**Лабораторная работа №1**

**Цель работы:** знакомство с основными типами данных и их объявление.

**Краткое теоретическое сведения**: каждое имя и каждое выражение имеет тип , определяющий операции, которые могут над ними производиться. Например, описание:

int inch;

Определяет, что inch имеет тип int ,то есть, inch является целой переменной. Описание –это оператор, которой вводить имя в программе. Описание создает тип этого имени. Тип определяет правильное использование имени или выражения. Для целых определены такие операции, как +,-,\* и /. После того, как включен файл stream.h, объект типа int может также быть вторым операндом <<, когда первый операнд оstream. Тип объекта определяет не только то, какие операции могут к нему применяться, но и смысл этих операций. Например, оператор:

cout<< inch << ” in=” << inch\*2.54<< ” cm/n”;

правильно обрабатывать четыре входных значений различным образом. Строки печатаются буквально, тогда как целое inch и значение с плавающей точкой inch\*2.54 преобразуются из их внутреннего представления в подходящее для человеческого глаза символьное представление. В С++ есть несколько основных типов и несколько способов создавать новые.

 **Основные типы данных**

 Основные (стандартные) типы можно использовать в арифметических операциях. Для описания основных типов определены следующие ключевые слова:

\* int (целый);

\* char (символьный);

\* wchar\_t (расширенный символьный);

\* bool (логический);

\* float (вещественный);

\* double (вещественный с двойной точностью).

 Первые четыре типа используются для представления целых, последние два – чисел с плавающей точкой. Переменная типа char имеет размер, естественный для хранения символа на данной машине (обычно, байт), а переменная типа int имеет размер, соответствующий целой арифметике на данной машине (обычно, слово). Диапазон целых чисел, которые могут быть представлены типом, зависит от его размера. В С++ размеры измеряются в единицах размера данных типа char, поэтому char по определению имеет размер единица. Соотношение между основными типами можно записать так:

 sizeof(float) <= sizeof(double) <= sizeof(long double)

 sizeof(char) <= sizeof(short) <= sizeof(int) <=sizeof(long)

Существует четыре спецификатора типа, уточняющих внутреннее представление и диапазон значений стандартных типов:

\* short (короткий)

\* long (длинный)

\* singed (знакомый)

\* unsigned (беззнаковый)

 К основному типу можно применять прилагательное const . Это дает тип, имеющий те же свойства, что и исходный тип, за исключением того, что значение переменных типа const не может изменяться после инициализации.

 const float pi=3.14; const char plus =’+’;

 Символ, заключенный в одинарные кавычки является, символьной константой. Заметьте, что часто константа, определенная таким образом, не занимает память; просто там, где требуется, ее значение может использоваться непосредственно. Константа должна инициализироваться при описании. Для переменных инициализация необязательна, но настоятельно рекомендуется. Оснований для введения локальной переменной без ее инициализации очень немного. К любой комбинации этих типов могут применяться арифметические операции:

 + (плюс, унарный и бинарный)

 - (минус, унарный и бинарный)

 \* (умножение)

 / (деление)

 А также операции сравнения:

 == (равно)

 != (не равно)

 < (меньше)

 > (больше)

 <= (меньше или равно)

 >= (больше или равно)

 Заметьте, что целое деление дает целый результат: 7/2 есть 3. Над целыми может выполняться операция % получение остатка: 7%2 равно 1. При присваивании и арифметических операциях С++ выполняет все осмысленные преобразования между основными типами, чтобы их можно было сочетать без ограничений:

 double d = 1; int i= 1; d = d + i; i = d + i;

**Практическая часть:**

**ЗАДАЧИ**

1. Объявите переменные, необходимые для вычисления площади прямоугольника.
2. Объявите переменные, необходимые для пересчета веса из фунтов в килограммы.
3. Определите исходные данные и объявите переменные, необходимые для вычисления дохода по вкладу.
4. Объявите переменные, необходимые для вычисления площади круга.
5. Объявите переменные, необходимые для вычисления площади кольца.
6. Объявите переменные, необходимо для вычисления объема и площади поверхности цилиндра.
7. Объявите переменные, необходимые для вычисления стоимости покупки, состоящей из нескольких тетрадей, карандашей и линейки.
8. Объявите переменные, необходимые для вычисления стоимости покупки, состоящей из нескольких тетрадей, и такого же количества обложек.

 **Лабораторная работа №2**

 **Организация форматированного ввода-вывода**

 **Цель работы:**

 Ознакомление со структурой программы на С++ и классами потоками ввода/вывода.

 **Краткое теоретическое сведения:**

 Язык С++поддерживает потоковую систему ввода-вывода. Определение своей системы ввода-вывода в С++ связано с тем, что в С не было механизма, обеспечивающего ввод-вывод типов данных, определенных пользователем. Поддержка потокового ввода-вывода обеспечивается в заголовочном файле iostream.h

**Классы потоков:**

Для создания потоков, обеспечивающих ввода-вывод используется следующие классы:

istream -поток ввода;

ostream -поток вывода;

iostream -поток ввода-вывода;

 **Преопределенные потоки.**

 Для консольного ввода-вывода существуют предопределенные потоки С++:

cin- поток, ассоциированный со стандартным вводом;

cout- поток, ассоциированный со стандартным выводом.

 **Использование потокового ввода-вывода:**

**Пример1**:

Написать программу пересчета величины временного интервала, заданного в минутах, в величину ,выраженную в часах и в минутах. Рекомендуемый вид экрана во время выполнения программы:

Ведите временной интервал- >**150**

15мин-это 2ч.30мин.

#include <iostream.h>

Int main (){

Int min,h,m;

сout<<”введите временной интервал->”;

сin>>min;

H=min%60;

M=min%60;

Cout<<min<<“мин-это“<<h<<“час“<<м<<“мин“:

Return 0;

}

 **Практическая часть:**

ЗАДАЧИ

1. Написать программу вычисления площади параллелограмма. Ниже приведен рекомендуемый вид экрана во время выполнения программы (данные, введенные пользователем, выделены полужирным шрифтом).

Вычисление объема прямоугольника

Введите исходные данные:

Длина (см)-**9**

Ширина (см)-**7.5**

Площадь параллелограмма:67.50 кв. см.

1. Написать программу вычисления объема параллелепипеда. Ниже приведен рекомендуемый вид экрана во время выполнения программы (данные, введенные пользователем, выделены полужирными шрифтом ).

Вычисление объема параллелепипеда.

Введите исходные данные:

Длина(см)-**9**

Ширина(см)-**7,5**

Высота(см)-**5**

Объем:337,50 куб. см.

1. Написать программу вычисления площади поверхности параллелепипеда. Ниже приведен рекомендуемый вид экрана во время выполнение программы (данные, введенные пользователем , выделены полужирным шрифтом).

Вычисление площади поверхности параллелепипеда.

Введите исходные данные:

Длина(см)-**9**

Ширина(см)-**7,5**

Высота(см)-**5**

Площадь поверхности:90,00куб.см

1. Написать программу вычисления объема куба. Ниже приведен рекомендуемый вид экрана во время выполнения программы (данные, , введенные пользователем , выделены полужирным шрифтом).

Вычисление объема куба.

Введите длину ребра (см) и нажмите клавишу <Enter>**9.5**

Объем куба:857.38 куб.см

1. Написать программму вычисление объема цилиндра. Ниже приведен рекомендуемый вид экрана во время выполнения программы (данные, , введенные пользователем , выделены полужирным шрифтом) .

Вычисление объема цилиндра.

Введите исходные данные:

Радиус основания(см)-**5**

Высота цилиндра(см)-**10**

Объем цилиндра 1570.80см.куб

Для завершения нажмите <Enter>

1. Написать программу вычисления стоимости покупки, состоящей из нескольких тетрадей и карандашей. Ниже приведен рекомендуемый вид экрана во время выполнение программы (данные, введенные пользователем , выделены полужирным шрифтом) .

