

Программа на языке Си состоит из набора директив препроцессора, определений функций и глобальных объектов. Директивы препроцессора управляют преобразованием текста до его компиляции. Глобальные объекты определяют используемые данные или состояние программы. А функции определяют поведение или действия программы. Точкой входа для выполнения программы является функция с фиксированным идентификатором main. Язык Си является блочно-структурированным. Каждый блок заключается в фигурные скобки. Каждое действие в языке Си заканчивается символом «точка с запятой» — ;. В качестве действия может выступать вызов функции или осуществление некоторых операций.

Константа — это величина, которая при выполнении программы остаётся неизменной. Переменная — это ячейка памяти для временного хранения данных. int a - переменная, `c` - символьная константа. Константу можно задать с помощью квалификатора const, макроса #define и перечисления enum.

Компиляция – это процесс превращения исходного кода в машинный код. Этапы компиляции:

1) Препроцессинг

Препроцессор — это макро процессор, который преобразовывает вашу программу для дальнейшего компилирования. На данной стадии происходит происходит работа с препроцессорными директивами. Например, препроцессор добавляет хэдеры в код (#include), убирает комментирования, заменяет макросы (#define) их значениями, выбирает нужные куски кода в соответствии с условиями #if, #ifdef и #ifndef.

2) Компиляция

На данном шаге gcc выполняет свою главную задачу — компилирует, то есть преобразует полученный на прошлом шаге код без директив в ассемблерный код. Это промежуточный шаг между высокоуровневым языком и машинным (бинарным) кодом. Ассемблерный код — это доступное для понимания человеком представление машинного кода.

3) Ассемблирование

Ассемблер преобразовывает ассемблерный код в машинный код, сохраняя его в объектном файле.

Объектный файл — это созданный ассемблером промежуточный файл, хранящий кусок машинного кода.

4) Компоновка

Компоновщик (линкер) связывает все объектные файлы и статические библиотеки в единый исполняемый файл, который мы и сможем запустить в дальнейшем. Для того, чтобы понять как происходит связка, следует рассказать о таблице символов.

Таблица символов — это структура данных, создаваемая самим компилятором и хранящаяся в самих объектных файлах. Таблица символов хранит имена переменных, функций, классов, объектов и т.д., где каждому идентификатору (символу) соотносится его тип, область видимости. Также таблица символов хранит адреса ссылок на данные и процедуры в других объектных файлах.



Язык Си предоставляет множество базовых типов. Большинство из них формируется с помощью одного из четырёх арифметических спецификаторов типа, (char, int, float и double), и опциональных спецификаторов (signed, unsigned, short и long).

В большинстве систем char - 1 байт, int - 4 байта, float - 8 байт, double - 16 байт.

char - символ, int - целое число, float - число с плавающей точкой, double - float с удвоенной точностью.

signed/unsigned - знаковый/беззнаковый, short/long - короткий/длинный (в 2 раза меньше/больше байт занимает в памяти).

void - тип без значения, \_Bool - логический тип.

Структура (struct) - сложный тип данных, состоящий из нескольких полей возможно разных типов.

Арифметические операторы: + – сложение; - – вычитание; \* – умножение; / – деление; % – остаток от деления.

Постфиксная/префиксная инкрементация/декрементация: x++,++x,x--,--x.

int x = 0;

x++ = 0, x = 1

++x = 1, x = 1

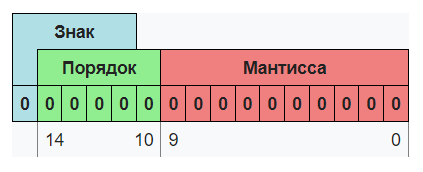
**Целые числа:**

Прямой код представляет собой одинаковое представление значимой части числа для положительных и отрицательных чисел и отличается только знаковым битом. В прямом коде число 0 имеет два представления «+0» и «–0».

Обратный код для положительных чисел имеет тот же вид, что и прямой код, а для отрицательных чисел образуется из прямого кода положительного числа путем инвертирования всех значащих разрядов прямого кода. В обратном коде число 0 также имеет два представления «+0» и «–0».

