Лабораторная работа №3

Ввод-вывод данных с использованием библиотеки потокового ввода вывода

Механизм для ввода-вывода в C++ называется потоком, так как информация вводится и выводится в виде потока байтов – символ за символом.

Библиотека потоков ввода-вывода (iostream.h) определяет три глобальных объекта: cout, cin и cerr.

Для использования возможностей библиотеки необходимо в начале программы указать директиву ***using namespace*** std;

cout называется стандартным выводом, ***cin*** – стандартным вводом, ***cerr*** – стандартным потоком сообщений об ошибках. cout и cerr выводят на терминал и принадлежат к классу ostream, cin имеет тип istream и вводит с терминала.

Вывод осуществляется с помощью операции <<, ввод с помощью операции >>. Выражение

cout << "Пример вывода: " << 34;

напечатает на терминале строку "Пример вывода", за которым будет выведено число 34. Выражение

int x**;**

cin **>>** x**;**

введет целое число с терминала в переменную x. (Разумеется, для того, чтобы ввод произошел, на терминале нужно напечатать какое-либо число и нажать клавишу возврат каретки.)

#include <iostream> подключает библиотеку потокового ввода-вывода. Файл заголовков определяет глобальный объект этого класса cout. Объект называется глобальным, поскольку доступ к нему возможен из любой части программы. Этот объект выполняет вывод на консоль. В функции main мы можем к нему обратиться и послать ему сообщение:

```
#include "stdafx.h"
#include <iostream>
using namespace std;
int main()
{
cout << "Hello world!" << endl;
return 0;
}
```

Операция сдвига << определена как "вывести". Таким образом, программа посылает объекту *cout* сообщения "вывести строку Hello world!" и "вывести перевод строки" (*endl* обозначает перевод на новую строку). В ответ на эти сообщения объект *cout* выведет строку "Hello world!" на консоль и переведет курсор на следующую строку.

Подключение заголовочного файла #include "stdafx.h" не является обязательным с точки зрения языка C++, однако среда разработки Visual Studio 2015 требует его подключения для включения прекомпиляции заголовочных файлов. Данная возможность позволяет ускорить компиляцию и запуск программы.

Манипуляторы и форматирование ввода-вывода

Часто бывает необходимо вывести строку или число в определенном формате. Для этого используются так называемые манипуляторы.

Манипуляторы – это объекты особых типов, которые управляют тем, как обрабатывают последующие аргументы. Некоторые манипуляторы могут также выводить или вводить специальные символы. Манипуляторы позволяют задавать формат вывода чисел.

Таблица 1 – Манипуляторы потокового ввода-вывода Манипулятор	Назначение
endl	при выводе перейти на новую строку;
ends	вывести нулевой байт (признак конца строки);
flush	вывести и очистить все промежуточные буферы;
dec	выводить числа в десятичной системе (по умолчанию);
oct	выводить числа в восьмеричной системе;
hex	выводить числа в шестнадцатеричной системе счисления;
setw (int n)	установить ширину поля вывода в n символов (n – целое число);
setfill(int n)	установить символ-заполнитель, которым выводимое значение будет дополняться до необходимой ширины;
setprecision(int n)	установить количество цифр после запятой при выводе вещественных чисел;
setbase(int n)	установить систему счисления для вывода чисел; n может принимать значения 0, 2, 8, 10, 16, причем 0 означает систему счисления по умолчанию, т.е. 10.

Использовать манипуляторы просто – их надо вывести в выходной поток. Выведем одно и то же число в разных системах счисления:

```
int x=53;
cout <<"Десятичный вид: " << dec << x << endl
<< "Восьмиричный вид: " << oct << x << endl
<<"Шестнадцатиричный вид:" << hex << x << endl;
```

Аналогично используются манипуляторы с параметрами. Вывод числа с разным количеством цифр после запятой:

```
#include "stdafx.h"
#include <iostream>
#include <iomanip>
using namespace std;
int main()
{ const double d1 = 1.23456789;
const double d2 = 12.3456789;
const double d3 = 123.456789;
const double d4 = 1234.56789;
const double d5 = 12345.6789;
cout << endl << "setprecision(" << 3 << ")" << setprecision(3);
cout << endl << "default display" << endl;
cout << "d1 = " << d1 << endl;
cout << "d2 = " << d2 << endl;
cout << "d3 = " << d3 << endl;
cout << "d4 = " << d4 << endl;
cout << "d5 = " << d5 << endl;
return 0;
}
```

