) (C++) float (вещественный)</pre>	<pre>} } }</pre>	<pre>целочисленные с плавающей точкой</pre>
--	------------------	---

double (вещественный с двойной точностью)

Существует 4 спецификатора типа, уточняющих внутреннее представление и диапазон стандартных типов

short (короткий)

long (длинный)

signed (знаковый)

unsigned (беззнаковый)

Тип данных	Определение	Размер	Диапазон	
(signed) char	Значениями являются элементы конечного упорядоченного множества символов. Каждому символу ставится в соответствие число, которое называется кодом символа.	1 байт	-128..127	
unsigned char			0..255	
wchar_t	Значениями являются элементы конечного упорядоченного множества символов в кодировке Unicode	2 байта	0..65535	
(signed) int	Значениями являются целые числа.	4 байта (для 32-разрядного МП)	-2147483648 ... +2147483647.	
(signed) long (int)			2 байта (для 32-разрядного МП)	0...+4294967 295.
unsigned int				-32768 ... +32767
unsigned long (int)		0 ... 65536;		
(signed) short int		2 байта (для 32-разрядного МП)	-32768 ... +32767	
unsigned short int			0 ... 65536;	
bool	Данные этого типа могут принимать значения true и false.	1 байт	false, true	
float	Значениями являются вещественные числа	4 байта	3.4E-38..3.4E+38	
double		8 байт	1.7E-308 ..1.7E+308	
long double		10 байт	3.4E-4932..1E+4932	

2.4. Переменные

Переменная в C++ – именованная область памяти, в которой хранятся данные определенного типа. У переменной есть имя и значение. Имя служит для обращения к области памяти, в которой хранится значение. Перед использованием любая переменная должна быть описана.

```
int a; float x;
```

2.5. Операции

В соответствии с количеством операндов, которые используются в операциях они делятся на унарные (один операнд), бинарные (два операнда) и тернарную (три операнда).

Операция	Описание
Унарные операции	

++	Увеличение на единицу: префиксная операция - увеличивает операнд до его использования, постфиксная операция увеличивает операнд после его использования.
--	Уменьшение на единицу: префиксная операция - уменьшает операнд до его использования, постфиксная операция уменьшает операнд после его использования.
sizeof	вычисление размера (в байтах) для объекта того типа, который имеет операнд
-	Унарный минус
+	Унарный плюс
!	Логическое отрицание (НЕ). В качестве логических значений используется 0 (false) - ложь и не 0 (true) - истина, отрицанием 0 будет 1, отрицанием любого ненулевого числа будет 0.
&	Получение адреса операнда
*	Получение значения, находящегося по указанному адресу (разыменованное)
new	Выделение памяти
delete	Освобождение памяти
(type)	Преобразование типа
Бинарные операции	
Мультипликативные	
*	умножение операндов арифметического типа
/	деление операндов арифметического типа (если операнды целочисленные, то выполняется целочисленное деление)
%	получение остатка от деления целочисленных операндов
Аддитивные	
+	бинарный плюс (сложение арифметических операндов)
-	бинарный минус (вычитание арифметических операндов)
Операции сравнения	
<	меньше, чем
<=	меньше или равно
>	больше
>=	больше или равно
=	равно
!=	не равно
Логические о	
&&	конъюнкция (И) целочисленных операндов или отношений, целочисленный результат ложь(0) или истина(не 0)
	дизъюнкция (ИЛИ) целочисленных операндов или отношений, целочисленный результат ложь(0) или истина(не 0)
Тернарная	
?:	Условная операция в ней используется три операнда. Выражение1 ? Выражение2 : Выражение3; Первым вычисляется значение выражения1. Если оно истинно, то вычисляется значение выражения2, которое становится результатом. Если при вычислении выражения1 получится 0, то в качестве результата берется значение выражения3. Например: x<0 ? -x : x ; //вычисляется абсолютное значение x.

