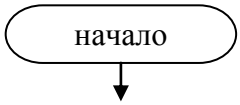
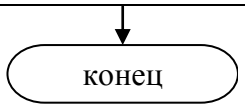
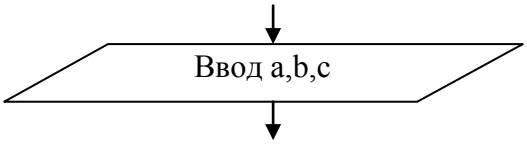
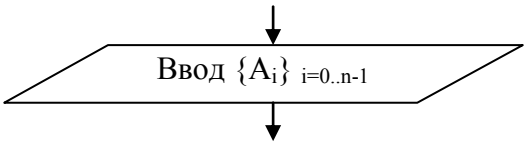
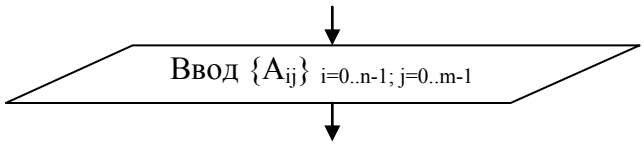
Блок-схема	Программный код

	int **main**(){ //начало главной программы либо
	int **main**(int argc, char* argv[]){//с параметрами
	float **f**(float x){ //функция f(x), на входе x

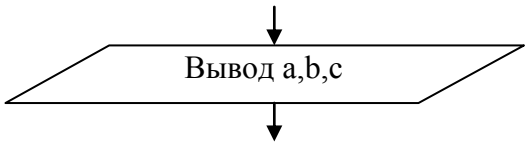
	**return** y; // возвращение результата *y* или 0
	// y **main** результат – код ошибки либо 0(нет ошибки)
	} //конец тела функции – фигурная скобка

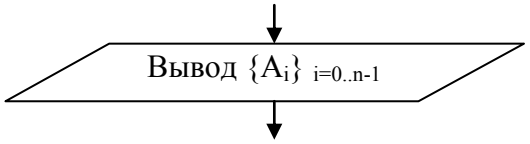
Ввод и вывод значений переменных	
	Ввод в одной строке: printf("Введите через пробел три **<u>целых</u>** числа: "); scanf("%**d**%d%d", &a, &b, &c); // %**d** – десятичное целое, // %**f** и %**lf** (**L**ong **F**loat)– вещественные // %**c** – символ, %**s** - строка(слово) Ввод в отдельных строках: printf("Введите a="); scanf("%d", &a); printf("Введите b="); scanf("%d", &b); printf("Введите c="); scanf("%d", &c);

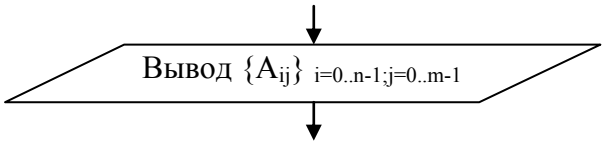
	Ввод в через пробел или Tab, Enter:
	
	printf("Массив A: "); // из цел.чисел: int A[20], n=7;
	for(int i=0; i<n; i++) scanf("%**d**", &A[i]);

	Ввод с произвольным расположением, в том числе в N строках по M элементов:
	
	// из цел.чисел: int A[10][10], n=2,m=3;
	printf("Введите матрицу A");
	for(int i=0; i<n; i++)
	  for(int j=0; j<n; j++)
	  scanf("%**d**", &A[i][j]);
	
	Ввод из N строк по M элементов:
	
	// из цел.чисел: int A[10][10], n=2,m=3;
	printf("'Введите матрицу A построчно ");
	for(int i=0; i<n; i++){ // для каждой строки i
	  for(int j=0; j<m; j++) // из каждого столбца ввести:
	  scanf("%**d**", &A[i][j]);

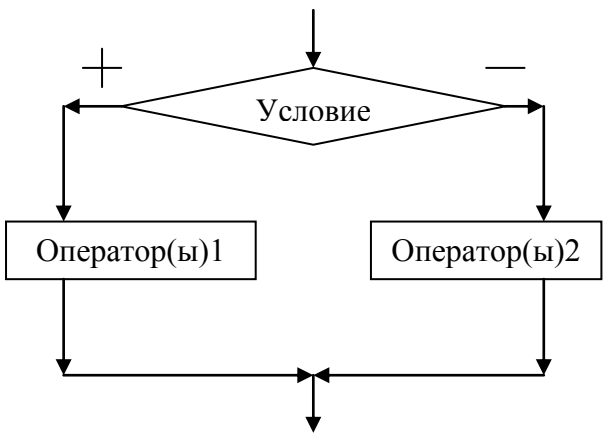
Блок-схема	Программный код

	Вывод в одной строке:
	
	// int a=7, b=9, c=-4;
	printf("Значения a=%d, b=%d, c=%d: ", a,b,c);
	// %**d** – десятичное целое, либо %i %u %o %x
	// %**f** и %**lf** (**L**ong **F**loat)– вещественные, либо %g %e
	// %**c** – символ, %**s** - строка(слово)
	