В результате получим:

d1 = 1,23 d2 = 12,3 d3 = 123 d4 = 1.23e+003 d5 = 1.23e+004
Те же манипуляторы (за исключением *endl* и *ends*) могут использоваться и при вводе. В этом случае они описывают представление вводимых чисел. Кроме того, имеется манипулятор, работающий только при вводе, это ws. Данный манипулятор переключает вводимый поток в такой режим, при котором все пробелы (включая табуляцию, переводы строки, переводы каретки и переводы страницы) будут вводиться. По умолчанию эти символы воспринимаются как разделители между атрибутами ввода.

```
int x;
// ввести шестнадцатеричное число
cin >> hex >> x;
```

Ввод вывод с использованием стандартной библиотеки ввода-вывода stdio.h
Все возможности организации ввода-вывода СИ реализованы в библиотечных функциях стандартной библиотеки stdio.h.
Для организации вывода используется функция
printf(форматная_строка,список_аргументов);
Форматная строка ограничивается кавычками «”» и может включать произвольный текст, управляющие символы и спецификации преобразования данных.
Список аргументов может отсутствовать.

```
#include "stdafx.h"
#include <stdio.h>
void main()
{
printf("\nhello!\n");
}
```

Препроцессорная директива *#include* <stdio.h> подключает стандартную библиотеку ввода-вывода. «\n» – перевод строки (управляющий символ).
При организации вывода данных на экран используются спецификации преобразования, которые имеют следующий обобщённый вид:
%флажки ширина_поля.точность модификатор спецификатор
Обязательными являются «%» и спецификатор.
Таблица 2 – Назначение флагов

Флаг	**Назначение**
-	Выравнивание результата по левому краю поля.
+	Результат всегда выводится с указанием знака «+» или «-».
Пробел	Если значение не отрицательное, то вместо плюса выводится пробел, для отрицательных значений выводится «-».
#	Аргументы могут быть преобразованы с использованием альтернативной формы

ширина_поля – целое положительное число, определяющее количество знакомест для вывода значения.
точность – целое положительное число, определяющее количество цифр после десятичной запятой для вывода значения с плавающей точкой.
Возможные модификаторы представлены в таблице 6.
Таблица 3 – Назначение модификаторов

Модификатор	Назначение
N	Для близкого указателя
F	Для дальнего указателя
h	Для значения short int
l	Для значения long
L	Для значения long double

Спецификаторы определяют тип выводимого значения и форму вывода.

Таблица 4 – Назначение спецификаторов

Спецификатор	**Тип аргумента**	**Назначение**
d	Целого типа	Для целых десятичных чисел (int)
i	Целого типа	Для целых десятичных чисел (int)
o	Целого типа	Для беззнаковых восьмеричных целых
u	Целого типа	Для беззнаковых десятичных целых
x	Целого типа	Для беззнаковых шестнадцатеричных целых (a,b,c,d,e,f)
X	Целого типа	Для беззнаковых шестнадцатеричных целых (A,B,C,D,E,F)
Спецификатор	Тип аргумента	Назначение
f	вещественный	Знаковое вещественное число в формате [+/-]ddd.dddd
e	вещественный	Знаковое вещественное число в формате [+/-]d.dddd или в экспоненциальной форме
g	вещественный	Знаковое вещественное число в формате или f, или e (в зависимости от выводимого значения)
E	вещественный	Такое же, как и e
G	вещественный	Такое же, как и g
s	строковый	ввод-вывод строковых данных
c	символьный	ввод-вывод символов

Например:
Printf(“|n summa=%f”,summa);
На экране будет выведено:
Summa=2102.3
После выполнения операторов:
float c=48.3, e=16.33;
int k=-83;
printf(“\nc=%f\tk=%d\te=%e”,c,k,e);
на экране будет выведено
c=48.299999 k=-83 e=1.63300e+01
Для тех же переменных:
printf(“\nc=%5.2f\tk=%5d\te=%8.2f\te=%11.4e”,c,k,e,e);
на экране будет выведено
c=48.30 k= -83 e=16.33 e= 1.6330e+01
В состав строки вывода могут входить управляющие последовательности:
‘*n*’ – перевод строки;
‘*t*’ – горизонтальная табуляция;
‘*r*’ – возврат каретки к началу строки;
‘\\’ – обратная косая черта \;
‘\’’ – апостроф ‘;
‘*0*’ – нулевой символ;
‘*a*’ – сигнал-звонок;
‘*b*’ – возврат на одну позицию;
‘*f*’ – перевод строки;
‘*v*’ – вертикальная табуляция;
‘*?*’ – знак вопроса.
Для организации ввода данных с клавиатуры используется функция

