Автономное образовательное учреждение
высшего образования Ленинградской области
«Государственный институт экономики, финансов, права и технологий»

Кафедра информационных технологий, безопасности и права

Е.В. Бенза, С.М. Бенза

ПРОГРАММИРОВАНИЕ C++

Учебно-методическое пособие для выполнения лабораторных работ
для студентов, обучающихся по направлению подготовки
38.03.05 – «Бизнес-информатика»
(уровень бакалавриат)

Гатчина
2022

Учебно-методическое пособие «Программирование C++» по курсу «Программирование» рассмотрено и утверждено на заседании кафедры информационных технологий, безопасности и права, протокол № 1 от 31.08.2022 г.

Составители: **Е.В. Бенза**, доцент кафедры информационных технологий, безопасности и права ГИЭФПТ, к.т.н., доцент;
С.М. Бенза, преподаватель кафедры информационных технологий, безопасности и права, системный администратор АО «Лентехностром»

Рецензент: **В.А. Драбенко**, зав. кафедрой информационных технологий, безопасности и права ГИЭФПТ, д.т.н., профессор

Данное пособие содержит справочные сведения по языку программирования C++, задания для выполнения лабораторных работ, а также требования по содержанию и оформлению отчёта по работе.

Учебное пособие предназначено для студентов всех форм обучения по специальности «Бизнес-информатика».

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Методические указания предназначены для студентов, обучающихся по специальности 38.03.05 – «Бизнес-информатика», профиль «Архитектура предприятия».

Данные методические указания содержат задания по лабораторным работам на языке программирования высокого уровня С++, приведены пояснения к их выполнению и правила оформления отчётов по выполненным работам. Также в данном пособии представлены краткие справочные сведения о языке программирования.

Эти методические указания предназначены для студентов, начинающих изучать программирование на С++. Задания, представленные в этом издании, ориентированы на создание программ, основанных на трех алгоритмических структурах: линейной, ветвящейся и циклической.

СПРАВОЧНЫЕ СВЕДЕНИЯ О ЯЗЫКЕ ПРОГРАММИРОВАНИЯ С++

Объявление функции и структура программы. Программа на языке С++ состоит из некоторого набора функций. Одну из функций принято называть **main.** Самая простая программа состоит только из этой функции. Для того, чтобы программа начала работать, функцию **main**нужно объявить. Варианты объявления этой функции зависят от того, получает она или нет параметры, возвращает или нет результат. Варианты объявления **main** представлены в табл. 1**.**

Таблица 1

Варианты объявления функции main

№	Условия использования объявления	Объявление функции
1	2	3

1	Функция main получает параметры и возвращает результат	int**main**(intargc, char* argv[])
		{
		//здесь инструкции программы
		(код)
		return (значение);
		}

2	Функция main не получает параметры, но возвращает результат	int**main**()
		{
		//здесь инструкции программы
		(код)
		return(значение);
		}

3	Функция main не получает параметры и не возвращает результат	void**main**()
		{
		//здесь инструкции программы
		(код)
		}

Примечание. В лабораторной работе № 1 программа для решения задачи линейной структуры будет состоять из единственной функции main, для объявления которой используется вариант второй из табл. 1. Это значит, что в лабораторной работе № 1 функции main не присваиваются параметры, но она возвращает результат.

Типы данных. К основным типам данных языка С++ относятся:

- целые числа (int и др.);
- дробные (действительные) числа (float и др.);
- символы (char).

Целые числа, числа с плавающей точкой и символы могут быть представлены в различных форматах (табл. 2, 3, 4).

Таблица 2

Формат представления целых чисел

Формат	Бит	Диапазон значений
int	16	–32 768 ...32 767
shortint	16	–32 768 ...32 767
unsignedint	16	0... 65 535
enum	16	–32 768 ...32 767
long	32	–2 147 483 648 ... 2 147 483 647
unsignedlong	32	0 ... 4 294 967 295

Таблица 3

Формат представления дробных чисел

Формат	Бит	Диапазон значений
float	32	$3{,}4\times10^{-38}...3{,}4\times10^{+38}$
double	64	$1{,}7\times10^{-308}... 1{,}7\times10^{+308}$
longdouble	8	3,4×10–4932... 1,1×10+4932

Таблица 4

Формат представления символов

Тип	Бит	Диапазон значений
unsignedchar	8	0 ... 255
char	8	–128... 127

При написании программ на языке С++ используются арифметические операторы, их обозначения и приоритеты выполнения представлены в табл. 5 и 6.

Таблица 5

Арифметические операторы С++

Оператор	Действие
+	Сложение
−	Вычитание, а также унарный минус
*	Умножение
/	Деление
%	Деление по модулю
−−	Декремент
++	Инкремент

Таблица 6

Приоритет использования арифметических операторов в С++

Приоритет	Операторы
++−−	Инкремент и декремент (высший приоритет)
−	Унарный минус
*/%	Умножение, деление и деление по модулю
+−	Сложение и вычитание (низший приоритет)

Основным оператором в С++ является оператор присваивания, инструкции по использованию этого оператора представлены в табл. 7.

Таблица 7

Составные операторы присваивания в С++

Инструкция	Соответствующая «обычная»инструкция присваивания
x++	x = x + 1
x—	x = x – 1
x+=y	x = x + y
x–=y	x = x – y
x*=y	x = x * y
x%=y	x = x % y

Пробелы и скобки. Для лучшего понимания выражения в С++ могут содержать пробелы или символы табуляции. Нижеприведённые выражения являются тождественными.

Y=45x/a*b(245/c) Y = 45 x / a * b (245 / c)

Круглые скобки, как и в математике, повышают приоритет операции. Операция в круглых скобках выполняется в первую очередь.

Ввод данных с клавиатуры компьютера. Для ввода данных с клавиатуры в программу используется оператор **>>**. В языке С++ этот оператор работает с потоком ввода/вывода **iostream**, который описан в заголовочном файле **iostream.h.** Для считывания данных с клавиатуры используется следующий формат этого оператора:

cin>>var;

Имя объекта **cin** составлено из частей слов **ConsoleInput** (консольный ввод). По умолчанию объект **cin** связывается с клавиатурой, хотя его можно перенаправить и на другие устройства. Элемент **var** означает переменную, указанную с правой стороны от оператора, принимающую вводимые данные. Например, если переменная **a** объявлена в программе как переменная целого типа **int a**; то строка кода "**cin>> a;**" означает «взять» данные из потока ввода и присвоить их переменной a.