	Вывод в отдельных строках:
	
	printf("Значения: ");
	printf("a=%d", a);
	printf("b=%d", b);
	printf("c=%d", c);

	Вывод в одной строке через пробел:
	
	printf("Массив A: "); //из вещ.чисел: float A[20]; int n=7;
	for(int i=0; i<n; i++) printf ("%**f** ", A[i]);
	printf("\n");
	
	Вывод в столбец шириной 7 символов:
	
	//например, из вещ.чисел: float A[20]; int n=7;
	printf("Массив A: \n");
	for(int i=0; i<n; i++) printf **("%7.1f \n",** A[i]);

	Вывод на N строках по M элементов:
	
	printf("Матрица A: \n");
	for(int i=0; i<n; i++) {
	for(int j=0; j<m; j++) printf ("%**f** ", A[i][j]);
	printf("\n");
	}

Ветвление. Оператор условного перехода



if (*Условие*)
Оператор;
else
Оператор;

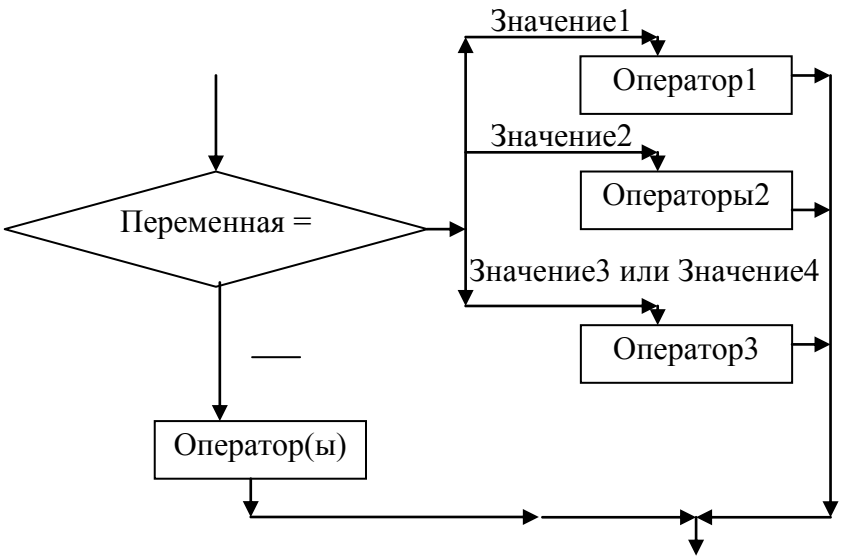
Для нескольких операторов:
if (*Условие*) {
Оператор1_1;
. . . . . .
Оператор1_K;
}
else {
Оператор2_1;
. . . . . .
Оператор2_R;
}

При отсутствии операторов на ветви else*:*
if (*Условие*){
Оператор1_1;
. . . . . .
Оператор1_K;
}

При отсутствии операторов на ветви then*:*
if (*Условие)* ;//пустой оператор;
else {
Оператор2_1;
. . . . . .
Оператор2_R;
}

Блок-схема	Программный код

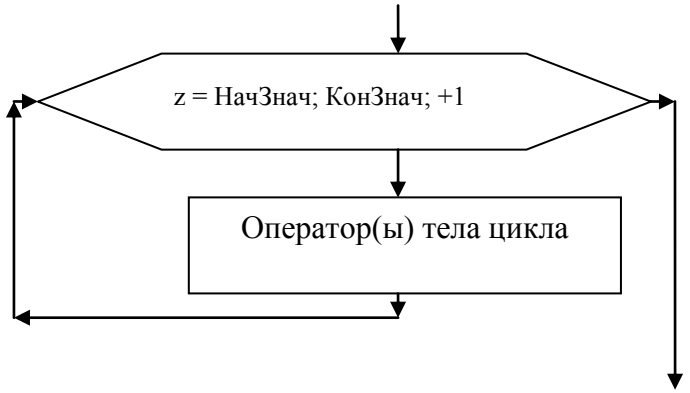
Оператор множественного выбора:




```
switch (Переменная)
    case Значение1: Оператор1; break;
    case Значение2:
                    Оператор2_1;
                    . . . . . .
                    Оператор2_K;
                  break;
    case Значение3:
    case Значение4: Оператор3; break;
    default:
                Оператор_1;
                . . . . . .
                Оператор_R;
}
```

Без *break* продолжается выполнение операторов для следующего значения. Значение определяет точку входа.