scanf(форматная_строка, список_аргументов);
Эта функция выполняет чтение значений вводимых с клавиатуры и присваивает их последовательно аргументам. Форматная строка представляет собой последовательность спецификаций, управляющих преобразованием вводимых значений.
*%*ширина_поля модификатор спецификатор*
''* в настоящее время не используется;
Ширина_поля – целое положительное число, позволяющее определить, какое количество байтов из входного потока соответствует вводимому значению.
модификатор и спецификатор – аналогичны функции *printf().*
Аргументами функции ввода могут быть адреса переменных, которым будут присвоены вводимые значения. Они задаются при помощи операции взятия адреса
"**&имя_переменной**"
Например:
scanf("%d%f%f",&n,&z,&x);
При организации ввода-вывода данных используются также функции, описанные в стандартной библиотеке ввода-вывода(*<stdio.h>):*
puts(const char* Строка); Выводит на экран строку символов и переводит курсор в начало следующей строки экрана. В качестве параметра функции можно использовать строковую константу или строковую переменную.
char *gets(char* s);Вводит с клавиатуры строку символов. Вводимая строка может содержать пробелы.
int putch(int c); Выводит на экран символ.
int getch(void);Возвращает код символа нажатой клавиши. Если нажата служебная клавиша, то функция *getch* возвращает 0. В этом случае, для того, чтобы определить, какая служебная клавиша нажата, нужно обратиться к функции *getch* еще раз. Замечание Функция *getch* не выводит на экран символ, соответствующий нажатой клавише.
cputs(const char* Строка); Выводит на экран строку.

Математические функции

Для выполнения математических вычислений в стандартной математической библиотеке ***<math.h>*** описаны следующие функции:
int abs (int к) ; double fabs(double x); Возвращает целое (*abs*) или дробное (*fabs*) абсолютное значение аргумента, в качестве которого можно использовать выражение соответствующего типа.
double acos (double x);
double asin (double x);
double atan (double x);
long double acosl(long double x) ;
long double asinl(long double x);
long double atanl(long double x);
Возвращает выраженную в радианах величину угла, арккосинус, арксинус или арктангенс которого передан соответствующей функции в качестве аргумента. Аргумент функции должен находиться в диапазоне от -1 до 1.
double cos (double x);
double sin (double x);
double tan (double x);
long double cosl(long double x);
long double sinl(long double x);
long double tanl(long double x);
Возвращает синус, косинус или тангенс угла. Величина угла должна быть задана в радианах.

```
#include "stdafx.h"
#include <stdio.h>
#include <math.h>
#include <iostream>
using namespace std;
int main(void)
{
double result;
double x = 0.5;
result = cos(x);
printf("Косинус числа %lf – %lf\n", x, result);
return 0;
}
```

double exp(double x); long double exp(long double lx); Возвращает значение, равное экспоненте аргумента (*e**, где *e* — основание натурального логарифма).

double pow (double x, double y); long double powl(long double (x), long double (y)); Возвращает значение, равное *xУ*.

```
#include "stdafx.h"
#include <stdio.h>
#include <math.h>
#include <iostream>
using namespace std;
int main(void)
{
double result;
double x = 4.0;
result = exp(x);
printf("'e' в степени %lf (e ^ %lf) = %lf\n", x, x, result);
return 0;
}
```

double sqrt(double κ);

Возвращает значение, равное квадратному корню из аргумента.

double log(double x);

double log10(double x);

long double logl(long double (x));

long double log10l(long double (x));

log, logl – возвращают значение натурального логарифма аргумента. *log10, log10l*– возвращают значение логарифма аргумента по основанию *10*.

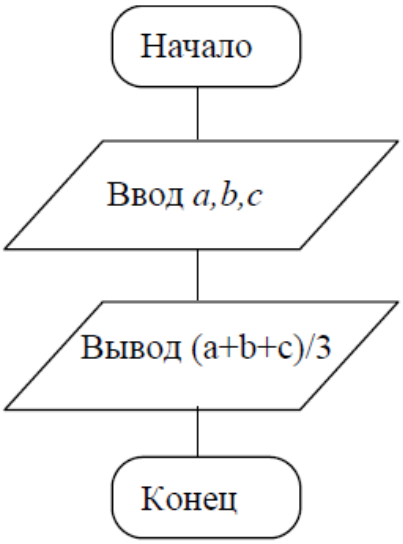
В библиотеке <*stdlib.h*> описаны генераторы случайных чисел.

int rand(void); Возвращает случайное целое число в диапазоне от 0 до *RAND_MAX*. Перед первым обращением к функции *rand* необходимо инициализировать генератор случайных чисел. Для этого надо вызвать функцию *srand*.

void srand(unsigned κ); Инициализирует генератор случайных чисел. Обычно в качестве параметра функции используют переменную, значение которой предсказать заранее нельзя, например это может быть текущее время.

