ЗАНЯТИЕ №4

Функции

- Функция – именованная последовательность инструкций.

```
type func_name (arg1. Type argument1,
…., argn, argumentn)
{
 //do something
 type result;
 //do something
 return result;
}
```

ФУНКЦИИ

Применение:

- Вычисления, которые логически отделены от других
- Отделение вычислений делает программу яснее
- Упрощение отладки программы
- Функцию можно использовать более чем в одном месте программы

ФУНКЦИИ

- Типы возвращаемого значения:
- void – пустота

Аргументы функции изменяются в функциях типа void

```
void foo (arg1, arg2….)
{
//do something
}
```

ФУНКЦИИ

Не пустые типы данных

```
int foo(int a)
{
    a=a+1;
    return a;
}
```

Аргументы функции не изменяют свое значение в основном блоке кода

ФУНКЦИИ

```
void foo1(int a)
{   a++;}

int foo2(int a)
{  a=a+1;
return a;}


int main()  {
```

ПЕРЕДАЧА ССЫЛОК И УКАЗАТЕЛЕЙ

```
void foo1(int *a)
{*a++;}
```

```
int foo2(int &a)
{   a++;
return a;}
```

ПЕРЕГРУЗКА ФУНКЦИЙ

- Перегрузка позволяет объявить одинаковое название для функций с разными аргументами.

```
int foo(int a)
{
    a=a+1;
    return a;
}
```

```
double foo (double a)
{
  return a+1.;
}
```

ПРОТОТИП ФУНКЦИИ

- **Прототип функции** — это функция, в которой отсутствует блок кода (тело функции).

```
double foo (double a);
int main()
{
   double num=foo(1.);
}

double foo (double a)
{
   return a+1.;
}
```