Вычисление стоимости покупки.

Введите исходные данные:

Цена тетради(руб)-**2,75**

Количество тетрадей-**5**

Цена карандаша(руб)-**0,85**

Количество карандашей-**2**

Стоимости покупки :15,45сом

1. Написать программу вычисления стоимости покупки, состоящей из нескольких тетрадей и такого же количество обложек к ним. Ниже проведен рекомендуемый вид экрана во время выполнения программы (данные, , введенные пользователем , выделены полужирным шрифтом) .

Вычисление стоимости покупки.

Введите исходные данные:

Цена тетради(руб)-**2,75**

Цена обложки(руб)**0,5**

Количество комплектов(шт)-7

Стоимости покупки:22,75

1. Написать программу вычисления стоимости некоторого количества (по весу) яблок. Ниже проведен рекомендуемый вид экрана во время выполнения программы (данные, , введенные пользователем , выделены полужирным шрифтом) .

Вычисление стоимости покупки.

Введите исходные данные

Цена одного килограмма яблок (руб)-**8,5**

Вес яблок (кг)-**2,3**

Стоимости покупки:19,55руб

1. Написать программу вычисления площади треугольника, если известна длина основания и высоты. Ниже проведен рекомендуемый вид экрана во время выполнения программы (данные, , введенные пользователем , выделены полужирным шрифтом) .

Вычисление площади треугольника .

Введите исходные данные:

Основание(см)-**8,5**

Высоты(см)-**10**

Площадь треугольника 42,50 кв. см

1. Написать программу вычисления площади треугольника если известны длина двух его сторон и величина угла между этими сторонами. Ниже проведен рекомендуемый вид экрана во время выполнения программы (данные, , введенные пользователем , выделены полужирным шрифтом) .

Вычисление площади треугольника .

Введите (через пробел) длина двух сторон(см) треугольника-**30**

Площадь треугольника:106,25кв.см.

1. Написать программу вычисления сопротивления электрической цепи, состоящей из двух параллельно соединенных сопротивлений. Ниже проведен рекомендуемый вид экрана во время выполнения программы (данные, , введенные пользователем , выделены полужирным шрифтом) .

Вычисление сопротивление электрической цепи.

При параллельном соединении элементов.

Введите исходные данные:

Величина первого сопротивления(ом) -**15**

Величина второго сопротивления(ом) -**20**

Сопротивление цепи:8,57 ом

1. Написать программу вычисления сопротивления электрической цепи, состоящей из двух последовательно соединенных сопротивлений. . Ниже проведен рекомендуемый вид экрана во время выполнения программы (данные, , введенные пользователем , выделены полужирным шрифтом) .

Вычисление сопротивление электрической цепи.

Введите исходные данные:

Величина первого сопротивления(ом) -**15**

Величина второго сопротивления(ом) -**27,3**

Сопротивление цепи(последовательное соединение):42,30 ом

 **Лабораторная работа №3**

 **Математические функции**

**Цель работы:**

Освоение студентами работу с математическими функциями.

**Сведения**:

**Заголовочный файл <math.h> (<cmath.h>)- математические функции.**

|  |  |
| --- | --- |
| Acos | Возвращает арккосинус аргумента |
| Asin | Возвращает арксинус аргумента |
| Atan | Возвращает арктангенс аргумента |
| atan2 | Возвращает арктангенс отношения аргумента |
| Ceil | Округляет вверх |
| Cos | Вычисляет косинус |
| Cosh | Вычисляет гиперболический косинус |
| Exp | Возвращает степень числа е |
| Fabs | Возвращает модуль числа |
| Floor | Округляет вниз |
| Fmod | Возвращает остаток от деления x на y. |
| Frexp | Выделяет из числа мантиссу и экспоненциальную часть |
| Ldexp | Преобразует мантиссу и показатель степени в число |
| Log | Вычисляет натуральный логарифм |
| log10 | Вычисляет логарифм по основанию 10 |
| Modf | Разбивает число на целую и дробную части |
| Pow | Возводит число в степень |
| Sin | Вычисляет синус |
| Sinh | Вычисляет гиперболический синус |
| Sqrt | Вычисляет квадратный корень |
| Tan | Возвращает тангенс аргумента |
| tanh | Возвращает гиперболический тангенс аргумента |
|  |  |

**Задания**

Написать программу для расчета по двум формулам. Предварительно подготовьте тестовые примеры по второй формуле с помощью калькулятора (результат вычисления по первой формуле должен совпадать со второй). Для их использования необходимо подключить к программе заголовочный файл <math.h> или <cmath.h>.

**Вариант №1**

**z1=2**$sin^{2}$**(3**$π-2α$**)**$cos^{2}$**(**$5π+2α$**)**

**z2=**$ \frac{1}{4}-\frac{1}{4}$**sin**$(\frac{5}{2}π-8α)$

**Вариант №2**

**z1=cos**$α+sinα$**+cos3**$α$**+sin3**$α$

**z2=2**$√2$**cos**$α∙ $**sin(**$\frac{π}{4}+2α)$

**Вариант №3**

**z1=**$\frac{sin2α+sin5α-sin3α}{cosα+1-2sin^{2}2α}$

**z2=2sin**$α$

**Вариант №4**

**z1=**$\frac{sin2α+sin5α-sin3α}{ cosα-cos3α+cos5α}$

**z2= tg3**$α$

**Вариант №5**

**z1=1**$-\frac{1}{4}sin^{2}$ **2**$α$**+cos2**$ α$

**z2=**$cos^{2}α$**+**$cos^{4}α$

**Вариант №6**

**z1=cos**$α$**+cos2**$α$**+cos6**$α+$**cos7**$α$

**z2=4 cos** $\frac{α}{2}$$∙$**cos** $\frac{5}{2}α∙$**cos4**$α$

**Вариант №7**

**z1 =**$ cos^{2}$**(**$\frac{3}{8}π-\frac{α}{4}$**)**$-cos^{2}$**(**$\frac{11}{8}π+\frac{α}{4}$**)**

**z2=**$\frac{√2}{2}$**sin**$\frac{α}{2}$

**Вариант №8**

**z1=**$cos^{4}x+sin^{2}y+\frac{1}{4}sin^{2}2x-1$

**z2=sin(y+x)**$∙$**sin(y-x)**

**Вариант №9**

**z1=** $(cosα-cosβ)^{2}-(sinα-sinβ)^{2}$

**z2=**$-4sin^{2}\frac{α-β}{2}∙cos⁡(α+β)$

**Вариант №10**

**z1=**$\frac{sin\frac{π}{2}+3α}{1-sin⁡(3α-π)}$

**z2=ctg**$(\frac{5}{4}π+\frac{3}{2}α)$

**Вариант №11**

**z1=**$\frac{1-2sin^{2}α}{1+sin2α}$

**z2=**$\frac{1-tgα}{1+tgα}$

**Вариант №12**

**z1=**$\frac{sin4α}{1+cos4α}∙\frac{cos2α}{1+cos2α}$

**z2= ctg(**$\frac{3}{2}π-α)$

**Вариант №13**

**z1=**$\frac{sinα+cos⁡(2β-α)}{cosα-sin(2β-α)}$

**z2=**$\frac{1+sin2β}{cos2β}$

**Вариант №14**

**z1=**$\frac{cosα+sinα}{cosα-sinα}$

**z2=**$tg2α+sec2α$

 **Лабораторная работа№4**

**Программирование на Си++ с использованием условных операторов и операторов цикла.**

**Цель работы**: освоение общих принципов работы с использованием операторов управления программой.

**Краткое теоретическое сведения**: В Си++ ложь означает ноль, истину-любое ненулевое значение. Специальный тип хранение истины и лжи в языке Си отсутствует. (В С++ такой тип есть, но вышеприведенный способ представления результатов операций поддерживается)

**Оператор ветвления: if,switch**

**Условный оператор if**

Условный оператор **if** используется для разветвления процесса вычислений на два направления (рис1).

 true false

оператор

оператор

 **Рис.1**

 **Синтаксис оператора if:**

If(выражение){

Последовательность операторов:

}

Else

{

Последовательность операторов:

}

**Пример2**. Производится выстрел по мишени, изображенной на рис.2. Определить количество очков.



 **Рис.2. Мишень .**

#include <iostream.h>

int main()

{

 float x ,y; int kol;

 cout <<”введите координаты выстрела\n”;

 cin>>x>>y;

 if (x\*x+y\*y<1) kol =2;

 else if (x\*x+y\*y<4) kol = 1;

 else kol = 0;

 cout <<”\n очков: “<<kol;

 return 0;

}

**Практическая часть:**

ЗАДАЧИ

1. Написать программу, которая вычисляет частное двух чисел. Программа должна проверять правильность введенных пользователем данных и, если они неверные (делитель равен нулю), выдавать сообщение об ошибке. Ниже приведен рекомендуемый вид экран во время выполнения программы (данные, введенные пользователем, выделены полужирным шрифтом ).