Вывод данных на экран монитора. Для вывода данных результата выполнения программы на экран монитора применяется оператор <<. В языке C++ этот оператор работает с потоком ввода/вывода **iostream**, который описан в заголовочном файле **iostream.h**. Для вывода данных на монитор используется следующий формат этого оператора:

cout<<var;

Имя объекта **cout** составлено из частей слов **ConsoleOutput** (консольный вывод). По умолчанию объект **cout** связывается с монитором (дисплеем). Элемент **var** означает переменную, указанную с правой стороны от оператора, которая содержит выводимые данные или некоторую строку символов. Например, строка кода "**cout<< a**;" означает «взять» данные из переменной **a** и поместить их в поток вывода. Строка кода "**cout<< "Введите число a: ";**" содержит инструкцию поместить в поток вывода строку символов "Введите число a:". Объект **cout** помещает строку символов (символы, заключенные в двойные кавычки) в поток вывода **iostream**. Ниже приведён пример линейной программы, использующей операторы ввода и вывода.

```
#include <iostream.h>
int main() {
intx,y,z;
cout<< "Введитех: ";
cin>>x;
cout<< "Введитеy: ";
cin>>y;
z = x*y;
cout<< "x + y = " <<z<< "\n";
return 0; }
```

Специальный символ **"\n"** – так называемый управляющий символ, не имеющий экранного отображения и использующийся для управления процессами, служит для установки указателя изображения символов на экране в первую позицию курсора, в начало следующей строки. Для этой же цели используется встроенный идентификатор **endl.** Он составлен из частей слов **EndLine** (конец линии) и автоматически под-

держивается средствами C++. Он обеспечивает перевод потока вывода к началу новой строки вывода на консоль. Нижеприведённые примеры написания кода приводят к одному результату.

cout<< "a + b = " << c << "\n"; **cout<< "a + b = " << c <<endl;**

Перед использованием в программном коде операторов >> и << нужно до описания главной функции подключить заголовочный файл iostream.h, в противном случае компилятор выдаст сообщение об ошибке. Для этого используется директива #include: **#include<iostream.h>**

В табл. 8 представлены стандартные математические функции C++.

Таблица 8

Стандартные математические функции языка C++

Функция	Описание
doublesin(double x);	Возвращает синус угла. Величина угла должна быть задана в радианах. Заголовочный файл: <math.h>
doublecos(double x);	Возвращает косинус угла. Величина угла должна быть задана в радианах. Заголовочный файл: <math.h>
doubletan(double x);	Функция тангенса. Угол задается в радианах. Заголовочный файл: <math.h>
doublesinh(double x);	Возвращает значение гиперболического синуса для x. Заголовочный файл: <math.h>
doublecosh(double x);	Возвращает значение гиперболического косинуса для x. Заголовочный файл: <math.h>
doubletanh(double x);	Возвращает значение гиперболического тангенса для x. Заголовочный файл: <math.h>
doubleasin(double x);	Возвращает выраженную в радианах величину угла, косинус, синус или тангенс которого передан соответствующей функции в качестве аргумента. Аргумент функции должен находиться в диапазоне от -1 до 1. Заголовочный файл: <math.h>
doubleacos(double x);	
doubleatan(double x);	
double atan2(double y, double x);	Функция арктангенса от значения y/x. Возвращает значение гиперболического синуса для x. Заголовочный файл: <math.h>
doubleexp(double x);	Возвращает значение, равное экспоненте аргумента x(e), где e – основание натурального логарифма). Заголовочный файл: <math.h>
doublelog(double x);	Возвращает значение натурального логарифма (lnx). Заголовочный файл: <math.h>
double log10(doublex);	Возвращает значение десятичного логарифма $\log_{10} x$. Заголовочный файл: <math.h>
doublepow(doublex, doubley);	Возвращает значение равное x в степени y. Заголовочный файл: <math.h>
doublesqrt(double x);	Возвращает значение, равное квадратному корню из аргумента *x*. Заголовочный файл: <math.h>
doublefabs (double x);	Возвращает целое (abs) или дробное (fabs) абсолютное значение аргумента, в качестве которого можно использовать выражение соответствующего типа. Заголовочный файл: <math.h>

Приоритеты выполнения операций и функций в выражениях в С++ следующие:

1. Выполняется выражение в скобках.
2. Вычисляются стандартные функции.
3. Возведение в степень.
4. Умножение, деление.
5. Сложение, вычитание.

Запись всех элементов выражений выполняется в одну строку. Поэтому суммы и разности в числителях и знаменателях дробей, а также произведения в знаменателях необходимо заключать в скобки. Примеры записи математических выражений на С++ приведены в табл. 9.

Таблица 9

Примеры записи математических выражений на С++

Математическое выражение	Запись на С++
$\frac{a+b}{c-df}+\frac{x}{y-z}$	(a+b)/(c-d*f)+x/(y-z)
$\left\|18{,}5-\sqrt{25+4{,}8tg^2y}\right\|$	abs(18.5-sqrt(25+4,8*pow(tan(y),2)))
$\sin ab^3+e^{-2z}$	sin(a*pow(b,3))+exp(-2*z)
$\sqrt[2a]{z}$	pow(z,1/2*a)

РЕКОМЕНДАЦИИ ПО ОФОРМЛЕНИЮ ОТЧЁТА ПО ЛАБОРАТОРНЫМ РАБОТАМ

Содержание отчёта.

1. Титульный лист (приложение).
2. Содержание.
3. Цель работы.
4. Основная часть должна содержать задание, структурную схему алгоритма, программный код, «скриншот» экрана с результатом выполнения работы.
5. Ответы на контрольные вопросы.
6. Выводы по проделанной работе.
7. Список использованных источников.