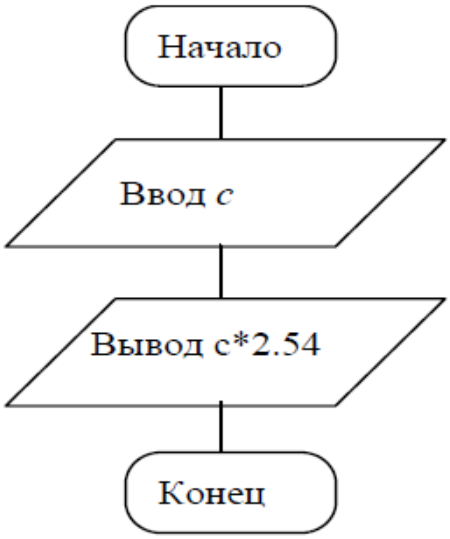
Примеры программ

Программа нахождения среднего арифметического из двух целых чисел и одного вещественного числа:




```
#include "stdafx.h"
#include <iostream>
using namespace std;
void main()
{
int a,b;
float c;
cout<<"Input 3 nambers"<<endl;
cin>>a>>b>>c;
cout<<"Rezult="<<(a+b+c)/3;
}
```

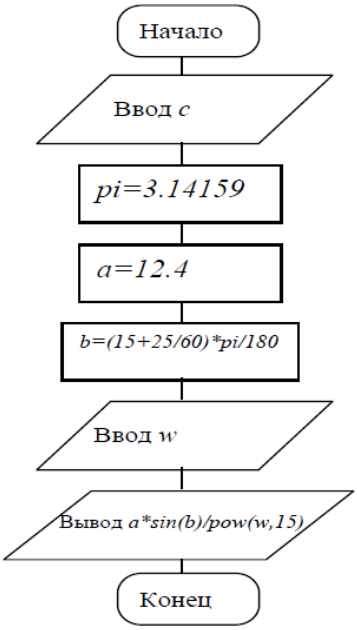
Программа перевода дюймов в сантиметры (1 дюйм = 2,54 см).




```
#include "stdafx.h"
#include <iostream>
using namespace std;
void main()
{
float c;
cout<<"Input nambe"<<endl;
cin>>c;
cout<<"Rezult="<< c*2.54;
}
```

Программа вычисления значения выражения: $Y = \frac{a \cdot \sin(b)}{w^{15}}$

a = 12.4, b =15 25 , a w – вводится с клавиатуры. Для возведения в степень используется функция pow заголовочного файла math.h.




```
#include "stdafx.h"
#include <iostream>
#include <math.h>
using namespace std;
void main()
{
const float pi=3.14159;
const float a=12.5;
const float b=(15+25/60)*pi/180;
float w;
cout<<"Input w"<<endl;
cin>>w;
cout<<"Rezult="<<a*sin(b)/pow(w,15);
}
```

Контрольные вопросы

1. Опишите структуру программы на языке С++.
2. Какие группы символов входят в алфавит языка С++.
3. Какие символы содержатся вы знаете.
4. Что такое управляющие последовательности, и каким образом они задаются?
5. Как задаются идентификаторы?
6. Перечислите ключевые слова языка С++.
7. Перечислите и опишите основные типы данных.
8. Как определить константу?
9. Опишите возможности ввода-вывода данных с помощью библиотеки потокового ввода вывода.
10. Опишите известные вам манипуляторы ввода-вывода.
11. Как производится ввод-вывод с использованием стандартной библиотеки ввода-вывода stdio.h.
12. Какие модификаторы и спецификаторы поддерживает функция printf.
13. Как осуществляется ввод при помощи стандартной библиотеки stdio.h.
14. Как подключить библиотеку с математическими функциями.
15. Какие стандартные математические функции содержит библиотека math.h.
16. Как получить случайное число.

double log10(double x);
long double logl(long double (x));
long double log10l(long double (x));
log, logl – возвращают значение натурального логарифма аргумента. *log10, log10l*– возвращают значение логарифма аргумента по основанию *10.*
В библиотеке <*stdlib.h*> описаны генераторы случайных чисел.
int rand(void); Возвращает случайное целое число в диапазоне от 0 до *RAND_MAX*. Перед первым обращением к функции *rand* необходимо инициализировать генератор случайных чисел. Для этого надо вызвать функцию *srand*. ***void srand(unsigned к);*** Инициализирует генератор случайных чисел. Обычно в качестве параметра функции используют переменную, значение которой предсказать заранее нельзя, например это может быть текущее время.

Примеры программ

Программа нахождения среднего арифметического из двух целых чисел и одного вещественного числа: Начало Ввод a,b,c Вы вод (a+b+c)/3 Конец
Рисунок 2.1 – Блок-схема алгоритма

```
#include "stdafx.h"
#include <iostream>
using namespace std;
void main()
{
int a,b;
float c;
cout<<"Input 3 nambers"<<endl;
cin>>a>>b>>c;
cout<<"Rezult="<<(a+b+c)/3;
}
```