Операции и выражения

Операторы

Классификация операций

- **Операция** — конструкция в языках программирования, аналогичная по записи математическим операциям, то есть специальный способ записи некоторых действий.

- Наиболее часто применяются:
- арифметические,
- логические
- строковые

Типы операций

- Операции делятся по количеству принимаемых аргументов на:

- *унарные* — один аргумент (отрицание, унарный минус);

- *бинарные* — два аргумента (сложение, вычитание, умножение и т. д.);

- *тернарные* — три аргумента («условие ? выражение1 : выражение2»).

Типовые операции

Знак	Выполняемая операция
a = b	присваивание
Арифметические	
a + b	сложение аргументов
a - b	вычитание
-a	изменение знака
a / b, a div b	деление
a % b, a mod b	остаток от деления (деление по модулю)
a++ a--	увеличение на 1 с присваиванием (инкремент) уменьшение на 1 с присваиванием (декремент)
a ^ b **ИЛИ** a ** b	возведение в степень
Логические	
a & b, **ИЛИ** a && b, **ИЛИ** a and b	И
a | b, **ИЛИ** a || b, **ИЛИ** a or b	ИЛИ
~a, **ИЛИ** !a, **ИЛИ** not a	НЕ
a = b **ИЛИ** a == b a <> b **ИЛИ** a != b	проверка на равенство проверка на неравенство
a > b, a >= b a < b, a <= b	больше, больше или равно меньше, меньше или равно
a ? b : c	тернарный условный оператор (если условие a истинно, всё выражение равно b, иначе c)

Операции инкремента и декремента

- **Инкремент** операция увеличивающая переменную.
- Обратную операцию называют **декремент**. Чаще всего унарная операция приводит переменную к следующему элементу данного типа
- «префиксный декремент» **--X** и «постфиксный декремент» **X--** действуют аналогично на переменную **x**, уменьшая её.

- **x = x + 1** тоже самое что **и** **x += 1**

Арифметические операции

- **+** — сложение;
- **-** — вычитание;
- ***** — умножение;
- **/** — деление;
- **%** — остаток от деления.

Арифметика

```
// arithmetic.cpp: определяет точку входа для консольного приложения.
#include "stdafx.h"
#include <iostream>
using namespace std;

int _tmain(int argc, char* argv[])
{
    double sum, razn, pow, div; // объявление переменных через запятую
    double a1; // отдельное объявление переменной a1
    double a2; // отдельное объявление переменной a2
    cout << "Vvedite pervoe chislo: ";
    cin >> a1;
    cout << »Введите второе число: ";
    cin >> a2;
    sum  = a1 + a2;  // операция сложения
    razn = a1 - a2;  // операция вычитания
    pow  = a1 * a2;  // операция умножения
    div  = a1 / a2;  // операция деления
    cout << a1 << "+" << a2 << "=" << sum  << endl;
    cout << a1 << "-" << a2 << "=" << razn << endl;
    cout << a1 << "*" << a2 << "=" << pow  << endl;
    cout << a1 << "/" << a2 << "=" << div  << endl;
    system ("pause");
    return 0;
}
```

логические операции

- Логическая операция И &&;
- Логическая операция ИЛИ ||;
- Логическая операция НЕ ! или логическое отрицание

Операции	Обозначение	Условие	Краткое описание
И	&&	a == 3 && b > 4	Составное условие истинно, если истинны оба простых условия
ИЛИ	||	a == 3 || b > 4	Составное условие истинно, если истинно, хотя бы одно из простых условий
НЕ	!	!(a == 3)	Условие истинно, если a не равно 3

логика

```
// or_and_not.cpp: определяет точку входа для консольного приложения.
#include "stdafx.h"
#include <iostream>
using namespace std;
int main(int argc, char* argv[])
{
    bool a1 = true, a2 = false; // объявление логических переменных
    bool a3 = true, a4 = false;
    cout << "Tablica istinnosti log operacii &&" << endl;
    cout << "true  && false: " << ( a1 && a2 )   << endl // логическое И
        << "false && true: "  << ( a2 && a1 )   << endl
        << "true  && true: "  << ( a1 && a3 )   << endl
        << "false && false: " << ( a2 && a4 )   << endl;
    cout << "Tablica istinnosti log operacii ||" << endl;
    cout << "true  || false: " << ( a1 || a2 )   << endl // логическое ИЛИ
        << "false || true: "  << ( a2 || a1 )   << endl
        << "true  || true: "  << ( a1 || a3 )   << endl
        << "false || false: " << ( a2 || a4 )   << endl;
    cout << "Tablica istinnosti log operacii !"  << endl;
    cout << "!true: " << ( ! a1 ) << endl // логическое НЕ
        << "!false: "<< ( ! a2 ) << endl;
    system("pause");
    return 0;
}
```

Побитовые операции

- Битовые операции — это тестирование, установка или сдвиг битов в байте или слове, которые соответствуют стандартным типам языка C char и int. Битовые операторы не могут использоваться с float, double, long double, void и другими сложными типами.

Оператор	Действие
&	И
|	ИЛИ
^	Исключающее ИЛИ
~	Дополнение
>>	Сдвиг вправо
<<	Сдвиг влево

операции отношений

Операторы отношения

Оператор	Действие
>	Больше чем
>=	Больше чем или равно
<	Меньше чем
<=	Меньше чем или равно
==	Равно
!=	Не равно

Приоритеты операций

- **Приоритет операций** — очерёдность выполнения операций в выражении, при условии, что в выражении нет явного указания порядка следования выполнения операций (с помощью круглых скобок).