Присваивание	
=	присваивание
*=	умножение с присваиванием (мультипликативное присваивание)
/=	деление с присваиванием
%=	деление с остатком с присваиванием
+=	сложение с присваиванием
-=	вычитание с присваиванием

Приоритеты операций.

Ранг	Операции
1	() [] -> .
2	! ~ - ++ -- & * (тип) sizeof тип()
3	* / % (мультипликативные бинарные)
4	+ - (аддитивные бинарные)
5	< > <= >= (отношения)
6	== != (отношения)
7	&& (конъюнкция «И»)
8	(дизъюнкция «ИЛИ»)
9	?: (условная операция)
10	= *= /= %= -= &= ^= = <<= >>= (операция присваивания)
11	, (операция запятая)

2.6. Выражения

Из констант, переменных, разделителей и знаков операций можно конструировать выражения. Каждое выражение представляет собой правило вычисления нового значения. Каждое выражение состоит из одного или нескольких операндов, символов операций и ограничителей. Если выражение формирует целое или вещественное число, то оно называется арифметическим. Пара арифметических выражений, объединенная операцией сравнения, называется отношением. Если отношение имеет ненулевое значение, то оно – истинно, иначе – ложно.

2.7. Ввод и вывод данных

В языке C/C++ нет встроенных средств ввода и вывода – он осуществляется с помощью функций, типов и объектов, которые находятся в стандартных библиотеках. Существует два основных способа: функции C и объекты C++.

Для ввода/вывода данных в стиле C используются функции, которые описываются в библиотечном файле `stdio.h`.

- `printf` (форматная строка, список аргументов);
форматная строка – строка символов, заключенных в кавычки, которая показывает, как должны быть напечатаны аргументы. Например:
`printf ("Значение числа Пи равно %f\n", pi);`

Форматная строка может содержать:

- символы печатаемые текстуально;
- спецификации преобразования;
- управляющие символы.

Каждому аргументу соответствует своя спецификация преобразования:

- `%d, %i` – десятичное целое число;
- `%f` – число с плавающей точкой;
- `%e, %E` – число с плавающей точкой в экспоненциальной форме;

%u – десятичное число в беззнаковой форме;
%c – символ;
%s – строка.

В форматную строку также могут входить управляющие символы:

\n – управляющий символ новая строка;
\t – табуляция;
\a – звуковой сигнал и др.

Также в форматной строке могут использоваться модификаторы формата, которые управляют шириной поля, отводимого для размещения выводимого значения. Модификаторы – это числа, которые указывают минимальное количество позиций для вывода значения и количество позиций для вывода дробной части числа:

%[-]m[.p]C, где

– – задает выравнивание по левому краю,
m – минимальная ширина поля,
p – количество цифр после запятой для чисел с плавающей точкой и минимальное количество выводимых цифр для целых чисел (если цифр в числе меньше, чем значение p, то выводятся начальные нули),
C – спецификация формата вывода.

```
printf("\nСпецификации формата:\n%10.5d – целое, \n \\ %10.5f –  
с плавающей точкой\n %10.5e – \\  
в экспоненциальной форме\n%10s – строка", 10, 10.0, 10.0, "10");
```

Будет выведено:

Спецификации формата:
00010 – целое
10.00000 – с плавающей точкой
1.00000e+001 – в экспоненциальной форме
10 – строка.

- scanf (форматная строка, список аргументов);
в качестве аргументов используются адреса переменных. Например:

```
scanf(" %d%f ", &x, &y);
```

При использовании библиотеки классов C++, используется библиотечный файл `iostream.h`, в котором определены стандартные потоки ввода данных от клавиатуры `cin` и вывода данных на экран `cout`, а также соответствующие операции

<< – операция записи данных в поток;
>> – операция чтения данных из потока.