Шрифт 14 пт (Times New Roman), с межстрочным интервалом – 1,5. Требования к полям: левое – 30 мм, правое – 10 мм, верхнее – 20 мм, нижнее – 20 мм. Выравнивание по ширине страницы. Каждый раздел, а также выводы и предложения начинаются с новой страницы. Точку в конце заголовка, располагаемого по ширине с абзацного отступа строки, не ставят. Не рекомендуется подчеркивать заголовки. Не допускается переносить часть слова в заголовке. Абзацы начинаются с новой (красной) строки, их печатают с отступом, равным 1,25 см. Каждый рисунок должен сопровождаться содержательной подпи-

сью, которая печатается под рисунком по центру страницы в одну строку с номером. Таблица должна иметь заголовок, помещаемый под словом «Таблица» над соответствующей таблицей. Слово «Таблица» и заголовок начинаются с прописной буквы с абзацного отступа и не подчеркиваются. Заголовки граф таблиц должны начинаться с прописных букв; подзаголовки – со строчных, если они составляют одно предложение с заголовком, и с прописных, если они самостоятельные. При переносе таблицы на следующую страницу шапку таблицы следует повторить, и над ней помещают слова «Продолжение таблицы» с указанием ее номера. Если заголовок таблицы громоздкий, допускается его не повторять; в этом случае пронумеровывают графы и повторяют их нумерацию на следующей странице. Не допускается оставлять в таблице пустые графы. В этом случае либо ставится прочерк, либо пишется «нет данных». Таблицы следует нумеровать арабскими цифрами порядковой нумерацией в пределах всей работы. При оформлении таблицы в левом верхнем углу начиная с красной строки с прописной буквы пишут слово «Таблица», далее ставится ее номер и без абзацного отступа через тире пишется заголовок таблицы. Заголовок (название таблицы) следует писать с прописной буквы, без точки в конце.

ЛАБОРАТОРНЫЕ РАБОТЫ

Лабораторная работа № 1. Алгоритм линейной структуры

Цель работы: ознакомиться с алгоритмами линейной структуры, изучить правила построения алгоритмов при помощи блок схем, научиться составлять алгоритмы линейных процессов и написать на их основе программный код.

Теоретические сведения. Алгоритмы – это точное предписание, которое определяет процесс, ведущий от исходных данных к требуемому конечному результату, например, правила сложения, умножения, решения алгебраических уравнений, умножения матриц и т.п. В информационных технологиях алгоритм определяет процесс, начинающийся с обработки некоторой совокупности возможных исходных данных и направленный на получение определенных этими исходными данными результатов. Для того, чтобы создать алгоритм, нужно описать следующие его элементы:

- набор объектов, составляющих совокупность возможных исходных данных, промежуточных и конечных результатов;
- правило начала;
- правило непосредственной переработки информации (описание последовательности действий);

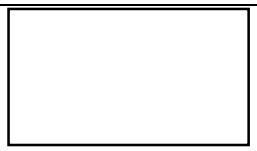
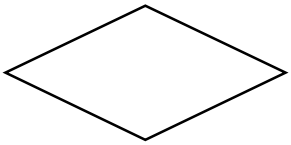
● правило окончания;

● правило извлечения результатов.

Алгоритмы можно описать разными способами – словесно – формульным, структурным или блок-схемным, с помощью графов – схем или с помощью сетей Петри. При блок-схемном описании алгоритм изображается геометрическими фигурами (блоками), связанными по управлению линиями (направлениями потока) со стрелками. В блоках записывается последовательность действий. Блок-схема оформляется в соответствии с *единой системой программной документации (ЕСПД)*, частью которой является Государственный стандарт – ГОСТ 19.701-90 «Схемы алгоритмов программ, данных и систем». Виды блоков и их функциональные значения представлены в табл. 10.

Таблица 10

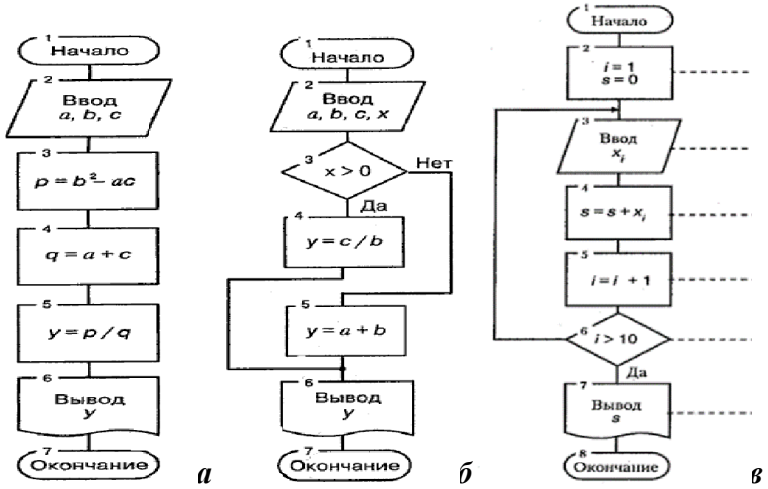
Основные элементы схем алгоритма

Наименование	Обозначение	Функции
Процесс		Выполнение операции или группы операций, в результате которых изменяется значение, форма представления или расположение данных.
Ввод-вывод		Преобразование данных в форму, пригодную для обработки (ввод) или отображения результатов обработки (вывод).
Решение		Выбор направления выполнения алгоритма в зависимости от некоторых переменных условий.
Предопределенный процесс		Использование ранее созданных и отдельно написанных программ (подпрограмм).
Документ		Вывод данных на бумажный носитель.
Пуск-остановка (начало – конец)		Начало, конец, прерывание процесса обработки данных.
Соединитель		Указание связи между прерванными линиями, соединяющими блоки.
Межстраничный соединитель		Указание связи между прерванными линиями, соединяющими блоки, расположенные на разных листах.
Комментарий	[	Связь между элементом схемы и пояснением.

Описание других элементов схем можно найти в соответствующих ГОСТ.

Правила оформления алгоритмов блок-схемным способом. Расстояние между параллельными линиями потоков должно быть не менее 3 мм, между остальными элементами схемы – не менее 5 мм. Горизонтальный и вертикальный размеры блока должны быть кратны 5 мм (делиться на 5 нацело). Отношение горизонтального и вертикального размеров блока b/a = 1.5 является основным. Блоки «Начало», «Конец» и «Соединитель» имеют высоту a/2, т. е. вдвое меньше основной высоты блоков. Для размещения блоков рекомендуется поле листа разбивать на горизонтальные и вертикальные (для разветвлявшихся схем) зоны. Для удобства описания блок-схемы каждый ее блок следует пронумеровать. Удобно использовать сквозную нумерации блоков. Номер блока располагают в разрыве в левой верхней части рамки блока.