Дополнительный код для положительных чисел имеет тот же вид, что и прямой код, а для отрицательных чисел образуется путем прибавления 1 к обратному коду. Добавление 1 к обратному коду числа 0 дает единое представление числа 0 в дополнительном коде. Однако это приводит к асимметрии диапазонов представления чисел относительно нуля.

**Числа с плавающей запятой:**



Знак — один бит, указывающий знак всего числа с плавающей точкой. Порядок и мантисса — целые числа, которые вместе со знаком дают представление числа с плавающей запятой в следующем виде:

(−1)S×M×BE, где S — знак, B — основание, E — порядок, а M

— мантисса. Десятичное число, записываемое как ReE, где R

— число в полуинтервале [1;10), E

— степень, в которой стоит множитель 10 ; в нормализированной форме модуль R будет являться мантиссой, а E

— порядком, а S будет равно 1

тогда и только тогда, когда R принимает отрицательное значение. Например, в числе −2435e9

S = 1

B = 10

M = 2435

E = 9



Результатом логической операции является либо 0, либо 1. Результат имеет тип int. && - логическое и, || - логическое или.

Результатом реляционного выражения является 1, если проверенная связь имеет значение true, и 0, если значение false. Результат имеет тип int.

< Первый операнд меньше второго операнда

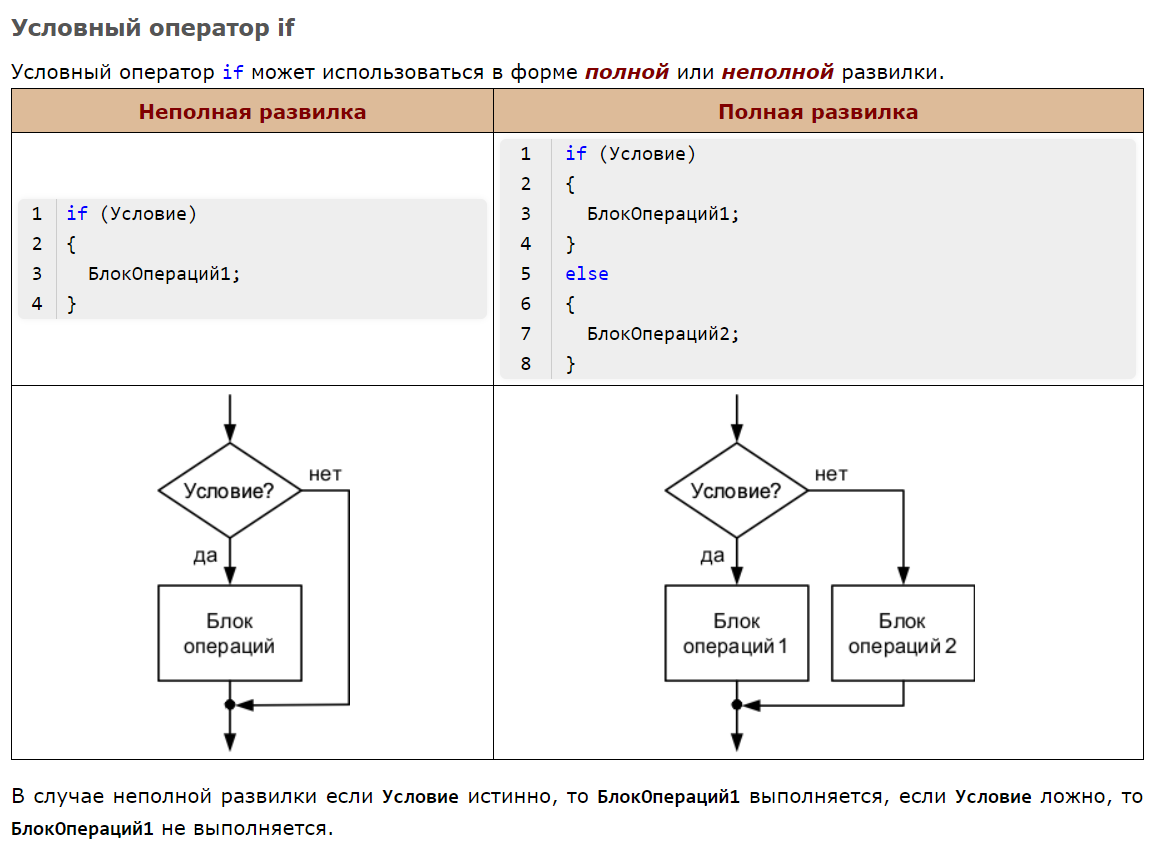
> Первый операнд больше второго операнда