Вычисление частного.

Введите в одной строке делимое и делитель, затем нажмите <Enter>.-12 0

Вы ошиблись. Делитель не должен быть равен нулю.

1. Написать программу вычисления площади кольца. Программа должна проверять правильность исходных данных. Ниже приведен рекомендуемый вид экран во время выполнения программы (данные, введенные пользователем, выделены полужирным шрифтом).

Вычисление площади кольца.

Введите исходные данные:

Радиус кольца(см)-**3,5**

Радиус отверстия (см)-**7**

Ошибка! Радиус отверстия не может быть больше радиуса кольца.

1. Написать программу, которая переводит время из минут и секунд в секунды. Программа должна проверять правильность введенных пользователем данных и в случае , если данные неверные, выводить соответствующее сообщение . рекомендуемый вид экрана во время выполнение программы приведен ниже ошибочные данные , введенные пользователем, выделены полужирным шрифтом.

Введите время (минут секунд)-**2,90**

Ошибка! Количество секунд не может быть больше 60

Для завершения нажмите <Enter>.

1. Написать программу, которая проверяет, является ли год високосным. Ниже приведен рекомендуемый вид экрана во время работы программы . данные введенные пользователем , выделены полужирным шрифтом.

Введите год, например 2000, и нажмите <Enter>.

-2001

201 год –не високосный

Для завершения нажмите <Enter>.

1. Написать программу вычисления сопротивления электрической цепи, состоящей из двух сопротивлений. Сопротивления могут быть соединены последовательно или параллельно. Ниже проведен рекомендуемый вид экрана во время выполнение программы (данные, введенные пользователем, выделены полужирным шрифтом).

Вычисление сопротивления электрической цепи.

Введите исходные данные:

величина первого сопротивления(ом)-**15**

величина второго сопротивления(ом)-**27,3**

тип соединения (1-последовательное,2-параллельное)-2

сопротивление цепи:9,68 Ом

1. Написать программу решение квадратного уравнения. Программа должна проверять правильность исходных данных и в случае, если коэффициент при второй степени неизвестного равен нулю , выводить соответствующее сообщение . Ниже проведен рекомендуемый вид экрана во время выполнение программы (данные, введенные пользователем, выделены полужирными шрифтом).

Решение квадратного уравнения.

Введите в одной строке значения коэффициентов нажмите <Enter>.

-12 27 -10

Корни уравнения:

Х1=-25,551

Х2=-28,449

1. Написать программу вычисления стоимости покупки с учетом скидки. Скидка в 10% предоставляется, если сумма пакупки больше 1000руб. Ниже проведен рекомендуемый вид экрана во время выполнение программы (данные, введенные пользователем, выделены полужирными шрифтом).

Вычислите стоимость покупки с учетом скидки.

Введите сумму покупки и нажмите <Enter>.

-1200

Вам предоставляется скидка 10%

Сумма покупки с учетом скидки:1080,00руб

1. Написать программу вычисления стоимости покупки с учетом скидки. Скидка в 3% предоставляется, если сумма покупки больше 500руб., в 5%-если сумма больше 1000руб. Ниже проведен рекомендуемый вид экрана во время выполнение программы (данные, введенные пользователем, выделены полужирным шрифтом).

Вычислите стоимость покупки с учетом скидки.

Введите сумму покупки и нажмите <Enter>.

-640

Вам предоставляется скидка 3%

Сумма покупки с учетом скидки:620,80руб

1. Написать программу проверки знание даты основания Санкт-Петербурга. В случае неправильного ответа пользователя, программа должна выводить правильный ответ. Ниже проведен рекомендуемый вид экрана во время выполнение программы (данные, введенные пользователем, выделены полужирным шрифтом).

В каком году был основан Санкт-Петербург?

Введите число и нажмите <Enter>.

-1703

Вы ошиблись, Санкт-Петербург был основан в 1703году.

1. Написать программу проверки знание даты начало второй мировой войны. В случае правильного ответа пользователя, программа должна выводить правильный ответ. Ниже проведен рекомендуемый вид экрана во время выполнение программы (данные, введенные пользователем, выделены полужирным шрифтом).

В каком году началась вторая мировая война?

Введите число и нажмите <Enter>.

-1939

Правильно.

1. Напишите программу проверки знания истории архитектуры . программа должна вести вопрос и три варианта ответа. Пользователь должна выбрать правильный ответ и вести его номер. Ниже проведен рекомендуемый вид экрана во время выполнение программы (данные, введенные пользователем, выделены полужирным шрифтом).

Архитектор Исаакиевского собора:

1.Доменико Трезини

2.Огюст Монферран

3.Карл Росси

Введите номер правильного ответа и нажмите <Enter>.

-**3**

Вы ошиблись.

Архитектор Исаакиевского собора - Огюст Монферран

1. Напишите программу проверки знания истории архитектуры. Программа должна вести вопрос и три варианта ответа. Пользователь должна выбрать правильный ответ и вести его номер. Ниже проведен рекомендуемый вид экрана во время выполнение программы (данные, введенные пользователем, выделены полужирным шрифтом).

Невский проспект получил свое название:

По имени реки, на берегах которой расположен Санкт-Петербург

По имени близлежащего монастыря Александро-Невской лавры

В память о знаменитом полководце Александре Невском

Введите номер правильного ответа и нажмите <Enter>.

**-1**

Вы ошиблись. Правильный ответ:2

 **Оператор SWITCH**

Оператор **SWITCH**(переключатель) предназначен для разветвления процесса вычислений на несколько направлений (рис.3)

 switch

 Case1

Оператор1

выражение

default

Операторn

Оператор2

 Case2

 Case n

 default

 рис.3

  **Синтаксис оператора**  **SWITCH:**

switch(выражение)

{

case константа1:

последовательность операторов

break;

case константа2:

последовательность операторов

break;

………

default:

последовательность операторов

}

 Оператор break применяется для выхода из оператора switch. Константа в вариантах case должны быть различными, и если проверяемое значение не совпадает ни с одной из констант, выбирается вариант default.

 **Привет3**.(программа реализует простейший калькулятор на 4действия):

#include <iostream.h>

int main() {

 int a, b, res;

 char op;

 cout << “\n Введите 1й операнд:”; cin>>a;

 cout << “\n Введите знак операции:”; cin>>op;

 cout << “\n Введите 2й операнд:”; cin>>b;

 bool f= true;

 switch (op)

 {

 case ‘+’: res = a+b; break;

 case ‘-’: res = a-b; break;

 case ‘\*’: res = a\*b; break;

 case ‘/’: res = a/b; break;

 default : cout << “\неизвестная операция” ; f = false;

 }

 if (f) cout << “\nРезультат :” <<res;

 return 0;

 }

 **Практическая часть :**

**ЗАДАЧИ**

1. Напишите программу, которая запрашивает у пользователя номер дня недели, затем выводит название дня недели или сообщение об ошибке , если введены неверные данные.
2. Напишите программу, которая вычисляет стоимость междугородного телефонного разговора (цена одной минуты определяется расстоянием до города, в котором находится абонент). Исходным данными для программы является код города и длительность разговора. На таб.1 приведены коды некоторых городов и рекомендуемый вид экрана во время работы программы (данные, введенные пользователем, выделены полужирным шрифтом).

|  |  |  |
| --- | --- | --- |
| Город  | Код  | Цена минуты (руб) |
| Владивосток  | 423 | 2,2 |
| Москва  | 095 | 1,0 |
| Мурманск  | 815 | 1,2 |
| Самара  | 846 | 1,4 |

Вычисление стоимости разговора по телефону.