Существует три основных типа алгоритмов: последовательность двух или более операций; выбор направления; повторение. Любой вычислительный процесс может быть представлен как комбинация этих элементарных алгоритмических структур. В соответствии с основными типами алгоритмы могут быть линейные, ветвящиеся и циклические, представлены на рис. а, б, в.



Виды алгоритмов.

Задания к лабораторной работе № 1

Построить блок-схему алгоритма для вычисления по варианту (табл. 11), написать программу на C++, полученный результат вывести на экран и ответить на контрольные вопросы. Вариант соответствует номеру студента по списку группы. Составить отчёт по выполненной работе (см. рекомендации к оформлению отчёта по лабораторной работе).

Ответить на контрольные вопросы:

1. Что такое алгоритм?
2. Какие типы блоков Вы знаете?

3. Перечислите основные свойства алгоритма.
4. Какие правила регулируют оформление блок схем алгоритмов.

Таблица 11

Варианты заданий к лабораторной работе № 1

№ варианта	Задание	Исходные данные

1	Дана функция $y=1/\ x-ax^2+\|bx\|$;
	Вычислить значение y при заданном a, b и x.
		a=12;
		b=1,25;
		x=-3,8

2	$u=\frac{\|a\|x+\sin x}{b}$.	a=1,8;
		b=18,5;
		x=4

3	$y=\frac{a+b}{2a-b}(f+c)sin^2x+e^{ab}$	a=28;
		b=28,5;
		x=5

4	$R=\sin x+\cos x^2+x^3$	X=0,5
5	$Y=\frac{5x-\frac{x^2}{2}}{x-\frac{x^2}{3}}$. При условии, что $x\neq 0$.	X=100

6	$z=\frac{x^2+\|bx-3c\|}{\ln(x^a+bc+a)}$	a=12;
		b=18,5;
		c=285
		x=5

7	$Y=\|x\|-\sin(x)-a$.	a=1,8; x=-4,45
8	$U=\frac{\|x\|-a\sin x}{a-x}$.	a=1,8; x=8,147
9	$y=3{,}85x^2ln\|a-b^{x+c}\|$	a=10 b=20; c=30 x=5
10	$U=AZ^2+\frac{\frac{A}{2}Z^2}{13{,}61}-AZ$	A=15 Z=100,456

11	$F=a\cos^2\frac{b}{a}\sin x+c$. при условии, что $a\neq 0$.	a=1,8;
		b=18,5;
		x=4;
		c=145,456

12	$z=lnb\cos x^5\sqrt{\frac{ac}{x^a}}$	a=124;
		b=25;
		x=5; c=45,6

13	$y=\frac{a+bx+cx^2}{1-\tan x}+\left(\frac{a}{b}\right)^4$	a=124;
		b=25;
		x=5;
		c=45,6

14	$U=\frac{5x^2+\sin x}{\cos x}$	x=147,741

15	$t = a^{\frac{1}{x}} + c\sqrt{\sin b + x}$	a=22,8;
		b=1,5;
		x=40;
		c=4,7

16	$R = \frac{(a+b)^2 \sin c}{d}$	a=28,456;
		b=5;
		c=12,45;
		d=100

17	$y = \sqrt{a + \sqrt{b + \sqrt[3]{c + x}}}$	a=150;
		b=20;
		x=2;
		c=6

18	$y = \frac{xa}{b}\sqrt{e^{2x} + 1}\frac{1}{a^x + 1}$	a=15;
		b=20;
		x=10;

19	$z = \frac{a + bx - cx^2}{\cos x + e^{\sqrt{x}}} + x^{a+b}$	a=15,8;
		b=2;
		x=4;
		c=12,5

20	$y = \frac{x^a}{1 + \frac{x^b}{1+x^c}}$	a=15,8;
		b=225;
		x=48,5;
		c=8

Лабораторная работа № 2.
Разветвляющиеся вычислительные процессы

Цель работы: изучить особенности построения разветвляющихся вычислительных процессов с использованием условных операторов **if** и **switch** и конструкций if{..}elseif{..}else{} и switch(){case:..;break;.. default:;},научиться составлять алгоритмы разветвляющихся процессов и написать на их основе программный код.

Теоретические сведения. Для того чтобы иметь возможность реализовать в программе логические переходы используются ***условные операторы*** if и switch. Эти операторы можно представить в виде узловых пунктов, достигая которых программа делает выбор по какому из возможных направлений двигаться дальше.

Оператор if. Конструкция if{..}elseif{..}else{}. Синтаксис данного оператора следующий:

if (логическое выражение)

<оператор>

Если значение параметра **«логическое выражение»** равно **«истинно» (или true)**, выполняется оператор, иначе (**«логическое выражение»** равно **«ложь» (или false**) он пропускается программой. **«Логическое выражение»** является условным выражением, в котором выполняется проверка некоторого условия. В табл. 12 представлены варианты простых логических выражений оператора **if**.

Таблица 12

Логические выражения оператора if

if(a < b)	Истинно, если переменная a меньше переменной b, и ложно в противном случае.
if(a > b)	Истинно, если переменная a больше переменной b, и ложно в противном случае.
if(a == b)	Истинно, если переменная a равна переменной b, и ложно в противном случае.
if(a <= b)	Истинно, если переменная a меньше либо равна переменной b, и ложно в противном случае.
if(a >= b)	Истинно, если переменная a больше либо равна переменной b, и ложно в противном случае.
if(a != b)	Истинно, если переменная a не равна переменной b, и ложно в противном случае.
if(a)	Истинно, если переменная a не равна нулю, и ложно в противном случае.

