

### 3.4. Операції

Для здійснення маніпуляцій з даними мова С++ застосовує широкий набір операцій (див. *табл. 3.4*), що виконують формування і, відповідно, подальше обчислення виразів. Вирази містять одну або декілька операцій, об'єкти яких називають операндами. **Операції** являють собою деяку дію, що виконується над **одним (унарна)** або **декількома (бінарна, триарна)** операндами, і мають позначення (наприклад, операція перевірки на рівність — позначення «==»), операція обчислення залишку від ділення цілих чисел — позначення «%» тощо).

Операції поділяються на:

- **унарні** або **одномісні** — **&, \*, -, +, ~, !, ++, --, sizeof;**
- **бінарні** або **двомісні** — **+, -, \*, /, %, <<, >>, &, :, ^, <, >, <=, ==, >=, !=, &&, ||, =, \*=, /=, %=, +=, -=, <<=, >>=, &=, |=, ^=, ., ->, ,, (), [ ];**
- умовну **триарну** або **тримісну** операцію — **?:** .

*Таблиця 3.4*

**Основні операції мови С++**

№	Операції	Порядок виконання
1	() , { } -> .	Л -> П
2	! ~ ++ -- & * (type)	П ->Л
3	sizeof	П ->Л
4	* / %	Л -> П
5	+ -	Л -> П
6	<< >>	Л -> П
7	< <= > >=	Л -> П
8	== !=	Л -> П
9	&	Л -> П
10	^	Л -> П
11		Л -> П
12	&&	Л -> П
13		Л -> П
14	?:	П ->Л
15	= += *= -= /= %=	П ->Л
16	,	Л -> П

Порядок застосування операції визначається **пріоритетом операції** (яка операція виконується раніше, а яка пізніше) та **асоціативністю** (виконується зліва направо або справа на ліво). У першу чергу реалізуються операції з найвищим пріоритетом.

У *табл. 3.4* літерою «Л» позначено величину, що стоїть ліворуч від знака операції, літерою «П» — величину, яка розташована праворуч від знака операції, а символом «->» напрямком виконання операції. Розглянемо основні операції.

### Арифметичні операції:

+ — додає величину **П** до **Л**;

- — віднімає **П** із **Л**;

- — унарна операція зміни знака величини **П**;

\* — множення **П** і **Л**;

/ — ділення **Л** на **П**;

% — залишок від ділення величини **Л** на величину **П** (для цілих чисел), наприклад, якщо **int g = 12**;, то операція **g = g % 9**; надасть результат: **g = 3**;

++ — унарна операція інкремент. Якщо змінна розташовується праворуч від знака операції (префіксна форма), то значення збільшується на 1 до використання. Якщо ж змінна знаходиться ліворуч від знака операції (постфіксна форма), то її значення збільшується на 1 після використання, наприклад:

**int d**;

**++d**; — префіксний інкремент,

**d++**; — постфіксний інкремент;

-- — унарна операція декремент аналогічно інкременту має дві форми: префіксну (змінна розташована праворуч від знака операції) — зменшення значення змінної на 1 відбувається до її використання; постфіксну (змінна знаходиться ліворуч від знака операції) — зменшення значення змінної на 1 після її використання.

### Операції присвоювання:

= — присвоювання значення **П** змінній **Л**;

+= — додає величину **П** до змінної **Л**;

-= — віднімає величину **П** від змінної **Л**;

\*= — множення змінної **Л** на величину **П**;

/= — ділення **Л** на **П**;

%= — видає залишок від ділення **Л** на **П**.

Просте присвоювання здійснює операція «**=**». Допускається одночасне зчіплювання декількох операцій присвоювання за умови, що всі операнди мають однаковий тип, наприклад:

**int i, j, c**;

**i = j = c = 0**; .

Операції «**+=**», «**-=**», «**\*=**», «**/=**» виконують складні присвоювання і дозволяють записувати вирази коротше, наприклад:

**s += 7**;      //s=s + 7;

**i \*= j + 5**;    //i=i\*(j +5);

**g%=9**;      //g=g%9;.

Операції відношення порівнюють значення **Л** зі значенням **П**:

< — менше;

<= — менше або дорівнює (не перевищує);

== — дорівнює;

> — більше;

>= — більше або дорівнює (не менше);

!= — не дорівнює.

У мові C++ «істина» — це ненульова величина, «неправда» — це нуль (**0**). У більшості випадків одиниця (**1**) використовується як ненульове значення.

Операції відношення повертають ціле значення 1, якщо умова вірна, або 0, якщо умова помилкова.

**Логічні операції** оперують з цілими розмірами або з розмірами, які можна перетворити на цілі. Обчислення зупиняється, які тільки визначиться, чи є вираз правдивим («істина») або помилковим («неправда»). При цьому, як і для операцій відношення, значенням «істина» відповідає 1, а значенням «неправда» — 0.