Введите исходные данные:

Код города- **423**

Длительность (целое количество минут) -**3**

Город: Владивосток

Цена минуты:2,20руб.

1. Напишите программу, которая по дате определяет день недели, по который это дата приходиться. Для вычисления дня недели воспользуйтесь формулой:

(d+[1/5(13m-1)]+Y[Y/4]+[c/4]-2c+777)mod7.

Здесь d-число месяца, m-номер месяца, если начинать счет с марта, как это делали в Древнем Риме, Y-номер года в столетии, с-количество столетии. Квадратные скобки означают , что надо взять целую часть от значение, находящегося в скобках. Вычисление по формуле значение определяет день недели: 1-понедельник, 2- вторник, …,6-суббота, 0-воскресенье.

 **Операторы цикла: for, while, do/while.**

 *Цикл с параметром (****for****)*

 *Синтаксис оператора* ***for***

for(инициализация: условие: увеличение)

{

Последовательность операторов;

}

Оператор **for имеет**  три главные части:

Инициализация- место, где обычно находится оператор присваивания, используемый для установки начального значения переменной цикла:

Условие- место ,где находится выражение, определяющее условие работы цикла:

Увеличение-место, где определяется характер изменение переменной цикла на каждый интеграции.

**Пример4.** (программа находит все делители целого положительного числа):

#include <iostream.h>

int main() {

int num, half, div;

cout<<”\n введите число: “; cin>>num;

for (half = num/2, div = 2; div<=half; div++)

 if (!(num%div)) cout <<div<<”n”;

return 0;

}

**ЗАДАЧИ**

1. Написать программу, которая выводит на экран ваша имя и фамилия 10 раз.
2. Написать программу, которая выводит таблицу квадратов первых целых положительных чисел. Ниже приведен рекомендуемый вид экрана во время работы программы.

Таблица квадратов

\_ \_ \_ \_ \_ \_ \_ \_

Число квадратов

\_ \_ \_ \_ \_ \_ \_ \_ \_

1. 1
2. 4
3. 9
4. 16

5 25

6 36

7 49

8 64

9 81

10 100

­\_ \_ \_ \_ \_ \_ \_ \_ \_ \_

1. Написать программу, которая выводит таблицу квадратов первых пять целых положительных нечетных чисел. Ниже приведен рекомендуемый вид экрана во время работы программы.

Таблица квадратных нечетных чисел.

\_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_

Число Квадрат

\_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_

1. 1
2. 9
3. 25
4. 49
5. 81

\_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_

1. Написать программу , которая вычисляет сумму первых n целых положительных чисел количество суммируемых чисел должно выводиться во время работы программы. Ниже проведен рекомендуемый вид экрана во время выполнение программы (данные, введенные пользователем, выделены полужирными шрифтом).

Введите количество суммируемых чисел-**20**

Сумма первых 20 положительных чисел равно 210

1. Написать программу , которая вычисляет сумму первых n целых положительных чисел количество суммируемых чисел должно выводиться во время работы программы. Ниже проведен рекомендуемый вид экрана во время выполнение программы (данные, введенные пользователем, выделены полужирными шрифтом).

Вычисление суммы положительных четных чисел.

Введите количество суммируемых чисел и нажмите <Enter>.

-**12**

Сумма первых 20положительных чисел равна156

1. Написать программу , которая вычисляет сумму первых n членов ряда: 1,3,5,7,….. количество суммируемых членов ряда задается во время работы программы. Ниже проведен рекомендуемый вид экрана во время выполнение программы (данные, введенные пользователем, выделены полужирными шрифтом).

Вычисление частичной суммы ряда: 1,3,5, 7….

Введите количество суммируемых членов ряда -**15**

Сумма первых 15 членов ряда равно330

1. Написать программу , которая вычисляет сумму первых n членов ряда: 1+1/2+1/3+1/4+…. Количество суммируемых членов ряда задается во время работы программы. Ниже проведен рекомендуемый вид экрана во время выполнение программы (данные, введенные пользователем, выделены полужирным шрифтом).

Вычисление частичной суммы ряда: 1+1/2+1/3+1/4+….

Введите количество суммируемых членов ряда -**15**

Сумма первых 15 членов ряда равно3,3182

1. Написать программу , которая выводит таблицу степеней двойки от нулевой до десяти. Ниже проведен рекомендуемый вид экрана во время выполнение программы.

 Таблица степеней двойки

\_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_

1. 1
2. 2
3. 4
4. 8
5. 16
6. 32
7. 64
8. 128
9. 256
10. 512
11. 1024

 \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_

1. Написать программу, которая вычисляет факториал введенного с клавиатуры чисел (факториалом чисел **n** называется произведение целых чисел от1 до n. Например, факториал 1 равен 1,8-40320).

Вычисление факториала.

Введите число, факториал которого надо вычислить-7

Факториал 7 равен 5040.

1. Написать программу , которая выводит таблицу значений функции у=-2,4х

**Оператор WHILE**

**Синтаксис оператора WHILE**

While (условие)

{

Последовательность операторов:

}

**Пример5**. (программа находит все делители целого положительного числа):

#include <iostream.h>

int main()

{

int num;

cout<<”\n введите число: “; cin>>num;

int=num/2; // половина числа

int div = 2; //кандидат на делитель

while (div <=half) {

 if (! (num % div)) cout <<div<<”\n”;

 div++;

 }

 return 0;

 }

 **ЗАДАЧИ**

1. Написать программу, которая определяет минимальное число во введенной с клавиатуры последовательности положительных чисел (длина последовательности неограниченна). Ниже проведен рекомендуемый вид экрана во время выполнение программы (данные, введенные пользователем, выделены полужирными шрифтом).

Определение минимального числа последовательности положительных чисел.

Вводите после стрелки числа. Для завершения вводите ноль.

-12

-75

-10

-9

-23

-0

Минимальное число: 9

1. Написать программу, которая проверяет, является ли введенное пользователем целое число простым. Рекомендуемый вид экран во время выполнение программы ниже. Данные, введенные пользователем, выделены полужирным шрифтом

Введите целое число и нажмите <Enter>.

-45

45-не простое число.

1. Напишите программу, которая выводит таблицу умножения.

 **Оператор DO/WHILE**

Синтаксис оператора **Do/while**

 do

 {

 последовательность операторов;

 }

 while (условие):

Пример использования оператора **do/while:**

**Пример 6**. Программа ожидания нажатия клавиши ‘a’:

#include <iostream.h>

int main()

{char ch;

do

{

ch=getch();

}

while(ch!=’a’)

return 0;

}

**ЗАДАЧИ**

1. Написать программу, вычисляющую сумму и среднюю арифметическую последовательность положительных чисел, которые вводятся с клавиатуры.

Ниже проведен рекомендуемый вид экрана во время выполнение программы (данные, введенные пользователем, выделены полужирным шрифтом).

Вычисление среднего арифметического последовательности положительных чисел.

Вводите после стрелки числа. Для завершения вводите ноль.

-45

-23

-15

-0

Введено чисел:3

Сумма чисел:83

Среднее арифметическое:27.67

1. Написать программу, которая определяет максимальное число из введенной с клавиатуры последовательных положительных чисел (длина последовательности неограниченна). Ниже проведен рекомендуемый вид экрана во время выполнение программы (данные, введенные пользователем, выделены полужирным шрифтом).