Пример использования оператора ветвления if в простом случае – программа определения знака введенного числа:

```
#include<stdio.h>
int main()
{
float x;
printf(“Введите число: ”);
scanf(“%f”,&x);
if(x< 0) printf(“Введенное число % f является отрицательным. \n”, x);
if(x>= 0) printf(“Введенное число % f является неотрицательным. \n”, x);
return 0;
}
```

В приведенном выше тексте программы два условных оператора **if** можно заменить одним, используя конструкцию:

if (логическое выражение)
<оператор1>
else
<оператор2>

которая интерпретируется следующим образом: если «**логическое выражение**» «**истинно**», то выполняется «оператор1», иначе – выполняется «оператор2». В случаях, когда при выполнении какого-либо условия необходимо записать более одного оператора, необходимо использовать фигурные скобки, т.е. использовать конструкцию вида:

```
if (выражение) {
<оператор1.1>
<оператор1.n>
```

}
else{
<оператор2.1>
<оператор2.k>
}

Формально после ключевого слова else можно поставить еще один оператор условия **if**, в результате получим конструкцию условных переходов показанную ниже, где n, k, lобозначают количество операторов:

if(логическое выражение 1)
{
<оператор1.1>
…
<оператор1.n>
}
elseif(логическое выражение 2)
{
<оператор2.1> …
<оператор2.k>
} else {
<оператор3.1> …
<оператор3.l>
}

Оператор switch. Конструкция switch(){case:..;break;.. default:;}. Условная операция **if** помогает в написании программ, в которых необходимо производить выбор между небольшим числом возможных вариантов. При условии необходимости осуществления в программе выбора одного из множества вариантов необходимо использовать следующую конструкцию:

switch(переменная)
{
case константа1: <операторы>;break;
case константа2: <операторы>;break;
…
default: <операторы>;
}

Данный оператор последовательно проверяет на равенство переменной константам, стоящим после ключевого слова **case**. Если ни одна из констант не равна значению переменно, то выполняются операторы, находящиеся после слова **default**. Оператор **switch** имеет следующую особенность: если значение переменной равно значению **константы1** и выполняются операторы, стоящие после первого клю-

чевого слова **case**. После этого выполнение программы продолжится проверкой переменной на равенство **константы2**, что часто приводит к неоправданным затратам ресурсов компьютера. Во избежание такой ситуации следует использовать оператор **break** для перехода программы к следующему оператору после **switch.** Пример использования оператора **switch**:

```
#include<stdio.h>
intmain()
{ intx; printf(“Введитечисло: ”); scanf(“%d”,&x);
switch(x)
{
case 1 : printf(“Введеночисло 1\n”);break;
case 2 : printf(“Введеночисло 2\n”); break;
default :printf(“Введено другое число\n”); }
charch; printf(“Введитесимвол: ”); scanf(“%c”,&ch);
switch(ch)
{
case ‘a’ : printf(“Введенсимвола\n”); break;
case ‘b’ : printf(“Введенсимвол b\n”); break;
default :printf(“Введен другой символ\n”);
} return 0;
}
```

Оператор **switch** может производить выбор только на основании равенства своего аргумента одному из перечисленных значений **case**, т.е. проверка выражений типа x<0 в данном случае невозможна.

Задания к лабораторной работе № 2

Построить блок-схему алгоритма для вычисления по варианту (таблица 12), написать на C++ программу с условным оператором if, полученный результат вывести на экран. Написать программу с использованием оператора switch в соответствии с номером своего варианта, полученный результат вывести на экран. Вариант соответствует номеру студента по списку группы. Ответить на контрольные вопросы. Составить отчёт по выполненной работе (см. рекомендации к оформлению отчёта по лабораторной работе).

Ответить на контрольные вопросы:

1. Напишите программный код с использованием условного оператора if для определения знака переменной val.
2. Для каких случаев предпочтительно использовать оператор switch?
3. Как можно записать логическое равенство в операторе if?

Таблица 12

Варианты заданий к лабораторной работе № 2

№ варианта	Оператор if	Оператор switch
1	$\begin{cases} \lg(x) & \text{при } 4 <= x < 6, \\ \sin^2(x) & \text{при } x >= 6, \\ e^{0.1x} & \text{при } -3 <= x < 4, \\ \dfrac{\cos(x)}{x+10} & \text{при } x < -3 \end{cases}$	Перевести введённые символы от a до f в верхний регистр
2	$\begin{cases} e^{\sqrt{x}} & \text{при } x > 2, \\ \cos^2(x) & \text{при } 2 >= x > 1, \\ \lg(20x) & \text{при } 1 >= x > 0.5, \\ e^{0.2x} & \text{при } x <= 0.5 \end{cases}$	Перевести введённые символы от A до F в нижний регистр
3	$\begin{cases} \dfrac{1}{x} & \text{при } x > 4, \\ e^{-x} & \text{при } 2 < x <= 4, \\ x^2 & \text{при } -1 < x <= 2, \\ \dfrac{\lvert x-1 \rvert}{2x} & \text{при } x <= -1 \end{cases}$	Заменить введенные символы от 0 до 9 соответствующим числом
4	$\begin{cases} \dfrac{2}{x} & \text{при } x < -5, \\ x\sin x & \text{при } 2 > x >= -5, \\ \dfrac{x+10}{2+x^2} & \text{при } 3 > x >= 2, \\ x+\sqrt[3]{x} & \text{при } x >= 3 \end{cases}$	Заменить введенные символы от 1 до 8 соответствующим символом
5	$\begin{cases} \sqrt{x} & \text{при } x > 7, \\ e^{-x} & \text{при } 2 < x <= 7, \\ e^{x} & \text{при } -1 < x <= 2, \\ \sin(x) & \text{при } x <= -1 \end{cases}$	Заменить введенные числа от 0 до 5 соответствующими символами, а все другие значения заменить буквой y
6	$\begin{cases} 3x^3 & \text{при } x > 20, \\ \ln(x+3) & \text{при } 20 >= x > 10, \\ e^{-x^2} & \text{при } 10 >= x > 3, \\ \lvert \sin(x) \rvert & \text{при } 3 >= x \end{cases}$	Заменить введенные числа от 0 до 5 соответствующим числами, а все другие символы заменять числом -8
7	$\begin{cases} \dfrac{\sin^2(x)}{2} & \text{при } x > 20, \\ \sqrt[3]{x} & \text{при } 20 >= x >= 6, \\ \sin^2(x) & \text{при } 6 >= x > -4, \\ 0 & \text{при } x <= -4 \end{cases}$	Перевести введённые символы от a до f на верхний регистр, а другие символы заменять на W