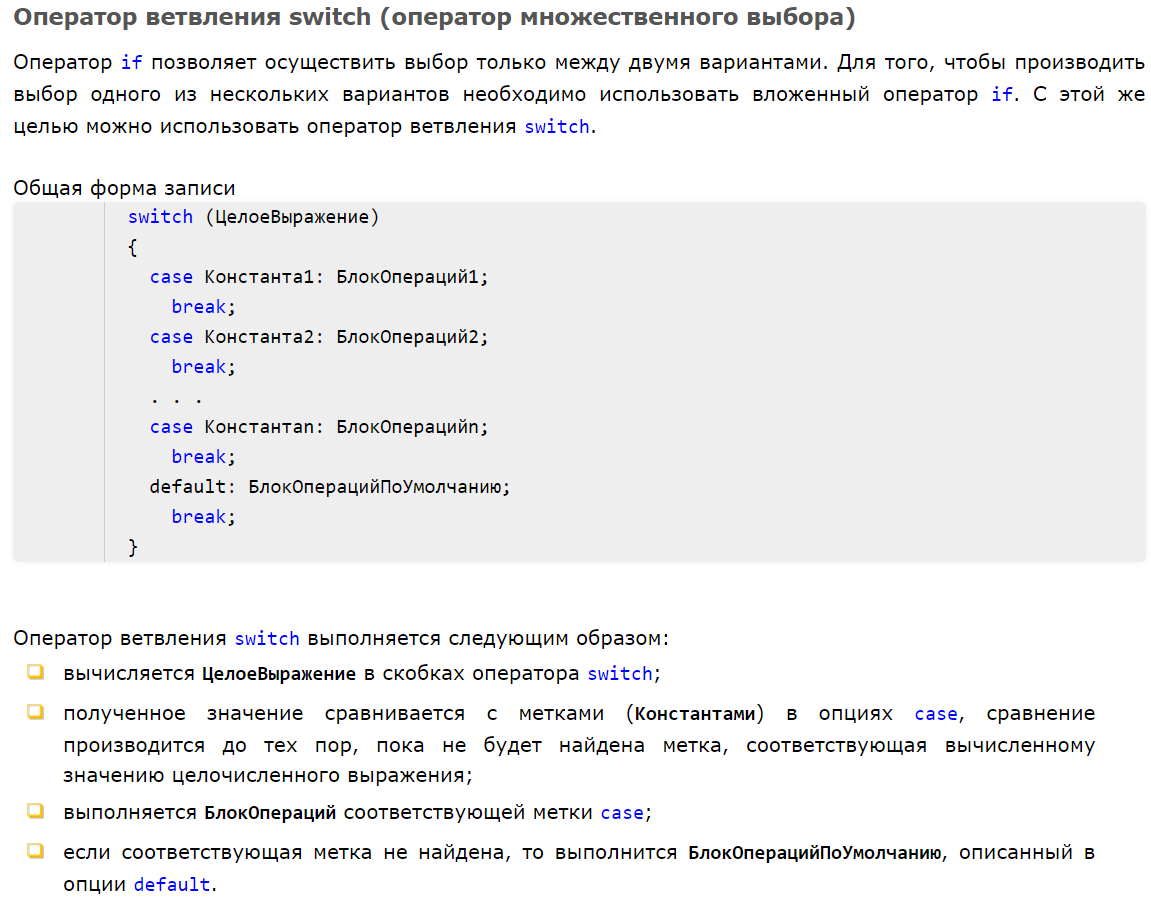
<= Первый операнд меньше или равен второму операнду