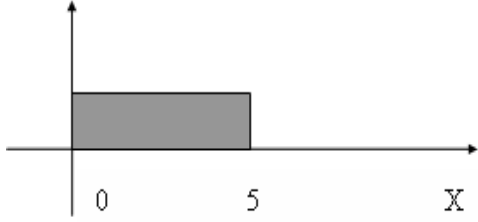
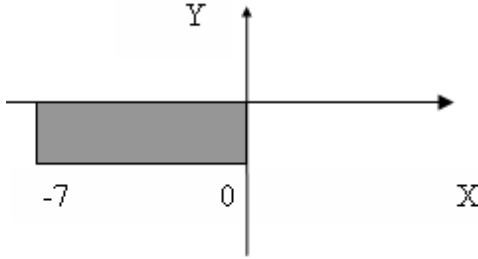
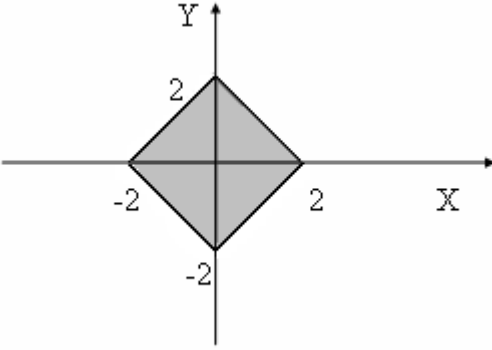
```
#include <iostream.h>;  
...  
cout << "\nВведите количество элементов: ";  
cin >> n;
```

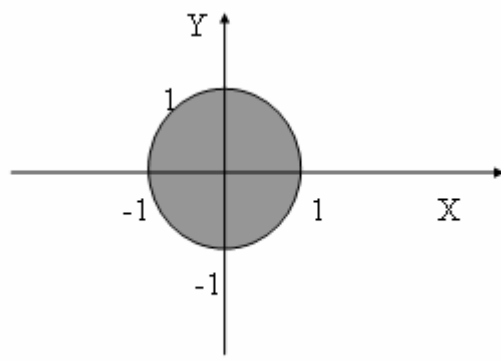
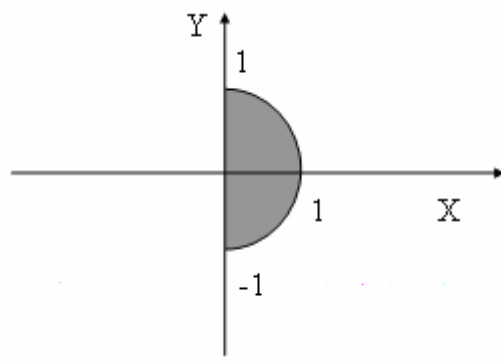
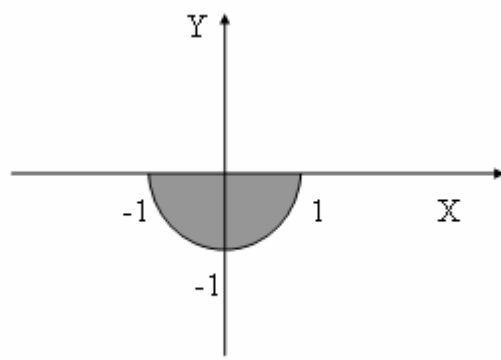
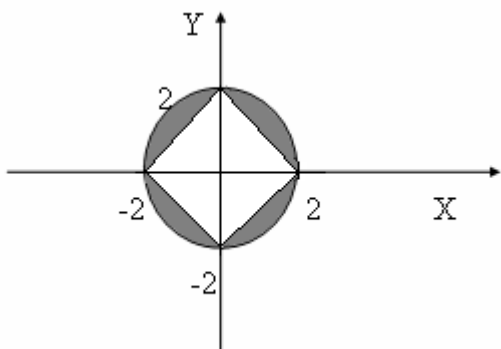
3. Постановка задачи