8	$\begin{cases} \sin(\sqrt{x}) & \text{при } x > 30, \\ \sqrt{x} & \text{при } 30 >= x > 10, \\ \lg(0.5 * x) & \text{при } 10 >= x > 2, \\ \lvert \sin(x) \rvert & \text{при } x <= 2 \end{cases}$	Перевести введённые символы от A до F в нижний регистр, а все другие символы заменять на V
9	$\begin{cases} 2\sqrt{(x^2 + 15)} & \text{при } x < -6, \\ 4\cos(x) & \text{при } -6 <= x < 2, \\ \frac{\sin(x - 3)}{2} & \text{при } 2 <= x < 10, \\ \frac{tgx}{10} & \text{при } x >= 10 \end{cases}$	Проверить введённое число на равенство со значениями 0, 4, 8, 9 и 30
10	$\begin{cases} 1 + x^2 & \text{при } x > 25, \\ 2 + x^2 & \text{при } 25 >= x > 8, \\ 3 + x^2 & \text{при } 8 >= x > 2, \\ 4 + x^2 & \text{при } x <= 2 \end{cases}$	Сравнить введённый символ с a, s, d, j и e
11	$\begin{cases} x \lvert x + 21 \rvert & \text{при } x < -14, \\ x^2 \ln \lvert x^2 + 48 \rvert & \text{при } -14 <= x < -5, \\ \frac{x}{3} + \sqrt{(x^2 + 16)} & \text{при } -5 <= x < 0, \\ 2 + \frac{x}{3} & \text{при } x >= 0 \end{cases}$	Перевести введённый символ из шестнадцатеричной системы счисления в десятичную систему
12	$\begin{cases} x + (\lvert x \rvert - 12)^2 & \text{при } x < -8, \\ \frac{\sin(x) + 3}{\cos(x) + 15} & \text{при } -8 <= x < 7, \\ x\sqrt{(x^2 - 36)} & \text{при } 7 <= x < 10, \\ x + 8 & \text{при } x >= 10 \end{cases}$	Перевести введённый символ из шестнадцатеричной системы счисления в восьмеричную систему
13	$\begin{cases} \frac{\cos(x) + 14}{\sin(x) + 7} & при \quad x <= 4, \\ \sqrt[3]{(x + \ln(\lvert x - 8 \rvert + 10))} & при \quad 4 < x < 12, \\ \sqrt{(x - 13)} & при \quad 12 <= x < 38 \\ 5x & при \quad x >= 38 \end{cases}$	Перевести введённый символ из восьмеричной системы счисления в десятичную систему
14	$\begin{cases} \sqrt{\lvert x \rvert} + \frac{\sin(x)}{2} & при \quad x < -12, \\ \sin(x - 5) & при \quad -12 <= x < 8, \\ \frac{\sqrt{(x + 6)}}{x + 3} & при \quad 8 <= x < 10, \\ \frac{4}{3 + x} & при \quad x >= 10 \end{cases}$	Заменить в введенном целом числе от 100 до 110 все нули на 9

15	$\begin{cases} \frac{x\sin(x)+5}{2+\sin(x)} & \text{при } x<5, \\ \sqrt{\lvert x-7\rvert}+\frac{x}{2} & \text{при } 5<=x<10, \\ \lg(x) & \text{при } 10<=x<111, \\ 2x & \text{при } x>=111 \end{cases}$	Возвести в квадрат введенные целые числа от -3 до 3.
16	$\begin{cases} \sqrt{(\lvert x\rvert-1)} & \text{при } x<=-20, \\ \frac{x+\sin(x)}{2} & \text{при } -20<x<6, \\ x^2+\ln(x-2) & \text{при } 6<=x<12, \\ x^2+\ln(10) & \text{при } x>=12 \end{cases}$	Возвести в третью степень введенные целые числа от -3 до 3.
17	$\begin{cases} \lg(1.5x) & \text{при } 4<=x<7, \\ \sin^4(x) & \text{при } x>=7, \\ 0.5e^{0.1x} & \text{при } -3.5<=x<4, \\ \frac{\cos(x)}{\lvert x\rvert+10} & \text{при } x<-3.5 \end{cases}$	Заменить в введенном целом числе от 100 до 110 все нули на Z
18	$\begin{cases} \frac{\sin(x)+2}{\cos(x)+10} & \text{при } x<=5, \\ \sqrt[3]{(2+\ln(\lvert x-8\rvert+2))} & \text{при } 5<x<15, \\ \sqrt{x-13} & \text{при } 15<=x<38, \\ 5+x & \text{при } x>=38 \end{cases}$	Возвести в квадрат введенные целые числа от -8 до 8.
19	$\begin{cases} \frac{x}{2} & \text{при } x>5, \\ 10e^{-x} & \text{при } 2<x<=5, \\ \frac{3+x^2}{2x} & \text{при } -2<x<=2, \\ \frac{\lvert x-100\rvert}{2x} & \text{при } x<=-2 \end{cases}$	Сравнить введённый символ с A, B, Y и W
20	$\begin{cases} \sqrt{x} & \text{при } x>9, \\ 5e^{-x} & \text{при } 2<x<=9, \\ e^x & \text{при } 1<x<=2, \\ \cos(x) & \text{при } x<=1 \end{cases}$	Сравнить введённый символ с a, b, y и w

Лабораторная работа № 3. Циклические процессы

Цель работы: изучить особенности построения циклических вычислительных процессов с использованием операторов цикла **while, for** и **dowhile**, научиться составлять алгоритмы циклических процессов и написать на их основе программный код.

Теоретические сведения. С помощью оператора цикла **while** реализуется цикл, который выполняется до тех пор, пока истинно условие цикла. Синтаксис данного оператора следующий:

while(<условие>)
{
<тело цикла>
}
Суммирование элементов ряда $\sum_{i=1}^{\infty} S_i$, пока $S<N; S_0=0$:
int N=20, i = 1; **long** S = 0; **while**(S < N)
{
S=S+i; i++; }

В данном примере реализуется цикл while с условием S < N. Так как начальное значение переменной S=0, а N=20, то условие истинно, и выполняется тело цикла, в котором осуществляется суммирование переменной i и увеличение ее на 1. На 20 итерации значение S=20, условие станет ложным, и цикл будет завершен.