Определение максимального числа последовательности положительных чисел.

Вводите после стрелки числа. Для завершения вводите ноль.

-56

-75

-10

-9

-23

-0

Максимальное число:75

1. Написать программу, которая определяет минимальное число из введенной с клавиатуры последовательных положительных чисел (длина последовательности неограниченна). Ниже проведен рекомендуемый вид экрана во время выполнение программы (данные, введенные пользователем, выделены полужирным шрифтом).

Определение минимальное числа последовательности положительных чисел.

Вводите после стрелки числа. Для завершения вводите ноль.

-12

-75

-10

-9

-23

-0

Минимальное число:9

1. Напишите программу, которая проверяет, является ли введенное пользователем целое число простым. Рекомендуемый вид экрана во время выполнения программы приведен ниже. Данные, введенные пользователем, выделены полужирным шрифтом.

Введите целое и нажмите <Enter>.

-45

45-не простое число.

 **Лабораторная работа №5**

 **Работа с массивами**

**Цель работы**: освоение студентами приемов программирования с использованием массивов.

**Сведение из теории**: как известно, **массив** - это конечное совокупность данных одного типа. Можно говорить о массивах целых чисел, массивов символов и т.д. Мы можем даже определить массив, элементы которого – массивы (массив массивов),определяя, таким образом, многомерные массивы. Любой массив в программе должен быть описан: после имени массива добавляют квадратные скобки [],внутри которых обычно стоит число, показывающее количество элементов массива.

Например**, int x[10];//**описания массива из 10 целых чисел;

 **float a[10];//**описания массива из 10 вещественных чисел.

 В случае многомерных массивов показывает столько пар скобок, какова размерность массива, а число внутри скобок показывает размер массива по данному измерению. Например, описание двумерного массива выглядит так: **int a[2][5];.** Такое описание можно трактовать как матрицу из 2 строк и 5сталбцов. Элементы массива нумеруются с нуля. Инициализирующие значения для массивов записываются в фигурных скобках. Значение элементом присваиваются по порядку. Если элементов в массиве больше, чем инициализаторов, элементы, для которых значение не указаны, обнуляются:

 **int b[5]** = {3, 2, 1}; **// b[0]=**3, **b[1]=**2, **b[2]=**1, **b[3**]=0**, b[4]=**0.

Индекс может быть не только целое константой или целой переменной, но и любым выражением целого типа. Переменная с индексами в программе используется наравне с простой переменной (например, в операторе присваивание, в функция ввода-вывода) быстро.

 В таб.2приведены примеры основных приемов работы с массивами:

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Одномерные массивы** | **Двумерные массивы** | **Строки** |
| Объявление массива | тип имя \_переменной [размер] Пример:int m[10]; | тип имя \_переменной [размер][размер]Пример:int mm[10] [15]; | char имя \_переменной [размер]Пример:char s[7]; |
| Инициализация при объявлении | int m[5]={9,7,8,2,3}; | int mm[2][3]={7,5,8,3,1,2}; | char s[6]=”hello”,либоchar s[6]={‘h’,’e’,’l’,’l’,’0’}; |
| Инициализация безразмерного массива | intm[]={9,7,8,2,3}; | int mm[][3]={7,5,8,3,1,2}; | char s[]=”hello”, |
| Организация доступа к элементом массива | m[0],m[1],m[2],m[3],m[4] (первый элемент с нулевым индексом) | mm[0],mm[0][1],mm[0][2], mm[1][0],mm[1][1],mm[1][2] | аналогично одномерному массиву: в s[0]-ASCII код символа ‘h’ |

Общие замечания:

 Приступая решению задач по данной работе, следует вспомнить , что:

**МАССИВ**-это структура данных, представляющая собой набор, совокупность элементов одного типа.

В инструкции объявления массива указывается количество элементов массива. Элементы массива нумеруется с нуля. Доступ к элементу массива осуществляется путем указания индекса (номера ) элемента. В качестве индекса можно использовать выражение целого типа – константу или переменную. Индекс может меняться от 0 до n-1, где n-количество элементов массива.

 Доступ к элементам массива можно осуществить при помощи указателя. Для ввода, вывода и обработки массивов удобно использовать инструкции циклов (for,while) .

 Типичной ошибкой при использовании массивов является обращение к несуществующему элементу, т.е. выход индекса за допустимое значение.

 **Пример7.**

Написать программу, которая вводить с клавиатуры одномерный массив из 5 целых чисел, после чего выводит количество ненулевых элементов. Перед вводом каждого элемента должно выводиться подсказка с номером элемента. Рекомендуемый вид экрана во время выполнения программы приведен ниже.

Ввод массива целых чисел.

После ввода каждого числа нажмите <Enter>.

a[0] \_> **12**

a[1] \_> **0**

a[2] \_> **3**

a[3] \_> **-1**

a[4] \_> **0**

В массиве 3 ненулевых элемента.

//Подсчет ненулевых элементов массива

#include<iostream.h>

int main()

{

int a[5]; //размер массива

int n=0; //количество ненулевых элементов

int i; //индекс

cout<<”введите массив целых чисел.\n”;

cout<<”после ввода каждого числа нажмите<Enter>\n”;

for(i=0;i<5;i++)

{

cout<<”a[”<<i+i<<”]\_>”;

 cin>> a[i];

 if(a[i]!=0)n++;

}

 cout<<”в массиве”<<n<<” ненулевых элемента\n”;

return 0;}

**Пример 8**

Посчитать сумму элементов массива.

#include<iostream.h>

int main(){

 const int n=10;

 int i, sum;

 int mass[n]={3,4,5,4,3};

 for(I=0,sum=0;i<n; i++)sum+=mass[i];

 cout<<”Сумма элементов:”<<sum;

 return 0;

}

 **Практическая часть.**

ЗАДАЧИ

1. Написать программу, которая выводит минимальный элемент введенного с клавиатуры массива целых чисел. Ниже приведен рекомендуемый вид экрана во время работы программы (данные, введенные пользователем, выделены полужирным шрифтом).

Поиск минимального элемента массива .

Введите в одной строке элементы массива(5 целых чисел) и нажмите <Enter>.

\_>23 0 45-5 12

Минимальный элемент массива:-5

1. Написать программу, которая выводит минимальный элемент введенного с клавиатуры массива целых чисел. Для доступа к элементам массива используйте указатель.
2. Написать программу, которая вычисляет среднее арифметическое не нулевых элементов введенного с клавиатуры массива целых чисел. Ниже приведен рекомендуемый вид экрана во время работы программы (данные, введенные пользователем, выделены полужирным шрифтом).

Введите элементы массива (10 целых чисел) в одной строке и нажмите <Enter>.

\_>23 0 45- 5 12 0-2 30 0 64

Сумма элементов массива:184

Количество не нулевых элементов:7

Среднее арифметическое не нулевых элементов:23.86

1. Написать программу, которая проверяет, находится ли введенное с клавиатуры число в массиве. Массив должен вводиться во время работы программы.
2. Написать программу, которая проверяет, представляют ли элементы введенного с клавиатуры массива возрастающую последовательность.
3. Написать программу, которая вычисляет, сколько раз введенное с клавиатуры число встречается в массиве.
4. Написать программу, которая объединяет два упорядоченных по возрастанию массива в один, также упорядоченный массив. Рекомендуемый вид экрана во время работы программы (данные, введенные пользователем, выделены полужирным шрифтом).

Объединение два упорядоченных по возрастанию массивов .

Введите в одной строке элементы первого массива, (5 целых чисел)

\_>2 4 6 8 10

Массив – результат

1 2 3 4 5 6 7 8 9 10

Для завершения работы нажмите <Enter>.

