**Лекция 2 – функции в C++**

Функции – блоки кода, выполняющие определенные операции. Определяется и выполняется отдельно от main.

**void func() {**

**cout << “Я функция, меня вызвали” << endl;**

**}** – пример функции.

**возвращаяемое\_значение имя\_функции(аргументы) {**

**тело\_функции (операторы);**

**return <значение>;**

**}** – функция в общем виде

ДЗ: разобраться с возможностью доступа к памяти в конкретной задаче

ДЗ: есть понятие “прототип функции”. Что это такое? С примером в коде в тетради

Прототип:

**int func(int a); //прототип функции**

**int main(){**

**func(2);**

**}**

**int func(int a) {**

**//тело**

**}**

Без прототипа выдастся ошибка, так как функция объявляется после main. Сама функция может быть вообще в другом файле.

В пототип входят:

- Возвращаемый тип данных

- Имя функции

- Аргументы

**Правила работы с функциями**

1. Тип выходных данных **не может быть массивом или функцией**;
2. Тип функции зависит от **типа выходных данных**;
3. Имя функции **не может быть ключевым словом** (namespace и т.д.);
4. **Нельзя** определять функцию внутри другой функции, но вызывать можно.
5. **Нет return = ошибка**, но не в main;

**Тип void в функциях**

**Void** показывает, что функция не возвращает результат. В теле функции нет return.

Пример неправильной функции с void:

**void sum(int a, int b){**

**return a + b; //void функция ничего не возвращает**

**}**

**int main(){**

**int c = sum(23, 4); //нельзя присвоить “ничего” к int**

**count << c << endl;**

**}**

**Функция main**

1. Функция main обозначает точку начала выполнения программы;
2. Стандарт – int main;
3. Другие использования – void main;
4. Лучше не использовать int/void main(параметры).

**Параметры функций**

1. Функции бывают **вызываемыми и вызывающими**; (см. пример выше);
2. Обмен параметрами между ними – **главный механизм работы функции**.
3. Виды параметров:
4. **Формальные параметры** – параметры, которые перечислены в описании функции (то есть в первом её объявлении);
5. **Фактические параметры (аргументы)** – находятся в операторе вызова функции. Именно с ними будет работать функция.
6. Способы передачи параметров в функцию:
7. **По значению** – функция работает с **копиями** аргументов, которые не могут изменить их исходные значения;
8. **По адресу** – работа идет с копиями адресов аргументов. Можно изменить исходные значения по этим адресам.

**Использование в функциях классов памяти**

1. Класс auto:
2. Используется **только** для локальных переменных (используется по умолчанию);
3. Память под auto заносится автоматически;
4. Используется как для входных, так и для выходных (auto fucn(auto a){});
5. Автоматически тип данных будет int, если присутствует тип памяти int.
6. Register:
7. Записывает переменные в регист процессора, если возможно, иначе работает аналогично auto;
8. Также применяется только для локальных переменных.
9. А
10. Static:
11. Используется для объявления переменных внутри блока.
12. Extern:
13. Используется для глобальных переменных во всей программе;
14. Доступ осуществляется возможен во всей программе.

**Передача в функцию массива**

1. В функцию передается указатель на первый элемент массива (то есть адрес всего массива);
2. В функцию передается кол-во элементов в массиве.

Дз: лабораторная работа 7

**Функция с переменным кол-во элементов**

1. Используется, когда не известно количество переменных;
2. Должен быть хотя бы один явный параметр;
3. Как определить окончание кол-во входных параметров?
4. Задать в явном виде;
5. Задать в виде признака окончания;
6. Автоматизация невозможна.

**Перегрузка функции**

1. Используется, чтобы функция с одним именем могла работать **с разными типами данных**;
2. Область видимости – та часть кода, **которая видна**;
3. Время жизни – сколько времени переменная живет в этой ячейки памятки;
4. Свойства:
5. Должны находиться в одной области видимости;
6. Тип данных входных и выходных параметров может отличаться.

**Шаблоны функции**

1. Шаблон автоматизирует создание перегруженных функций, тогда перегруженная функция определяется 1 раз для разных типов данных;
2. Пример:

**template <typename параметр>**

**параметрт имя\_функции (параметр входные\_данные){**

**}**