Работа оператора цикла **for** подобна оператору while с той лишь разницей, что оператор for подразумевает изменение значения некоторой переменной и проверки ее на истинность. Работа данного оператора продолжается до тех пор, пока истинно условие цикла. Синтаксис оператора for следующий:

for(<инициализация счетчика>;<условие>;<изменение значения счетчика>)
{
<тело цикла>
}
Нахождение суммы всех целых чисел от0до 100:
```
int sum = 0;
inti;
for (i = 1; i< 100; i = i + 1)
// заголовок цикла
sum = sum + i;      // тело цикла
```
То же самое можно записать следующим образом:
```
int sum = 0;
inti = 1;
for (; i<= 100; ) {
sum = sum + i;
i = i + 1;
}
```
Другой вариант записи:
```
int sum = 0;
inti = 1;
for (; ;) {
if (i> 100)
break;
```

```
sum = sum + i;
i = i + 1;
}
```

Оператор **break** завершает выполнение цикла.

Оператор цикла dowhile. Все представленные выше операторы циклов, так или иначе, проверяют условие перед выполнением цикла, благодаря чему существует вероятность, что операторы внутри цикла никогда не будут выполнены. Такие циклы называют *циклы с предусловием*.

Однако бывают ситуации, когда целесообразно выполнять проверку условия после того, как будут выполнены операторы, стоящие внутри цикла. Это достигается путем использования операторов цикла *dowhile*, которые реализуют цикл с постусловием. Например, требуется прочитать символы с терминала до тех пор, пока не будет введен символ «звездочка».

```
charch;
do {
ch = getch();  // функциягetchвозвращает
// символ, введённый с
                // клавиатуры
} while (ch != '*');
```

В операторах while и do также можно использовать операторы break и continue.

Задания к лабораторной работе № 3

Построить блок-схему алгоритма для вычисления по варианту (табл. 13), написать на C++ программу с операторами цикла **while** и **for** и **dowhile**, полученный результат вывести на экран. Вариант соответствует номеру студента по списку группы. Ответить на контрольные вопросы. Составить отчёт по выполненной работе (см. рекомендации к оформлению отчёта по лабораторной работе).

Ответить на контрольные вопросы:

1. В чем отличия между операторами while и dowhile?
2. Что такое цикл с предусловием?
3. Что такое цикл с постусловием?

Таблица 13

Варианты заданий к лабораторной работе № 3

№ варианта	Операторы циклов while и for	Оператор цикла dowhile
1	Вычислить $\sum$ ^{2} с использованием оператора for	Ввести произвольные числа до тех пор, пока не будет введено число 0
2	Вычислить $f(x) = kx + b$, при x = 1,2,...,100 с использованием оператора while	Ввести произвольные символы до тех пор, пока не будет введен символ q
3	Вычислить S $\sum_{i=1}^{10} i^3$ пока S<80 с помощью цикла while	Подсчитать произведение 10 чисел, вводимых с клавиатуры
4	Найти сумму отрицательных значений функции Z = sin (5 - x) / cos (x - 2) для x, изменяющегося на отрезке [-5, 12] с шагом 1.	Подсчитать произведение5 чисел, введенных с клавиатуры
5	Найти сумму значений функции, больших 2, Z = sin (1/ x) + 5cos (1 /(x - 3)) + x для x, изменяющегося на отрезке [-3, 8] с шагом 1.	Вычислить модули введенных чисел до тех пор, пока пользователь не введет 0
6	Составить таблицу перевода дюймов в сантиметры для расстояний от 1 до 13 дюймов с шагом 1. 1дюйм= 2,54 см.	Определить знак введенных чисел до тех пор, пока пользователь не введет 0
7	Вычислить $f(x) = x2 + b$, при x = −10,−9,...,10с использованием оператора for	Определить минимальное введенное число из 10 чисел
8	Вычислить с помощью цикла for S	Определить максимальное введенное число из 9 чисел
9	Найти сумму значений функции y = cos (x/a) + x/(a-2) для x, изменяющегося от 2 до 13 с шагом 1 (a — произвольное число). Для организации циклов использовать оператор for.	Определить минимальное число среди положительных введенных 10 чисел
10	Вычислить $f(x) = 1/x$,$x \neq 0$ при x = −10,−9,...,10 с использованием оператора for	Определить максимальное число среди отрицательных введенных 7 чисел
11	Вычислить S $\sum$ $^2)$ пока S<50 с помощью цикла while	Определить среднее число из введенных 10 чисел
12	Вывести на печать значения функции, меньшие 2, Z = sin (1/ x) + 5cos (x - 3) + x для x, изменяющегося на отрезке [-7, 4] с шагом 1.	Определить среднее квадратичное из введенных 8 чисел
13	Вычислить $f(x) = (1+x)/(x-1)$, $x \neq 0$ при x = −20,−17,-14...,5 с использованием оператора for	Ввести произвольное число до тех пор, пока не будет введено число 0
14	Вычислить = $\sum_{i=1}^{\infty} Si^2$ пока S<50 с помощью цикла for	Ввести произвольный символ до тех пор, пока не будет введен символ q

15	Напечатать значения функции Y = tg (x/b) + x/(b - 2) для x, изменяющегося от 0 до 10 с шагом 1 (b – произвольное число).	Определить произведение нечетных чисел из 10 введенных
16	Напечатать значения функции z = 1/(x - 2) + 1/(x - 5) + 1n(12,8 - x) для x, изменяющегося на отрезке [-4, 14] с шагом 1.Для организации циклов использовать оператор for.	Определить сумму четных чисел из 10 введенных
17	Напечатать значения функции *z*= 1/(*x* - 2) + 1/(x - 5) + 1n(12,8 - *x)* для *x,* изменяющегося на отрезке [-4, 14] с шагом 1.	Ввести произвольное число до тех пор, пока не будет введено число 0
18	Вывести на печать отрицательные значения функции *z* = sin(5 - *x)*/cos(x - 2) для x, изменяющегося на отрезке [-6, 13] с шагом 1 (учесть область допустимых значений функции).	Ввести произвольный символ до тех пор, пока не будет введен символ q
19	Вывести на печать значения функции, меньшие 2, Z= sin(1/*x*) + 5cos(*x* - 3) +*x* для x, изменяющегося на отрезке [-7, 4] с шагом 1.	Определить произведение нечетных чисел из 10 введенных
20	Найти сумму отрицательных значений функции Z= sin(5 - *x*) / cos(*x*- 2) для *x,* изменяющегося на отрезке [-5, 12] с шагом 1.	Определить сумму четных чисел из 10 введенных