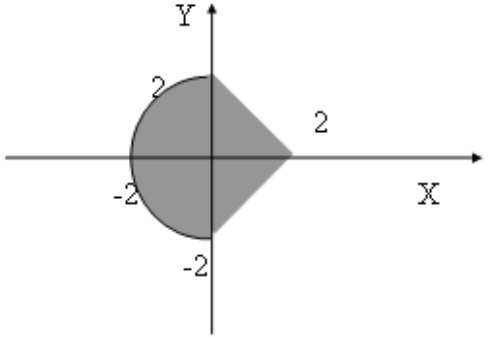
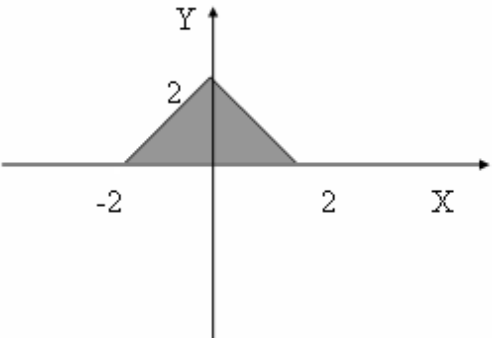
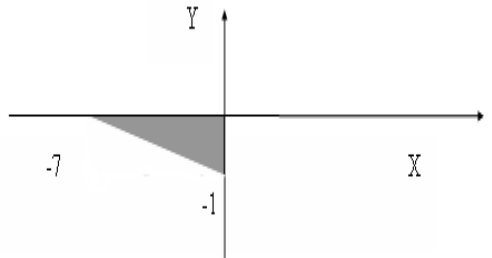
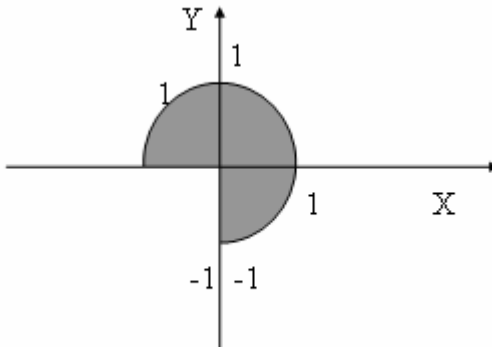
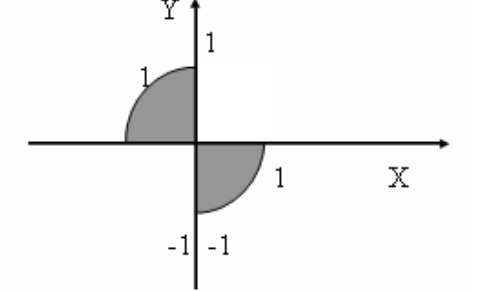
1. Для задачи 1 определить тип заданных выражений и найти их значения.

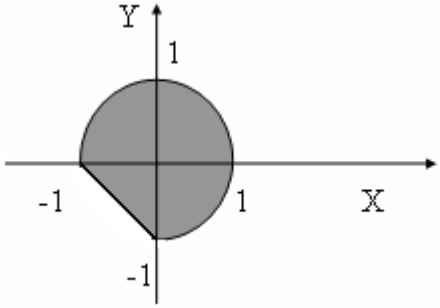
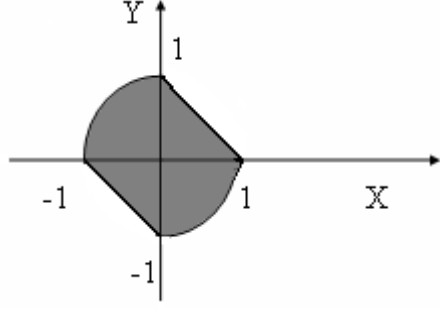
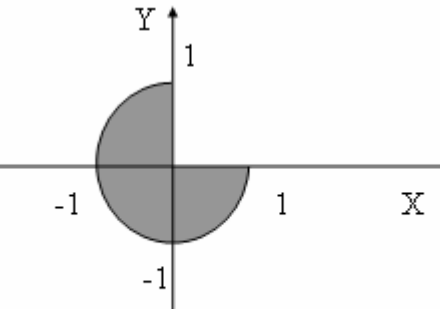
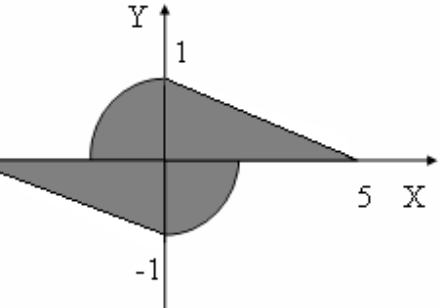