Параметрический цикл ДЛЯ (с параметром цикла z)

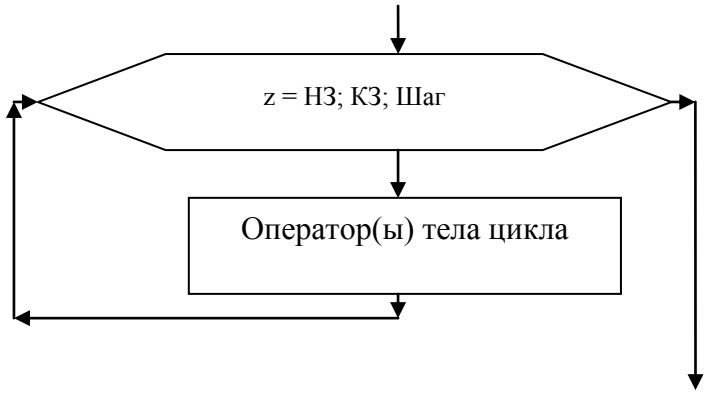


Если *НачЗнач ≤ КонЗнач*

```
for (z = НачЗнач; z< КонЗнач+1; z++)
    Оператор;
```

Для нескольких операторов:

```
for (z=НачЗнач; z< КонЗнач+1; z++)  {
    Оператор1;
    . . . . . .
    ОператорK;
}
```



Если *НЗ ≤ КЗ*

```
for (z=НЗ; z< КЗ+1; z+=Шаг)
    Оператор;
```

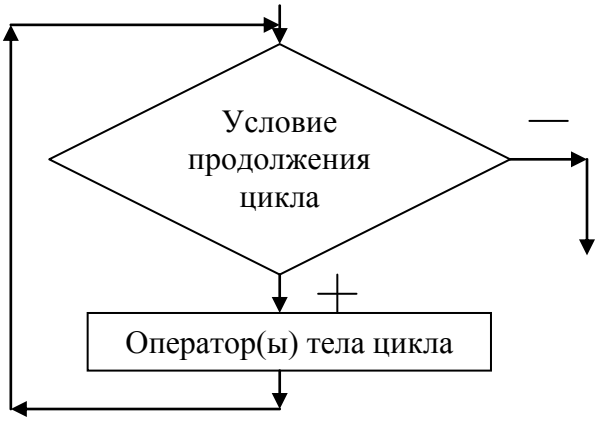
Если *НЗ ≥ КЗ*

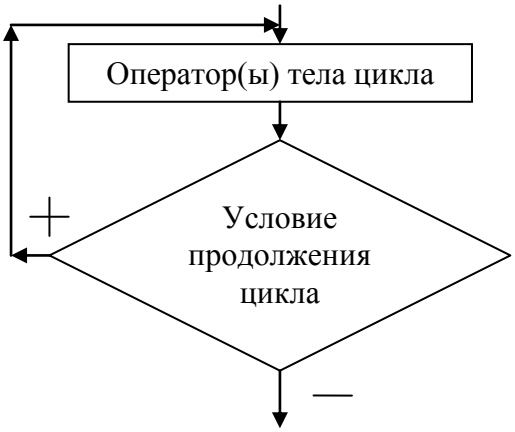
```
for (z=НЗ; z> КЗ-1; z=z-Шаг)
    Оператор;
```

Для нескольких операторов и *НЗ ≤ КЗ*:

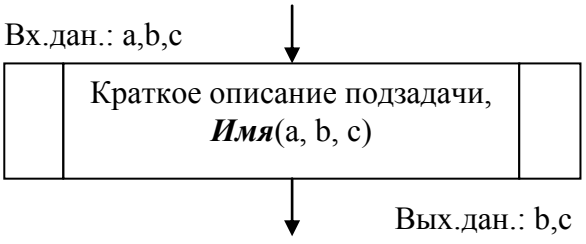
```
for (z=НачЗнач; z< КонЗнач+1; z+=Шаг)  {
    Оператор1;
    . . . . . .
    ОператорK;
}
```

Блок-схема	Программный код
Итерационные циклы ПОКА с пред- и пост- условиями	

	while (*Условие_продолжения*)
	  Оператор;
	
	Для нескольких операторов:
	
	while (*Условие_продолжения*) {
	  Оператор_1;
	  . . . . . .
	  Оператор_K;
	}

	do
	  Оператор;
	while (*Условие_продолжения*);
	
	Для нескольких операторов:
	
	do {
	  Оператор_1;
	  . . . . . .
	  Оператор_K;
	} while (*Условие_продолжения*);

Подпрограммы: процедуры и функции	

	**void *Имя***(*TunA **a**, TunB ***b***, TunC **&***c*){ // по ссылке **C++**
	// Локальные описания переменных, констант, типов
	  Оператор(ы); //изменить значение по адресу ***b***=
	  return;
	}
	*a – лок.**копия**, b – по **адресу**(явно), c – по **ссылке** (C++)*
	
	Вызов процедуры (функция без результата):
	Имя(a, **&**b, c); // если *TunA a; TunB b*; *TunC c;*

	TunB Имя(*TunA a, TunB **b**, TunC **&***c*){ //по ссылке **C++**
	// Локальные описания переменных, констант, типов
	  Оператор(ы);
	  return **b**;
	}
	b: в параметрах – копия входного значения (если нужна), в результате – возвращается изменённое
	
	Вызов функции с возвращаемым значением:
	b*=*Имя(a,***b***,c); // если *TunA a; TunB b*; *TunC c;*