Для ввода программного кода и отладки программ можно использовать компиляторы онлайн:

https://www.onlinegdb.com/online_c++_compiler#;
https://www.jdoodle.com/online-compiler-c++/;

https://ideone.com/

ВОПРОСЫ К ЭКЗАМЕНУ

Тема 1. Значение моделирования, алгоритмизации и программирования при решении задач в профессиональной области

1.1 Этапы подготовки решения задач на компьютере.
1.2 Алгоритм, свойства алгоритма.
1.3 Способы описания алгоритмов.
1.4 Структурные схемы алгоритмов.

Тема 2. Программирование (языки программирования).

2.1 Алгоритм и программа.
2.2 Язык программирования.
2.3 Уровни языков программирования.
2.4 Поколения языков программирования.
2.5 Языки программирования высокого уровня.
2.6 Сущность трансляции.
2.7 Компиляторы и интерпретаторы.

2.8 Фазы трансляции и выполнения программы.
2.9 Лексический анализ.
2.10 Синтаксический анализ.
2.11 Семантический анализ.
2.12 Генерация кода.

Тема 3. Язык программирования Си++.

3.1 Имена, переменные константы.
- 3.1.1 Файл заголовков.
- 3.1.2 Обозначение имен.
- 3.1.3 Использование ключевых слов в идентификаторах имён.
- 3.1.4 Правила объявления переменных, операция присваивания.
- 3.1.5 Константы.
- 3.1.6 Определения выражения.

3.2 Конструкция языка Си++
- 3.2.1 Формы записи выражений. Инфиксная запись.
- 3.2.2 Формы записи выражений. Префиксная запись.
- 3.2.3 Формы записи выражений. Постфиксная запись.
- 3.2.4 Операция инкремент.
- 3.2.5 Операция декремент.
- 3.2.6 Операция присваивания.
- 3.2.7 Операции для работы с битами, сравнения и логические.
- 3.2.8 Порядок вычисления выражений.
- 3.2.9 Уровни приоритета операций.

3.3 Встроенные типы данных.
- 3.3.1 Представление целых чисел.
- 3.3.2 Преобразование типов.
- 3.3.3 Вещественные числа.
- 3.3.4 Логические величины.
- 3.3.5 Символы и байты.

3.4 Функции.
- 3.4.1 Объявление функции.
- 3.4.2 Определение функции.
- 3.4.3 Имена функций.
- 3.4.5 Необязательные аргументы функций.
- 3.4.6 Функции стандартного ввода-вывода**.**
- 3.4.7 Оператор включения.
- 3.4.8 Оператор извлечения.
- 3.4.9 Рекурсия.

3.5 Операторы.
- 3.5.1 Порядок выполнения.
- 3.5.2 Оператор-выражение.
- 3.5.3 Операторы объявления имен.
- 3.5.4 Операторы управления.
- 3.5.5 Условные операторы.
- 3.5.6 Оператор выбора.

3.5.7 Оператор цикла for.
3.5.8 Оператор цикла while.
3.5.9 Оператор перехода goto.

ПЕРЕЧЕНЬ УЧЕБНОЙ ЛИТЕРАТУРЫ ПО ДИСЦИПЛИНЕ

1. Канцедал С.А. Алгоритмизация и программирование: учебное пособие. М.: ФОРУМ: ИНФРА-М, 2021. 352 с. (Среднее профессиональное образование). ISBN 978-5-8199-0727-6.
2. Немцова Т.И. Программирование на языке высокого уровня. Программирование на языке С++: учебное пособие / Т.И. Немцова, С.Ю. Голова, А.И. Терентьев; под ред. Л.Г. Гагариной. М.: ФОРУМ: ИНФРА-М, 2021. 512 с. + Доп. материалы (Среднее профессиональное образование). ISBN 978-5-8199-0699-6. Текст: электронный. URL: https://znanium.com/catalog/product/1172261
3. Объектно-ориентированное программирование на С++: учебник / И.В. Баранова, С.Н. Баранов, И.В. Баженова [и др.]. Красноярск: Сиб. федер. ун-т, 2019. 288 с. ISBN 978-5-7638-4034-6. Текст: электронный. URL: https://znanium.com/catalog/product/1819676
4. Хорев П.Б. Объектно-ориентированное программирование с примерами на C#: учебное пособие. М.: ФОРУМ: ИНФРА-М, 2023. 200 с. (Среднее профессиональное образование). ISBN 978-5-00091-713-8. Текст: электронный. URL: https://znanium.com/catalog/product/1895650

Приложение

Автономное образовательное учреждение
высшего профессионального образования Ленинградской области
«Государственный институт экономики, финансов, права и технологий»

Кафедра информационных технологий, безопасности и права

Дисциплина «ПРОГРАММИРОВАНИЕ»

Лабораторная работа №

на тему

«_____»

(вариант № ___)

Выполнил(а) студент(ка)
___ курса
_____ группы
заочного отделения
Ф.И.О. студента

Проверил(а):
Ф.И.О. преподавателя

Гатчина
20__

Елена Владимировна Бенза,
кандидат технических наук,
Сергей Маркович Бенза

ПРОГРАММИРОВАНИЕ С++

Учебно-методическое пособие для выполнения лабораторных работ
для студентов, обучающихся по направлению подготовки
38.03.05 – «Бизнес-информатика»
(уровень бакалавриат)

Ответственный редактор В. Андронатий
Корректор Ю. Чиркова
Компьютерная верстка И. Иванова

Подписано в печать 07.10.2022 г. Формат 60x84 1/16

Усл.печ.л. 0,9 Тираж 550 экз. Заказ 1420

Издательство Государственного института экономики, финансов, права и технологий
188300 Ленинградская обл., г. Гатчина, ул. Рощинская, д. 5

Цена свободная