Функции

Функция — это самостоятельная единица программы, созданная для решения конкретнной задачи.

Например, мы уже использовали несколько функций из стандартной библиотеки (`printf()`, `rand()`, `abs()` и др.)

Общий вид функции выглядит следующим образом:

```
тип_возвращаемого_значения имя_функции(список_параметров) {
    тело функции;
    return возвращаемое_значение;
}
```

Каждая функция имеет имя, принимаемые параметры и тип возвращаемого значения.

Для облегчения понимания кода имя для функции следует выбирать в соответствии с ее назначением. Например, если ваша функция что-то суммирует, то лучше выбрать для нее название `summa()`.

Параметры функции — это необязательный атрибут. В качестве параметров в функцию можно передавать значения переменных, которые требуются для работы функции. Любая функция может, как совсем не принимать параметров, так и принимать их практически неограниченное множество.

Тип возвращаемого значения соответствует типу выражения или переменной, которая возвращается в качестве результата работы функции. Если при работе функция не возвращает никакого значения, то за тип возвращаемого значения принимается тип `void`. В этом случае у функции отсутствует оператор `return`.

```
void имя_функции(список_параметров) {
    тело функции;
}
```

Объявление функции

Объявление функции сообщает компилятору, что далее в программе будет описана и вызвана объявляемая функция. Функция, как и переменная, должна быть объявлена до своего непосредственного использования.

При объявлении функции следует указать имя функции, тип параметров, которые принимает функции и тип значения, которое функция возвращает.

Примеры объявления функций:

```
int summa(int, double);
double calculations(double, double, int);
void print_massiv();
```

Описание функции

Под описанием функции понимается описание действий функции при ее вызове. Описания функций располагаются последовательно друг за другом после функции main(). Это позволяет лучше ориентироваться в больших программах.

При описании функции параметры функции должны включать не только типы, но и имена переменных. Тело функции описывается в фигурных скобках.

```
тип_возвращаемого_значения имя_функции(список_параметров){
    --------------
    тело функции
    --------------
    return возвращаемое_значение;
}
```

Нижеприведенные функции позволяют находить среднее арифметическое двух целых чисел и находить наибольшее из двух целых чисел.

```
double average (int a, int b) {
    return (double) (a + b) / 2;
}

int max(int a, int b) {
    int max;
    if (a > b) {
        max = a;
    }
    else {
        max = b;
    }
    return max;
}
```

Последний пример можно видоизменить, убрав переменную `max`. В этом случае следует использовать несколько операторов **`return`**.

```
int max(int a, int b) {
    if (a > b) {
        return a;
    }
    else {
        return b;
    }
}
```

Если функция имеет тип **void** и не возвращает значения, то оператор **return** отсутствует.

```
void print_mass() {
     int i;
     for (i = 0; i < N; i++) {
         printf("%5d", mass[i]);
     }
}
```

Вызов функции

Функция может быть вызвана из любой другой функции, описанной в программе, даже из самой себя. Если функция в процессе работы вызывает сама себя, то это называется рекурсией. Важной особенностью вызова функции является то, что параметры, передаваемые в функцию, должны быть указаны в правильном порядке в соответствии с их типом так же, как они указаны при объявлении функции. При этом тип параметров указывать не нужно.

Если функция возвращает значение, то результат ее работы должен быть присвоен какой-либо переменной или должен быть использован в выражении.

Примеры вызова функций:

```
int maximum = max(a, b);
double average = (summ1(x, y) + summ2(a, b) )/ 2;
printf("%d + %d = %d", a ,b, summa(a, b));
```

Если функция не возвращает значения, то ее вызов осуществляется без последующего использования результата. При этом вызов такой функции происходит простой записью имени функции и передачей туда параметров.

```
print_v();
calculations(num1, num2);
```

Связь между объявлением, описанием и вызовом функции

```
double count (double, int); //объявление функции

int main(void) {
    double a = 0;
    int b = 0;
    scanf("%lf %d", &a, &b);
    double sum = count(a, b); //вызов функции
    printf("сумма %.2lf и %d равна %.2lf", a, b, sum);
    return EXIT_SUCCESS;
}

double count (double x, int y) { //описание функции
    return x + y;
}
```

При передаче в функцию происходит копирование значений переменных `a` и `b` в переменные `x` и `y`. Внутри функции `count()` переменные `a` и `b` не существуют, но вместо них существуют переменные `x` и `y` со значениями переменных `a` и `b`.

Примеры задач с использованием функций

Пример. Задать с клавиатуры целое число. Вывести на экран объем куба со стороной, равной этому числу. Расчет произвести с помощью функции.

```
int cube_volume(int);

int main(void) {
    int side = 0;
    scanf("%d", &side);
    int volume = cube_volume(side);
    printf("объем куба равен %d", volume);
    return EXIT_SUCCESS;
}
```

```
int cube_volume(int a) {
    return pow(a, 3);
}
```

Пример. Задав значения переменных с клавиатуры, вычислить результат с помощью функции.

```
double calculations(double, double, double);

int main(void) {
    double a = 0, x = 0, f = 0, t = 0;
    printf("Введите значения дробных переменных a, x, f");
    scanf("%lf %lf %lf", &a, &x, &f);
    if (((f - 16) > 0) && (cos(a + x) != 0)) {
        t = calculations(a, x, f);
        printf("Результат вычислений равен %.4lf", t);
    }
    else {
        printf("Неверные входные данные");
    }
    return EXIT_SUCCESS;
}

double calculations(double x, double y, double z) {
    return x + log(z - 16) / cos(x + y);
}
```

Пример. Задав три целых числа с клавиатуры, с помощью функций определить, какое из них является средним по значению.

```
int compare(int, int, int);

int main(void) {
    int first = 0, second = 0, third = 0;
    printf("Введите значения трех переменных\n");
    scanf("%d %d %d", &first, &second, &third);
    int average = compare(first, second, third);
    printf("Средняя по значению переменная равна %d",
average);
    return EXIT_SUCCESS;
}

int compare(int a, int b, int c) {
    int avr;
    if (((b > a) && (a > c)) || ((c > a) && (a > b))) {
        avr = a;
    }
    if (((a > b) && (b > c)) || ((c > b) && (b > a))) {
        avr = b;
    }
    if (((a > c) && (c > b)) || ((b > c) && (c > a))) {
        avr = c;
    }
    return avr;
}
```