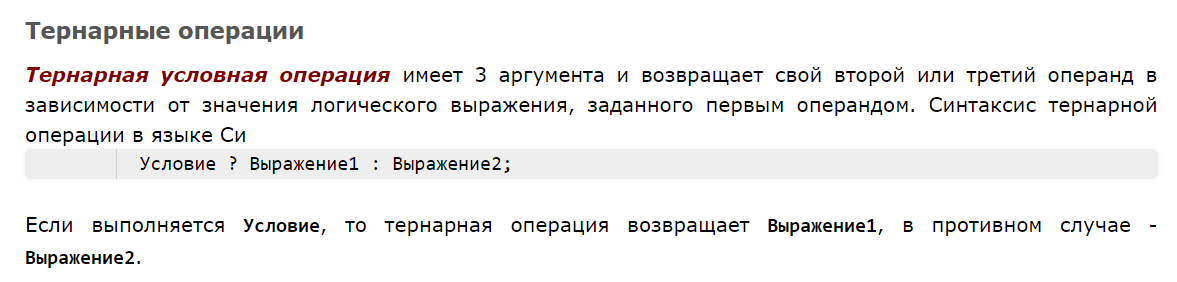
>= Первый операнд больше или равен второму операнду

== Первый операнд равен второму операнду

!= Первый операнд не равен второму операнду

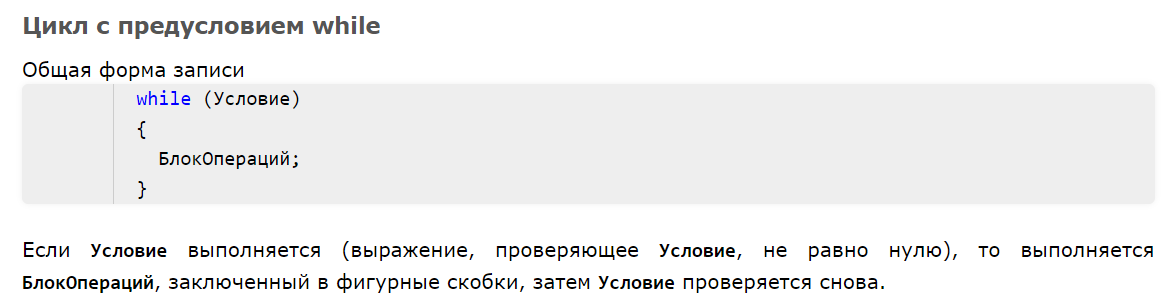


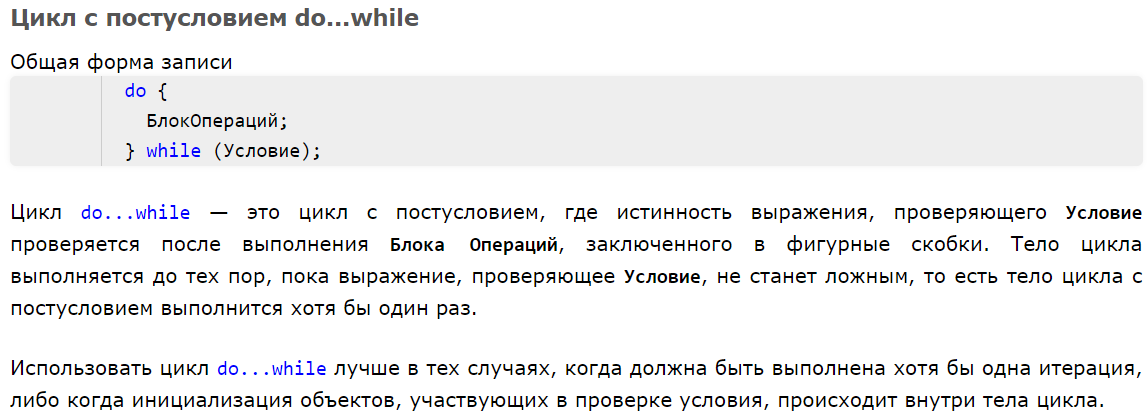


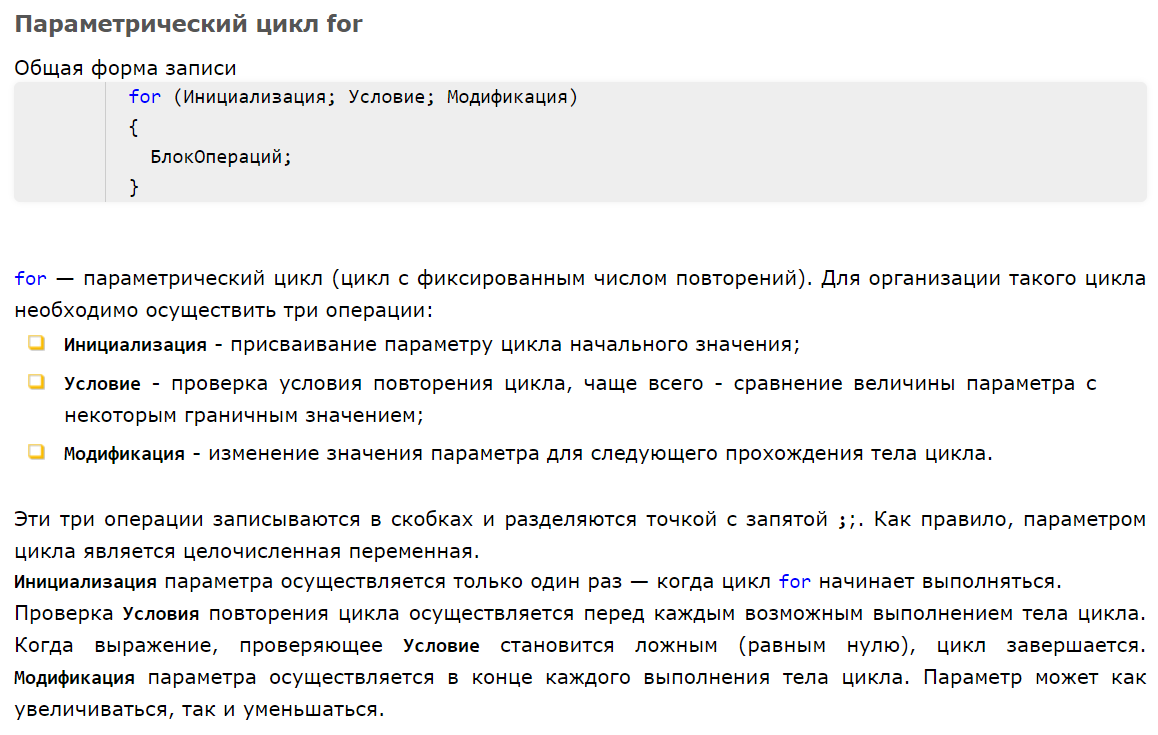


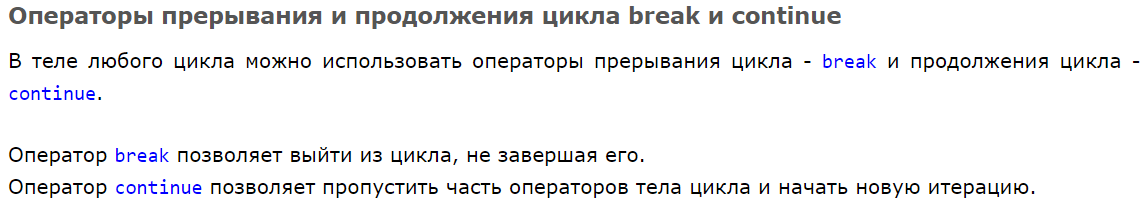


Циклом называется блок кода, который для решения задачи требуется повторить несколько раз.  
Каждый цикл состоит из блока проверки условия повторения цикла и тела цикла.



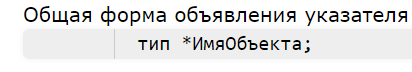








Указатель — переменная, содержащая адрес объекта.



Тип указателя— это тип переменной, адрес которой он содержит.

Для работы с указателями в Си определены две операции:

операция \* (звездочка) — позволяет получить значение объекта по его адресу — определяет значение переменной, которое содержится по адресу, содержащемуся в указателе;

операция & (амперсанд) — позволяет определить адрес переменной.

Арифметика указателей: увеличение на единицу означает, что мы хотим перейти к следующему объекту в памяти, который находится за текущим и на который указывает указатель. А уменьшение на единицу означает переход назад к предыдущему объекту в памяти. Аналогично указатель будет изменяться при прибавлении/вычитании не единицы, а какого-то другого числа. В отличие от сложения операция вычитание может применяться не только к указателю и целому числу, но и к двум указателям одного типа.



Массив — это непрерывный участок памяти, содержащий последовательность объектов одинакового типа, обозначаемый одним именем.