1. Написать программу, которая, используя метод бинарного поиска, выполняет поиск в упорядоченном по возрастанию массиве.
2. Написать программу, которая, определяет количество учеников в классе, чей рост превышает средний. Рекомендуемый вид экрана во время работы программы приведен ниже. Введенные пользователем данные выделены полужирным шрифтом.

\*\*\*Анализ роста учеников \*\*\*

Введите рост (см) и нажмите <Enter>.

Для завершения введите ноль и нажмите <Enter>.

-175

-170

-168

-170

-**0**

Средний рост: 172.6см

У 2 человек превышает средний.

1. Написать программу, которая вводит по строкам с клавиатуры двумерной массив и вычисляет сумму его элементов по столбцам.
2. Написать программу, которая вводит по строкам с клавиатуры двумерный массив и вычисляет сумму его элементов по строкам.
3. Написать программу, которая обрабатывает результаты экзамен. Для каждой оценки программа должна вычислить процент от общего количество оценок. Рекомендуемый вид экрана во время работы программы приведен ниже. Данные, введенные пользователем, выделены полужирным шрифтом.

Обработка результатов экзамена.

Введите исходные данные:

Пятерок -**12**

Четверок-**10**

Троек-**7**

Двоек-**1**

Общее количество оценок-30

Результат экзамена

\_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_

 Пятерок 40%

Четверок 33.3%

Троек 23.3%

Двоек 3.3%

\_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_

Для завершения программы нажмите <Enter>.

 **Лабораторная работа №6**

 **Структуры**

**Цель работы:** освоение работы в С++ со структурами.

**Краткое теоритическое сведение**:

Структура –это совокупность переменных, объединенных одним именем. Структура может содержать элементы разных типов. Элементы структуры называются полями структуры и могут иметь любой тип.

Стандартный вид объявление структуры:

struct [имя]

{

тип 1 элемент 1:

тип 2 элемент 2:

…

тип n элемент n;

} [список описателей];

**Пример9:**

//описание структуры Студент

struct Student

{ string fio[20];

 int GroupNo;

 unsigned short uspevaemost [5];

};

 Для использования полученного типа Student объявим соответствующую переменную st:

Student st [n];

 Доступ к полям структуры выполняется с помощью операции выбора. (точка) при обращении к полю через имя структуры и -> при обращение через указатель, например:

for(i=0;i<n;i++)

 { cout<<”Familia “<<i+1<<”-studenta”; cin>>st[i].fio;

cout<<”\n”;

 cout<<”nomer gruppy”<<i+1<<”- studenta”;

cin>>st[i].GroupNo; cout<<”\n”;

 cout<<”Bally”; cin>>st[i].uspevaemost[0]

cin>>st[i]. uspevaemost[1]; Cin>>st[i]. uspevaemost[2];

cin>>st[i]. uspevaemost[3]; Cin>>st[i]. uspevaemost[4];

 cout<<”\n”;

}

**ЗАДАЧИ**

1. Описать структуру с именем STUDENT, содержащую следующее поля:

\*фамилия и инициалы:

\*номер группы:

\*успеваемость (массив из 5 элементов).

Написать программу, выполняющую следующее действие:

 \*ввод с клавиатурой данных в массив, состоящий из 10 структур типа STUDENT; записи должны быть упорядочены по возрастанию номера группы:

 \*вывод на дисплей фамилий и номеров групп для всех студентов, включенных в массив, если средний бал студента больше 4.0:

 \*если таких студентов нет, вывести соответствующее сообщение.

 2. Описать структуры с именем STUDENT, содержащую следующие поля:

 \*фамилия и инициалы

 \*номер группы:

 \*успеваемость(массив из 5 элементов).

Написать программу, выполняющую следующее действие:

 \*ввод с клавиатурой данных в массив, состоящий из 10 структур типа STUDENT: записи должны быть упорядочены по возрастанию среднего бала;

 \*вывод на дисплей фамилий и номеров групп для всех студентов, имеющих оценку 4 и 5;

 \*если таких студентов нет, вывести соответствующее сообщение.

3. Описать структуру с именем STUDENT, содержащую следующие поля:

 \*фамилия и инициалы:

 \*номер группы:

 \*успеваемость (массив из 5 элементов).

Написать программу, выполняющую следующее действие:

 \*ввод с клавиатурой данных в массив, состоящий из 10 структур типа STUDENT: записи должны быть упорядочены по алфавиту:

 \*вывод на дисплей фамилий и номеров групп для всех студентов, имеющих хотя бы одну оценку 2:

 \*если таких студентов нет, вывести соответствующее сообщение.

4. Описать структуру с именем AEROFLOT, содержащую следующие поля:

 \*название пункта назначение рейса;

 \*номер рейса;

 \*тип самолета.

Написать программу, выполняющую следующие действия:

 \*ввод с клавиатуры данных в массив, состоящей из 7 элементов типа AEROFLOT; записи должны быть упорядочены по возрастанию номера рейса;

 \*вывод на экран номеров рейсов и типов самолетов, в пункт назначения, название которого совпадало с названием, введенным с клавиатуры;

 \*если таких рейсов нет, выдать на дисплей соответствующее сообщение.

5. Описать структуру с именем AEROFLOT, содержащую следующие поля:

 \*название пункта назначение рейса;

 \*номер рейса;

 \*тип самолета.

Написать программу, выполняющую следующие действия:

 \*ввод с клавиатуры данных в массив, состоящей из 7 элементов типа AEROFLOT; записи должны быть размешены в алфавитном порядке по названиям пунктов назначения;

 \*вывод на экран пунктов назначения и номеров рейсов, обслуживаемых самолетом, тип которого введен с клавиатуры;

 \*если таких рейсов нет, выдать на дисплей соответствующее сообщение.

6. описать структуру с именем TRAIN, содержащую следующие поля:

 \*названия пункта назначения;

 \*номер поезда;

 \*время отправления.

Написать программу, выполняющие следующие действия:

 \*ввод с клавиатуры данных в массив, состоящей из 8 элементов типа TRAIN; записи должны быть размешены в алфавитном порядке по названиям пунктов назначения

 \*вывод на экран информация о поездах, отправляющих после введенного с клавиатуры времени;

 \*если таких поездов нет, выдать на дисплей соответствующее сообщение.

7. . Описать структуру с именем TRAIN, содержащую следующие поля:

 \*названия пункта назначения;

 \*номер поезда;

 \*время отправления.

Написать программу, выполняющие следующие действия:

 \*ввод с клавиатуры данных в массив, состоящей из 6 элементов типа TRAIN; записи должны быть упорядочены по времени отправления поезда;

 \*вывод на экран информация о поездах, направляющих в пункт, название которого ведено с клавиатуры;

 \*если таких поездов нет, выдать на дисплей соответствующее сообщение.

 **Лабораторная работа №7**

 **Функции в языке С++**

**Цель работы:** изучение программирование в С++ с использованием функции.

**Краткое теоретическое сведение:**

 C увеличением объема программы становится невозможным удерживать в памяти все детали. Естественным способом борьбы со сложностью любой задачи является ее разбиение на части. В С++ задача может быть разделена на более простые и обозримые с помощью функции, после чего программу можно рассматривать в более укрупненном виде – на уровне взаимодействия функций. Это важно, поскольку человек способен помнить ограниченное количество фактов. Использование функций является первым шагом к повышению степени абстракции программы и ведет к упрощению ее структуры.

 Разделение программы на функции позволяют избежать избыточности кода, поскольку функцию записывают один раз, а вызывать ее на выполнение можно многократно из разных точек программы.

**Функция -** это именованное последовательность описаний и операторов, выполняющая какое-либо законченное действие. Функция может принимать параметры и возвращать значение.

 Любая программа на С++ состоит из функции, одно из которых должна иметь имя main (с нее начинается выполнение программы). Функция начинает выполняться в момент вызова. Любая функция должна быть объявлена и определена.

 Как и для других величин, объявлений может быть несколько, а определения только одно. Объявления функции должно находиться в тексте раньше его вызова для того, чтобы компилятор мог осуществить проверку правильности вызова.