2. Составить систему тестов и вычислить полученное выражение для нескольких значений X, определить при каких X выражение не может быть вычислено.
3. Для задачи 2 записать выражение, зависящее от координат точки X1 и Y1 и принимающее значение TRUE, если точка принадлежит заштрихованной области, и FALSE, если не принадлежит.
4. Составить систему тестов и вычислить полученное выражение для нескольких точек, принадлежащих и не принадлежащих заштрихованной области.
5. Для задачи 3 вычислить значение выражения, используя различные вещественные типы данных (float и double).
6. Объяснить полученные результаты.
7. Результаты всех вычислений вывести на печать.

4. Варианты

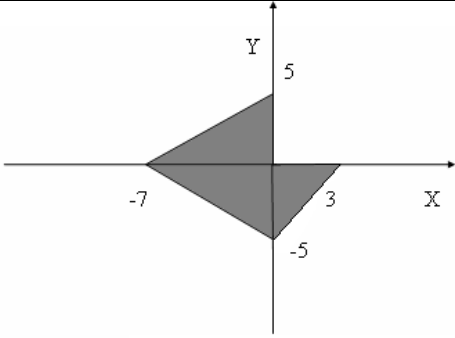
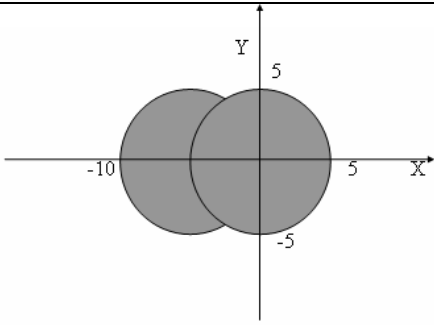
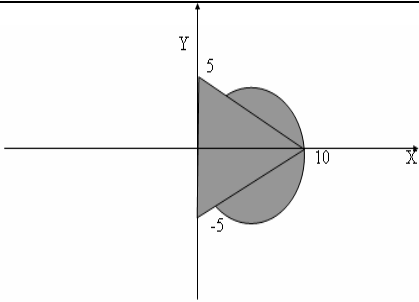
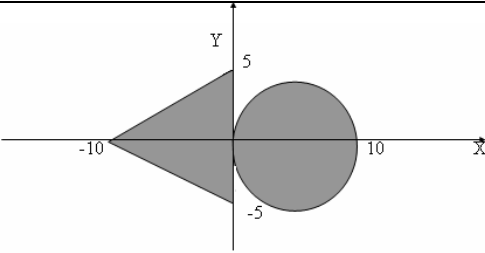
№	Задача 1	Задача 2	Задача 3
1	1) $n+++m$ 2) $m-->n$ 3) $n-->m$ 4) $\sin(x) + x^3 + \frac{1}{x^2 + 1}$		$\frac{(a+b)^2 - (a^2 + 2ab)}{b^2}$ $a=1000, b=0.0001$
2	1) $++n*++m$ 2) $m++<n$ 3) $n++>m$ 4) $x + \frac{1}{x^3 - x} - 2$		$\frac{(a-b)^2 - (a^2 - 2ab)}{b^2}$ $a=1000, b=0.0001$
3	1) $m--n$ 2) $m++<n$ 3) $n++>m$ 4) $x^4 - \cos(\arcsin(x))$		$\frac{(a+b)^3 - (a^3 + 3a^2b)}{3ab^2 + b^2}$ $a=100, b=0.001$