Приоритет	Операция	Ассоциативность	Описание
1	::	слева направо	унарная операция разрешения области действия
	[ ]		операция индексирования
	()		круглые скобки
	.		обращение к члену структуры или класса
	->		обращение к члену структуры или класса через указатель
2	++	слева направо	постфиксный инкремент
	—		постфиксный декремент
3	++	справа налево	префиксный инкремент
	—		префиксный декремент
4	*	слева направо	умножение
	/		деление
	%		остаток от деления
5	+	слева направо	сложение
	—		вычитание
6	>>	слева направо	сдвиг вправо
	<<		сдвиг влево
7	<	слева направо	меньше
	<=		меньше либо равно
	>		больше
	>=		больше либо равно
8	==	слева направо	равно
	!=		не равно
9	&&	слева направо	логическое И
10	||	слева направо	логическое ИЛИ
11	?:	справа налево	условная операция (тернарная операция)
12	=	справа налево	присваивание
	*=		умножение с присваиванием
	/=		деление с присваиванием
	%=		остаток от деления с присваиванием
	+=		сложение с присваиванием

Операторы. Общая классификация

- Операторы управляют процессом выполнения программы.
- Составной оператор ограничивается фигурными скобками. Все другие операторы заканчиваются точкой с запятой.
- **Пустой оператор – ;**
- **Составной оператор – {…}**
- **Оператор обработки исключений**
  try { *<операторы>* }
- **catch** (*<объявление исключения>*) { *<операторы>* }
- **catch** (*<объявление исключения>*) { *<операторы>* }
- **… catch** (*<объявление исключения>*) { *<операторы>* }

Операторы. Общая классификация

- **Условный оператор**
  if (*<выражение>*) *<оператор 1>* [**else** *<оператор 2>*]
- **Оператор-переключатель**
  switch (*<выражение>*) { **case** *<константное выражение 1>*:
  <операторы 1> **case** *<константное выражение 2>*:
  <операторы 2> **... case** *<константное выражение N>*:
  <операторы N> [**default**: *<операторы>*] }
- **Оператор цикла с предусловием**
  while (*<выражение>*) *<оператор>*
- **Оператор цикла с постусловием**
  do *<оператор>* **while** *<выражение>*;

Операторы. Общая классификация

- **Оператор пошагового цикла**
  for ([*<начальное выражение>*]; [*<условное выражение>*]; [*<выражение приращения>*]) *<оператор>*
- **Оператор разрыва**
  break;
- **Оператор продолжения**
  continue;
- **Оператор возврата**
  return [*<выражение>*];

Операторы управления

- механизмы, с помощью которых можно изменять порядок выполнения программы.
- С предоставляет три категории операторов управления программой: итерационные операторы, операторы выбора и операторы переходов.
- Итерационные операторы - это while, for и do/while. Они чаще всего называются циклами.
- Операторы выбора или условные операторы - это if и switch.
- Операторы перехода - это break, continue и ~~goto~~. (Оператор return, в принципе, также является оператором перехода, поскольку он воздействует на программу.) Функция exit() она также влияет на выполнение программы.

Операторы- выражения

- *Выражение* представляет собой последовательность из одного или нескольких операндов и от нуля до нескольких операторов, которую можно вычислить, получив в результате одно значение, объект, метод или пространство имен. Выражение может состоять из литерала, вызова метода, оператора или его операндов, а также из *простого имени*. Простые имена могут быть именами переменной, элемента типа, параметра метода, пространства имен или типа.

- **((x < 10) && (x > 5)) || ((x > 20) && (x < 25));**
  System.Convert.ToInt32("35");

Операторы выбора

- два оператора выбора:
  1)Оператор выбора if
  2) Оператор выбора switch

Операция в C++	Условие	Смысл записанных условий в C++
==	a == b	a равно b
!=	a != b	a не равно b
>	a > b	a больше b
<	a < b	a меньше b
>=	a >= b	a больше или равно b
<=	a <= b	a меньше или равно b

if

- if (/*проверяемое условие*/)
- {/*оператор1*/;}
- else {/*оператор2*/;}

switch

- switch (/*variable*/) {
- case const1:
- /*Тут находится код, который необходимо выполнить, если переменная variable будет равна const1*/
- break;
- case const2:
- /*этот код выполнится, если variable будет равна const2*/
- break;
- /*...*/
- default:
- /*Код, который выполнится, если ниодно из константых значению не соответствует значение в переменной variable*/
- break;
- }

Правила организации циклических алгоритмов

- В C# имеются четыре различных вида циклов (for, while, do...while и foreach), позволяющие выполнять блок кода повторно до тех пор, пока удовлетворяется определенное условие.

- *for (инициализатор; условие; итератор) оператор (операторы)*

- *while(условие) оператор (операторы);*

Функции

- Функция (в программировании) — это фрагмент кода или алгоритм, реализованный на каком-то языке программирования, с целью выполнения определённой последовательности операций.

- в С. предусмотрено объявление своих функций, также можно воспользоваться функциями определёнными в стандартных заголовочных файлах языка программирования С. Чтобы воспользоваться функцией, определённой в заголовочном файле, нужно его подключить. Например, чтобы воспользоваться функцией, которая возводит некоторое число в степень, нужно подключить заголовочный файл <cmath> и в запустить функцию pow() в теле программы.

Объявление и определение функции,

вызовы функций

тип возврата

глобальные переменные

формальные и фактические параметры