  **Стандартный вид функции:**

 тип\_ возвращаемого \_значения имя \_функции (список объявляемых параметров)

{

тело функции

}

  **Два метода передачи аргументов**

 Существуют два стандартных способа передачи аргументов функцию. Первый – **передача по значению**. При этом содержимое аргумента копируется в формальный параметр функции, все изменение произведенные над параметром не как не влияют назначение аргументов, используемых при вызове.

 **Передача по ссылке** является ко вторым способам передачи аргументов. В данном методе копируется адрес аргумента. В функции адрес используется доступа к настоящему аргументу. Соответственно, есть возможность влиять на содержимое переменных, используемых при вызове.

Пример:

int x;

cin>>x;

-функции ввода передается адрес переменной целочисленного типа х, по которому функция запишет значения, введенное при помощи клавиатуры(если бы функции передавалось само значения –она не смогла бы его изменить).

 **Передача массива**

 При передачи массива передается только его адрес (а не копия всего массива)

Пример: массив, объявленный как

int m[10];

Передается в функцию, возвращающую сумму его элементов следующим образом:

S=sum\_mas(m);

Как известно, m-указатель на первый элемент массива функция должна быть объявлена, как:

int sum\_mas(int \* m )

{

тело функции

}

Т.к. m-формальный параметр в функции, совпадение его имени с именем аргумента не обязательно, можно например:

int sum\_mas(int \*mas)

 **Где должно находиться описание функции?**

Существует два варианта: первый – функция полностью описывается перед первым ее вызовом, второй – перед первым вызовом функция только объявляется (помещается ее **прототип**), сама функция может находиться ниже в данном файле, в библиотечном модуле. Прототип функции имеет вид:

 Тип \_возвращаемого \_значения имя \_функции (список параметров);

Прототип функции может быть помещен в заголовочный файл.

Рассмотрим несколько примеров объявления, вызова и определения функций.

//Объявление функций:

int MyFunction (int Number, float Point);

char InputSymbol();

void EmptySample (int, char);

//Вызов функций:

Result = MyFunction (varbl, 3.14);

Symb = InputSymbol();

EmptySample (2, smb.l);

//Определение функций:

 int MyFunction (int Number, float Point);

{

 int my\_x;

 …..

 return my\_x; // my \_x - тип int

};

Char inputSymbol()

{

 char symbol;

 cin>>symbol;

 return symbol;

};

void EmptySample(int x, char ch)

{

};

Для того чтобы было более понятно место функции в программе, детально рассмотрим пример функции, возвращающей сумму двух целых чисел:

**Пример 10**

#include<iostream.h>

int sum(int a, int b); //объявление функции

int main() {

 int a=2, b=3, c, d;

 c=sum (a, b); //вызов функции

 cin>>d;

 cout<<sum(c, d); //вызов функции

 return 0;

}

int sum(int a, int b) { //определение функции

 return (a+b);

}

**ЗАДАЧА**

1. Написать функцию, которая вычисляет объем цилиндра. Параметрами функции должны быть радиус и высота цилиндра.
2. Написать функцию, которая возвращает максимальное из двух целых чисел, полученных в качестве аргумента.
3. Написать функцию, которая сравнивать два целых числа и возвращает результат сравнения в виде одного из знаков:>,<или=.
4. Написать функцию, которая вычисляет сопротивление и тип соединения (последовательное и параллельное). Функция должна проверять корректность параметров: если неверно указан тип соединения, то функция должна возвращать -1.
5. Написать функция “Faktorial” и программу, использующую эту функцию для вывода таблица факториалов.
6. Написать функцию “Dohol”, которая вычисляет доход по вкладку. Исходными данными для функции является: величина вклада, процентная ставка (годовых) и срок вклада (количество дней).
7. Написать функцию “glans”, которая возвращает1, если символ, полученный функцией в качестве аргумента, является гласной буквой русского алфавита, и ноль – в противном случае.
8. Написать функцию “sogl”, которая возвращает 1, если символ, полученный функцией в качестве аргумента , является согласной буквой русского алфавита, и 0-в противном случае.
9. Написать функцию, которая вычисляет объем и площадь поверхности параллелепипеда.

  **Лабораторная работа №8**

 **Классы**

**Цель работы:**

Освоение основ работы с классами С++

**Краткие теоретические сведения:**

 Классы является основой С++. Класс является типом данных, определяемых пользователем, и представляет собой модель реального объекта в виде данных и функций для работы с ними. Перед тем, как создать объект, необходимо определить его общую форму, используя ключевое слово **class**. Класс определяет новый тип данных, который соединяет в себе код и данные. По синтаксису класс аналогичен структуре, однако, в отличие от нее может включать функции, а не только данные.

 **Общий вид объявления класса:**

class <имя\_класса>

{

private:

<описание скрытых элементов> //закрытые функции-члены и данные-члены класса

public:

<описание доступных элементов> //открытые функции-члены и данные-члены класса

} <список объектов>;

 Функции и переменные, объявленные внутри класса, становятся членами этого класса. Переменные, объявленные внутри объявления класса, называется данными членами этого класса; функции, объявленные внутри объявления класса, называется функциями - членами класса. По умолчанию все функции и переменные, объявленные в классе, становятся закрытыми для этого класса. Это означает, что они доступны только других членов этого класса. Можно явно объявить их таковыми, используя ключевое слово **private**.

 Для объявления открытых членов класса используется ключевое слово **public**, за которыми следует двоеточие. Все функции и переменные, объявленные в классе после **public**, доступно и для других членов класса, и для другой части программы, в которой содержится весь класс.

 Вот пример объявления класса:

Class AnyClass

 {

 private:

 Int a;

 public:

 int get\_a();

 void set\_a(int num);

};

 Хотя функции **get\_a** и **set\_a** и объявлены в классе **AnyClass,**  они еще не определены. Для определения функции-члена нужно связать имя класса, частью которой является функция член, с именем функции. Это достигается путем написания имени функции, вслед за ними имеем класс, с двумя двоеточиями. Операцию, записываемую в виде двух двоеточий, называется операцией расширения области видимости. Для задания функции-члена используется следующая общая форма:

 <тип> <имя\_класса>: :<имя\_функции>

(<список\_параметров>)

{

//тело функции

}

 Ниже приведены примеры определения функций-членов get\_a() и set\_a():

Void AnyClass:: get\_a()

 {

 return a;

 }

 int AnyClass:: set\_a(int num)

 {

 a=num;

 }

Объявление класса **AnyClass** не приводит к созданию объектов типа **AnyClass.** Чтобы создать объект соответствующего класса, нужно просто использовать имя класса как спецификатор типа данных. Например :

 AnyClass ob1,ob2;

После того как объект создан, можно обращаться к открытым членом класса, используя операцию “точка”, так же как к полям структуры. Например:

 ob1. set\_a(10);

 ob2. set\_a(37);

 Эти операторы устанавливают значения переменной **а** в объектах **ob1** и **ob2**. Каждый объект содержит собственную копию всех данных, объявленных в классе. Это значит, что значение переменной **а**  в **ob1**отлично от значения этой переменной в ob2.

 Можно создать константный объект, значения данных которого изменять запрещается. К нему должны применяться только константы методы:

 \*объявляется с ключевым словом const после списка параметров;

 \* не может изменять значения данных классов;

 \*может вызываться для любых (не только константных) объектов.

 Рекомендуется описывать как константные те методы, которые предназначены для получения значений данных.

**Указатель this:** каждый объект содержит свой экземпляр данных класса. Функции класса находятся память в единственном экземпляре, поэтому необходимо обеспечить работу с функций с данными того объект, для которого они были вызваны. Это обеспечивается передачей в функцию скрытого параметра **this,** в котором хранится константный указатель на вызвавший функцию объект.