4	1) $n++*m$ 2) $n++<m$ 3) $--m>n$ 4) $\sqrt[3]{x-x^2+x^5}$		$\frac{(a-b)^3 - (a^3)}{3ab^2 - b^3 - 3a^2b}$ $a=100, b=0.001$
5	1) $--m-n++$ 2) $m*m<n++$ 3) $n-->++m$ 4) $tg(x) - (5-x)^4$		$\frac{(a-b)^3 - (a^3 - 3a^2b)}{3ab^2 - b^3}$ $a=100, b=0.001$
6	1) $m-++n$ 2) $m++>--n$ 3) $m--<++n$ 4) $25x^5 - \sqrt{x^2+x}$		$\frac{(a-b)^3 - (a^3 + 3ab^2)}{-3a^2b - b^3}$ $a=100, b=0.001$
7	1) $m+--n$ 2) $m++<--n$ 3) $--m>n--$ 4) $\sqrt[5]{x^3+x^4} + ctg(arctg(x^2))$		$\frac{(a-b)^3 - (a^3)}{-b^3 + 3ab^2 - 3a^2b}$ $a=100, b=0.001$ a) $Y = \sqrt[5]{x^3+x^4} + ctg(arctg(x^2))$

8	1) $n/m++$ 2) $m++<--n$ 3) $(m/n)++<n/m$ 4) $\sqrt{ x^3 - 1 } - 7 \cos \sqrt[3]{x^4 + x}$		$\frac{(a+b)^3 - (a^3)}{b^3 + 3ab^2 + 3a^2b}$ $a=100, b=0.001$
9	1) $m++/n--$ 2) $++m<n--$ 3) $n-->m$ 4) $\sin x^3 + x^4 + \sqrt[5]{x^2 + x^3}$		$\frac{(a+b)^3 - (a^3 + 3ab^2)}{3a^2b + b^3}$ $a=100, b=0.001$
10	1) $m/--n++$ 2) $m/n<n--$ 3) $m+n++>n+m$ 4) $x^5 \sqrt{ x-1 } + 25 - x^5 $		$\frac{(a+b)^3 - (a^3 + 3a^2b)}{3ab^2 + b^3}$ $a=100, b=0.001$
11	1) $n+++m--$ 2) $n^*m<n++$ 3) $n-->+++m$ 4) $2^x x \cos(x) + 1$		$\frac{(a+b)^4 - (a^4 + 4a^3b)}{6a^2b^2 + 4ab^3 + b^4}$ $a=10, b=0.01$
12	1) $n+++*m$ 2) $m--<n$ 3) $+++m>n$ 4) $\sqrt{x + \sqrt[4]{ x }} + x $		$\frac{(a+b)^4 - (a^4)}{6a^2b^2 + 4ab^3 + b^4 + 4a^3b}$ $a=10, b=0.01$

13	1) $(n++/--m)++$ 2) $++m < n--$ 3) $--m > ++n$ 4) $\sqrt[3]{e^x + \operatorname{tg}x} + \frac{1}{x}$		$\frac{(a+b)^4 - (a^4 + 6a^2b^2 + 4ab^3)}{b^4 + 4a^3b}$ $a=10, b=0.01$
14	1) $n++*--m$ 2) $n--<m++$ 3) $--n>--m$ 4) $\sqrt[4]{ x+1 } + \frac{1}{x^2}$		$\frac{(a+b)^4 - (a^4 + 6a^2b^2 + b^4)}{4ab^3 + 4a^3b}$ $a=10, b=0.01$
15	1) $n++/--m$ 2) $n-->n/m++$ 3) $m < n++$ 4) $1 + x \cos^2(x) + \sin^3(x)$		$\frac{(a-b)^4 - (a^4 - 4a^3b)}{6a^2b^2 - 4ab^3 + b^4}$ $a=10, b=0.01$
16	1) $m/--n++$ 2) $m/n < n--$ 3) $m+n++ > n+m$ 4) $\sqrt{ \sin(x) + x^2 + x }$		$\frac{(a-b)^4 - (a^4)}{6a^2b^2 - 4ab^3 + b^4 - 4a^3b}$ $a=10, b=0.01$
17	1) $n+++m--$ 2) $n*m < n++$ 3) $n-->++m$ 4) $\arcsin(x + x^2)$		$\frac{(a-b)^4 - (a^4 + 6a^2b^2 - 4ab^3)}{b^4 - 4a^3b}$ $a=10, b=0.01$