Массив характеризуется следующими основными понятиями:

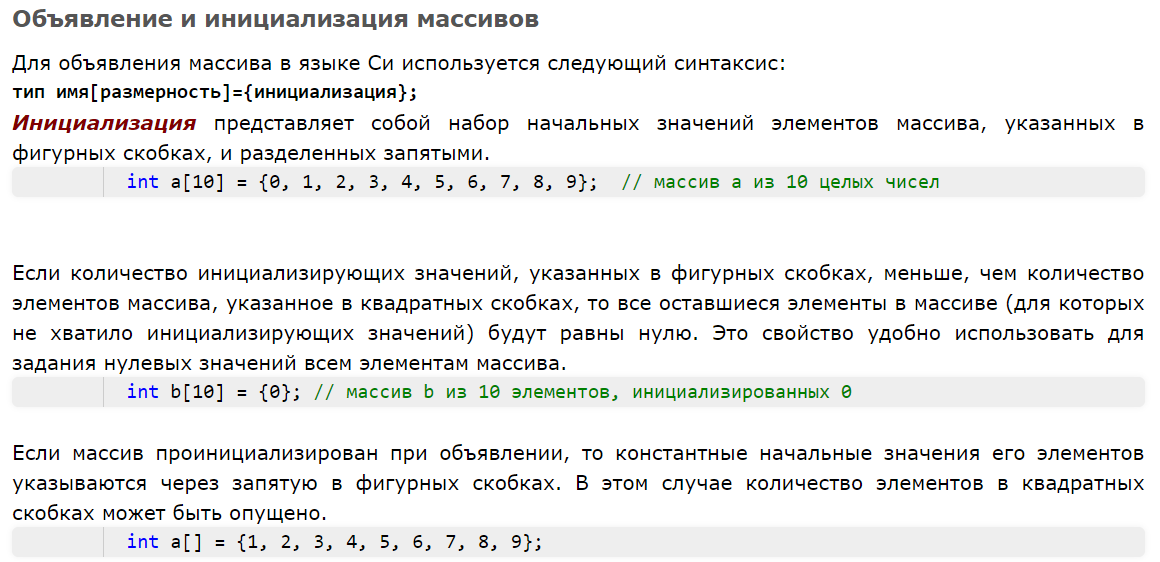
Элемент массива (значение элемента массива) – значение, хранящееся в определенной ячейке памяти, расположенной в пределах массива, а также адрес этой ячейки памяти.

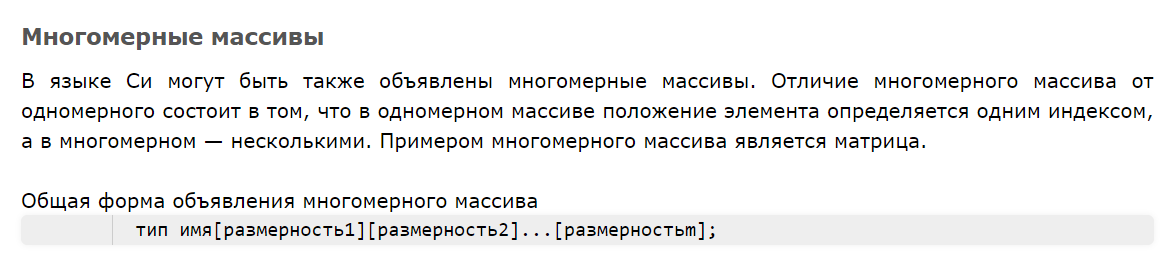
Каждый элемент массива характеризуется тремя величинами:

адресом элемента — адресом начальной ячейки памяти, в которой расположен этот элемент; индексом элемента (порядковым номером элемента в массиве);

значением элемента.

Адрес массива – адрес начального элемента массива.





Передача массива в функцию

Обработку массивов удобно организовывать с помощью специальных функций. Для обработки массива в качестве аргументов функции необходимо передать адрес массива и размер массива.

Исключение составляют функции обработки строк, в которые достаточно передать только адрес.

При передаче переменные в качестве аргументов функции данные передаются как копии. Это означает, что если внутри функции произойдёт изменение значения параметра, то это никак не повлияет на его значение внутри вызывающей функции.