Подпрограмма, после окончания работы, обычно должна вернуть в главную программу результат работы.

Возврат результатов работы можно выполнить двумя способами:

1. Оператором **return.**
2. Возврат результатов через **параметры**.

# Оператор return

С помощью оператора **return** можно вернуть расчетное значение в главную программу.

Пример 1,

**#include <iostream>**

**using namespace std;**

**int fun(int b) {**

**return 10\*(b\*b+2);**

**}**

**int main()**

**{**

**int a;**

**cout << "a="; cin >> a;**

**int ko = fun(a);**

**cout << "ko= " << ko;**

**}**

Пример 2,

**int fun(int b) {**

**int y=10\*(b\*b+2);**

**return y;**

**}**

Пример 3,

**#include <iostream>**

**using namespace std;**

**float fun(int b) {**

**float y=(float)10/(b\*b+2);**

**return y;**

**}**

**int main()**

**{**

**int a;**

**cout << "a="; cin >> a;**

**float ko = fun(a);**

**cout << "ko= " << ko;**

**}**

Пример 3,

**#include <iostream>**

**using namespace std;**

**float fun(int a,int b) {**

**float y=(float)10/(a\*b+2);**

**return y;**

**}**

**int main()**

**{**

**int a,b;**

**cout << "a="; cin >> a;**

**cout << "b="; cin >> b;**

**float ko = fun(a,b);**

**cout << "ko= " << ko;**

**}**

Часто подпрограмма возвращает через **return** не расчетное значение, а признак ошибки. Например, ko=0 значит нормальное завершение, ko=1 – деление на ноль, ko=2 – слишком большое число и пр.

Пример 4,

**#include <iostream>**

**using namespace std;**

**float fun(int a,int b) {**

**float ko;**

**if (a>b) float ko=(float)10/(a-b);**

**else**

**if(a==b) ko=1;**

**else ko=2;**

**return ko;**

**}**

**int main()**

**{**

**int a,b;**

**cout << "a="; cin >> a;**

**cout << "b="; cin >> b;**

**float ko = fun(a,b);**

**cout << "ko= " << ko;**

**}**

**Пример 5,**

**#include <iostream>**

**using namespace std;**

**float fun(int a,int b) {**

**float ko;**

**if (a>b) ko=(float)(10/(a-b));**

**else**

**if(a==b) ko=1;**

**else ko=2;**

**return ko;**

**}**

**int main()**

**{**

**int mas[2];**

**cout << "mas[0]="; cin >> mas[0];**

**cout << "mas[1]="; cin >> mas[1];**

**float ko = fun(mas[0],mas[1]);**

**cout << "ko= " << ko;**

**}**

**Параметры** подпрограммы предназначены для передачи входных данных в подпрограмму, а также, в некоторых случаях приема результатов работы подпрограммы.

Параметры, которые указываются при вызове подпрограммы, называются **фактическими**. Так в примерах 1-5, при вызове:

**int ko = fun(a); //а – фактический параметр**

**float ko = fun(a,b); //а и b – фактические параметры**

**float ko = fun(mas[0],mas[1]); //mas[0] и mas[1] – также факт. Парам**

Во всех примерах (1-5) фактические параметры передавались **по значению**. Существует возможность передачи фактических параметров по ссылке (адресу). Это более экономично и эффективно.

Пример 6,

**#include <iostream>**

**using namespace std;**

**float fun(int a[]) {**

**float ko;**

**if (a[0]>a[1]) ko=(float)(10/(a[0]-a[1]));**

**else**

**if(a[0]==a[1]) ko=1;**

**else ko=2;**

**return ko;**

**}**

**int main()**

**{**

**int mas[2];**

**cout << "mas[0]="; cin >> mas[0];**

**cout << "mas[1]="; cin >> mas[1];**

**float ko = fun(&mas[0]);**

**cout << "ko= " << ko;**

**}**

В данном примере **&mas[0] –** это адрес начала массива.

Пример 7,

**#include <iostream>**

**using namespace std;**

**float fun(int a){**

**a=a\*a+10;**

**cout << "a=" << a << endl;**

**return 0;**

**}**

**int main()**

**{**

**int a=3;**

**float ko = fun(a);**

**cout << "a= " << a;**

**}**

Из этого примера ясно, что в подпрограмме fun формальный параметр **a** – внутренний и после своей работы не изменяет значения **a** в главной программе.

Чтобы переменная **a** стала «общей» требуется передать ее по **ссылке**

Пример 8,

**#include <iostream>**

**using namespace std;**

**float fun(int& a){**

**a=a\*a+10;**

**cout << "a=" << a << endl;**

**return 0;**

**}**

**int main()**

**{**

**int a=3;**

**float ko = fun(a);**

**cout << "a= " << a;**

**}**

# Возврат результатов через параметры.

Параметры, которые указаны в описании подпрограммы называются **формальными.**

Главная особенность формальных параметров заключается в следующем:

1. Количество формальных параметров должно соответствовать количеству фактических.
2. Порядок следования формальных параметров должен соответствовать порядку следования фактических.

Пример 9,

**#include <iostream>**

**using namespace std;**

**bool fun(int a,int& y){**

**bool ko;**

**if (a>0) {y=a\*a+10; ko=true;}**

**else ko=false;**

**return ko;**

**}**

**int main()**

**{**

**int a=-3;**

**int y=0;**

**bool ko = fun(a,y);**

**if (ko) cout << "y= " << y;**

**else cout << "a<=0";**

**}**

Конец лекции