18	1) $n++*m$ 2) $m--<n$ 3) $++m>n$ 4) $\cos(\arctg(x))$		$\frac{(a-b)^4 - (a^4 + 6a^2b^2 + b^4) - 4ab^3 - 4a^3b}{b^2}$ $a=10, b=0.01$
19	1) $(n++/--m)++$ 2) $++m<n--$ 3) $--m>++n$ 4) $7\arctg(x^2)$		$\frac{(a+b)^2 - (a^2 + 2ab)}{b^2}$ $a=1000, b=0.0001$
20	1) $n++*--m$ 2) $n--<m++$ 3) $--n>--m$ 4) $5x^3\sqrt{\frac{1}{x^2} + \frac{1}{x^3}}$		$\frac{(a-b)^2 - (a^2 - 2ab)}{b^2}$ $a=1000, b=0.0001$
21	1) $n++/--m$ 2) $n-->n/m++$ 3) $m<n++$ 4) $\sqrt[3]{e^x - \sin x}$		$\frac{(a+b)^3 - (a^3 + 3a^2b)}{3ab^2 + b^2}$ $a=100, b=0.001$

22	1) $n++*m$ 2) $n++<m$ 3) $--m>n$ 4) $2^{-x} \sqrt{x+4\sqrt{ x }}$		$\frac{(a-b)^3 - (a^3)}{3ab^2 - b^3 - 3a^2b}$ $a=100, b=0.001$
23	1) $--m-n++$ 2) $m*m<n++$ 3) $n-->+m$ 4) $1 + \frac{1}{x} + \frac{1}{x^2}$		$\frac{(a-b)^3 - (a^3 - 3a^2b)}{3ab^2 - b^3}$ $a=100, b=0.001$
24	1) $m-++n$ 2) $m++>--n$ 3) $m--<+n$ 4) $\arcsin(x+1)$		$\frac{(a-b)^3 - (a^3 + 3ab^2)}{-3a^2b - b^3}$ $a=100, b=0.001$
25	1) $m+--n$ 2) $m++<--n$ 3) $--m>n-$ 4) $\arccos(x+x^2)$		$\frac{(a-b)^3 - (a^3)}{-b^3 + 3ab^2 - 3a^2b}$ $a=100, b=0.001$

5. Методические указания

1. Для ввода и вывода данных использовать операции >> и << и стандартные потоки cin, cout.
2. Ввод данных для заданий А и Б организовать с клавиатуры.
3. При вычислении выражений подключить библиотеку <math.h> для вычисления функций (например, pow(x,y) для вычисления x^y).
4. Вывод результатов для задания А организовать в виде:

```
n?1
n?2

n=3 n=1 m+++n=3
Press any key to continue
```

5. При выполнении задания Б использовать переменную логического типа, а не условный оператор.
6. При выполнении задания В использовать вспомогательные переменные для хранения промежуточных значений. Например:
`c=pow(a,3); d=3*pow(a,2)*b; e=3*a*pow(b,2); f=pow(b,3);`

6. Содержание отчета

- 1) Постановка задачи (общая и конкретного варианта).
- 2) Формулы, используемые при решении задачи (математическая модель).
- 3) Программы для решения задач на языке C++.
- 4) Описание используемых в программе стандартных функций.
- 5) Система тестов для проверки правильности работы программы и результаты выполнения тестов.
- 6) Объяснение результатов работы программы.