**Конструкторы и деструкторы:**

При создании объекта класса ему должны быть переданы значения его данных. Для хранения этих знаний должен быть выделен необходимый объем памяти туда должны быть записаны переданные значения, все это должен сделать **конструктор.** По окончании работы с объектом выделенная память должна быть освобождена , а все это делает **деструктор**.

**Использование конструкторов и деструкторов в С++** : чтобы реализовать данные требования - в С++ существует автоматически вызываемые при создании и разрушении объектов специальные функции - члены класса, выполняющие задачи по инициализации и уничтожению объектов. При отсутствии явного описания конструктора и деструктора, используется автоматически генерируемые конструктор и деструктор.

**Это нужно запомнить:**

\*конструктор и деструктор - это функции-члены класса (соответственно, они и описывается, как члены касса);

\*конструктор и деструктор -это открытые члены класса-соответственно они объявляется после ключевого слово public;

\*конструктор и деструктор не имеют возвращаемых значений, а потому и не нуждаются в описании типа возвращаемого значения (даже **void**-не нужен) кроме того, деструктор не имеет параметров;

\*имя конструктора всегда совпадает с именем класса;

\*имя деструктора образуется путем добавления значка -(тильда) перед именем класса;

\*у класса может быть несколько конструкторов, они различаются списком параметров.

**Друзья класса**

В некоторых ситуациях желательно, чтобы функция, не являющая членом класса, тем не менее, имела доступ к закрытым членом этого класса. С++ позволяет это сделать, если функция объявлена другом класса при помощи ключевого слово **friend.**

#include “iostream.h”

//\_ \_ \_ \_ \_ \_ \_ начало описания класса\_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_

class pUgolnic

{private: //закрытые функции и данные класса

float shirina, dlina;

public: //открытые функции и данные класса

pUgolnic (float shir=0; float dlin=0;) //коструктор

void setLength (float dlina); //объявление функций в классе

void SetWidth (float shirina);

float GetLength();

float GetWidth();

float GetPloshad();

};

//\_ \_ \_ \_ \_ \_ начало реализации класса\_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_

pUgolnic::pUgolnic(float shir, Float dlin) //коструктор

{this->shirina=shir;

this->dlina=dlin;

}

void pUgolnic:: SetLength(float dlina) //определение функций класса

{this->dlina=dlina; }

void pUgolnic:: SetLength(float shirina)

{this->shirina=shirina; }

float pUgolnic:: GetLength()

{ return dlina; }

float pUgolnic:: GetWidth()

{ return shirina; }

float pUgolnic:: GetPloshad()

{ return shirina\*dlina; }

//\_ \_ \_ \_ \_ \_ \_Конец реализации класса\_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_

void main()

{float dlina?shirina;

cout<<”Vvedite dlinu”;

cin>>dlina;

cout<<”\nVvedite shirinu”;

cin>> shirina;

pUgolnic ob1; //создание объекта класса

ob1.SetLength (dlina); //обращение к открытым членам класса

ob1. SetWidth (shirina);

cout<<”\nPolshad=”;

cout<<ob1.GetPloshad();

}

**ЗАДАЧА**

Примечание: во всех классах должно быть хотя бы одно поле, память под которое выделяется динамически.

1. Описать класс, реализующий стек. Написать программу, использующую этот класс для моделирования Т-образного сортировочного узла на железной дороге. Программа должна разделять на два направления состав, состоящий из вагонов двух типов (на каждое направления формируется состав из вагонов одного типа). Предусмотреть возможность формирования состава из файла и с клавиатуры.

2. Описать класс, реализующий бинарное дерево, обладающее возможностью добавления новых элементов, удаления существующих, поиска элемента по ключу, а также последовательного доступа ко всем элементам.

Написать программу, использующую этот класс для представления англо-русского словаря. Программа должна содержать меню, позволяющее осуществить проверку всех методов класса. Предусмотреть возможность формирования словаря из файла и с клавиатуры.

3. Построить систему классов для описания плоских геометрических фигур: круг, квадрат, прямоугольник. Предусмотреть методы для создания объектов, перемещения на плоскости, изменения размеров и вращения на заданный угол. Написать программу, демонстрирующую работы с этими классами.

Написать программу, демонстрирующую работу с этими классами. Программа должна содержать меню, позволяющее осуществить проверку всех методов классов.

4. Построить описание класса, содержащего информацию о почтовом адресе организации. Предусмотреть возможность раздельного изменения составных частей адрес, создания и уничтожения объектов этого класса. Длину строк задавать динамически.

Написать программу, демонстрирующую работу с этими классами. Программа должна содержать меню, позволяющее осуществить проверку всех методов классов.

5. Составить описание класса для представления комплексных чисел. Обеспечить выполнение операций сложения, вычитания и умножения комплексных чисел.

Написать программу, демонстрирующую работу с этими классами. Программа должна содержать меню, позволяющее осуществить проверку всех методов классов.

6. Составить описание класса для объектов-векторов, задаваемых координатами концов в трехмерном пространстве. Обеспечить операции сложения и вычитания векторов с получением нового вектора (суммы и разность), вычисления скалярного произведения двух векторов, длина вектора, косинуса угла между векторами.

Написать программу, демонстрирующую работу с этими классами. Программа должна содержать меню, позволяющее осуществить проверку всех методов классов.

7. Составить описание класса прямоугольника со сторонами, параллельными осям координат. Предусмотреть возможность перемещения прямоугольников

На плоскости, изменения размеров, построение наименьшего прямоугольника, содержащего два заданных прямоугольника, и прямоугольника, являющегося общей частью(пересечением) двух прямоугольников.

Написать программу, демонстрирующую работу с этими классами. Программа должна содержать меню, позволяющее осуществить проверку всех методов классов.

8. Составить описания класса для определения одномерных массивов целых чисел (векторов). Предусмотреть возможность обращения к отдельному элементы массива с контролем выхода за пределы массива, возможность задания произвольных границ индексов при создании объекта и выполнения операций поэлементного сложения, и вычитания массивов с одинаковыми границами индексов, умножения и удаления всех элементов массива на скаляр, вывода на экран элемента массива по заданному индексу и всего массива.

Написать программу, демонстрирующую работу с этими классами. Программа должна содержать меню, позволяющее осуществить проверку всех методов классов.

9. Составить описание класса для определения одномерных массивов строк фиксированной длины. Предусмотреть возможность обращения к отдельным строкам массива по индексам, контроль выхода за пределы массива, выполнения операции поэлементного сцепления двух массивов с образованием нового массива, слияние двух массивов с исключением повторяющих элементов, вывод на экран элемента массива по заданному индексу и всего массива.

Написать программу, демонстрирующую работу с этими классами. Программа должна содержать меню, позволяющее осуществить проверку всех методов классов.

1. Составить описание класса многочленов от одной переменной , задаваемых степенью многочлена и массива коэффициентов. Предусмотреть методы для вычисления значения многочлена для заданного аргумента, операции сложения, вычитания и умножения многочленов с получением нового объекта-многочлена, вывод на экран описания многочлена.

Написать программу, демонстрирующую работу с этим классом. Программа должна содержать меню, позволяющее осуществить проверку всех методов класса.

1. Составить описания класса одномерных массивов строк, каждая строка задается длиной и указателем на выделенную для него память. Предусмотреть возможность обращение к отдельным строкам массива по индексам, контроль выхода за пределы массивов, выполнения операции поэлементного сцепления двух массивов с образованием нового массива, слияния двух массивов с исключением повторяющих элементов, вывод на экран элемента массива и всего массива.

Написать программу, демонстрирующую работу с этим классом. Программа должна содержать меню, позволяющее осуществить проверку всех методов класса.

1. Составить описания класса, обеспечивающего представления матрицы произвольного размера с возможностью изменения числа строк и столбцов, вывод на экран подматрицы любого размера и всей матрицы.

Написать программу, демонстрирующую работу с этим классом. Программа должна содержать меню, позволяющее осуществить проверку всех методов класса.