**&&** — логічне «**AND**» (кон'юнкція);

**||** — логічне «**OR**» (диз'юнкція);

**!=** — логічне «**NOT**» (заперечення).

Результат операції «**&&**» є «істина» (**1**), якщо обидва її операнди правдиві (не рівні 0). Результат операції «**||**» — «істина» (**1**), якщо хоча б один з її операндів є «істина». Логічне заперечення «**!=**» перетворює свій операнд на «істину» (**1**), якщо він дорівнює **0**, і на «неправду» (**0**), якщо він не дорівнює **0**.

З використанням логічних операцій та операцій відношення записуються різні умовні вирази, наприклад, умова  $3 < x < 5$  матиме вигляд:  $x > 3 \ \&\& \ x < 5$ .

**Операції обробки окремих бітів** застосовують для обробки даних як послідовностей бітів (розрядів), кожний з яких набуває значення **0** або **1**.

**&** — операція бітового множення (кон'юнкція);

**|** — операція бітового додавання (диз'юнкція);

**^** — додавання за модулем 2;

**~** — інвертування;

**>>** — зсув праворуч;

**<<** — зсув ліворуч.

**Змінна-показчик** зберігає значення, що є адресою об'єкта в пам'яті комп'ютера. Через показчик можна звертатися до об'єкта.

**Операції з адресами та показчиками:**

**&** — одержання адреси: видає адресу змінної, ім'я якої розташоване праворуч від позначення операції;

**\*** — непряма адресація (розіменування): видає значення, записане за адресою, на яку посилається показчик.

**Додаткові операції:**

**sizeof()** — знаходить розмір (у байтах) операнда, розташованого праворуч від назви операції;

**(type)** — операція приведення типу перетворює наступне за нею значення в тип, визначений ключовим словом, укладеним у круглі дужки, наприклад:

**i = i+(int)\*3.14;**

**?:** — **триарна** (з трьома операндами) операція, що має вигляд:

**вираз1? вираз2 : вираз3; ,**

тут, якщо результат обчислення першого операнда (**вираз1**) не дорівнює **0** («істина»), то результатом операції буде значення другого операнда (**вираз2**), інакше — третього операнда (**вираз3**). Наприклад, знаходження найбільшої з двох величин **a** і **b**, можливо здійснити операцією: **max = (b > a)? b : a;**

Мова C++ налічує широкий спектр математичних функцій (табл. 3.5). Для їх використання слід включити в код програми заголовний файл **math.h**.

Таблиця 3.5

**Математичні функції (заголовний файл **math.h**)**

Прототип функції	Ім'я	Призначення
Double sin (double _x);	<b>sin (x)</b>	синус x (в радіанах) — <b>sin x</b>
Double cos (double _x);	<b>cos (x)</b>	косинус x (в радіанах) — <b>cos x</b>
Double tan (double _x);	<b>tan (x)</b>	тангенс x (в радіанах) — <b>tg x</b>
Double asin (double _x);	<b>asin (x)</b>	арксинус x — <b>arcsin x</b>
Double acos (double _x);	<b>acos (x)</b>	арккосинус x — <b>arccos x</b>
Double atan (double _x);	<b>atan (x)</b>	арктангенс x — <b>arctg x</b>
Double atan2 (double _y, Double_x);	<b>atan2 (y,x)</b>	арктангенс y/x — <b>arctg (y/x)</b>
Double sinh (double _x);	<b>sinh (x)</b>	синус гіперболічний x — <b>sh x</b>
Double cosh (double _x);	<b>cosh (x)</b>	косинус гіперболічний x — <b>ch x</b>
Double tanh (double _x);	<b>tanh (x)</b>	тангенс гіперболічний x — <b>th x</b>
Double log (double _x);	<b>log (x)</b>	натуральний логарифм x — <b>ln x</b>
Double log10 (double _x);	<b>log10 (x)</b>	десятковий логарифм x — <b>log x</b>
Double exp (double _x);	<b>exp (x)</b>	піднесення e до степеня x — <b>e<sup>x</sup></b>
Double pow (double _x, double_y);	<b>pow (x,y)</b>	піднесення x до степеня y — <b>x<sup>y</sup></b>
Double pow 10 (int _p)	<b>pow10 (p)</b>	повертає <b>10<sup>p</sup></b>
Double sqrt (double _x);	<b>sqrt (x)</b>	корінь із x, x > 0
Double hypot (double_x, double_y);	<b>hypot (x,y)</b>	корінь із (x <sup>2</sup> +y <sup>2</sup> )
Double fabs (double __x);	<b>fabs (x)</b>	абсолютне значення x —  x  типу <b>double</b>
int abs (int _x);	<b>abs (x)</b>	абсолютне значення x —  x  типу <b>int</b>
long labs (long _x);	<b>labs (x)</b>	абсолютне значення x —  x  типу <b>long</b>
Double fmod (double __x, double_y);	<b>fmod (x,y)</b>	залишок від ділення x на y
Double ceil (double __x);	<b>ceil (x)</b>	округлення до більшого
Double floor (double _x);	<b>floor (x)</b>	повертає найближче ціле, не більше за x
Double modf (double _x, double);	<b>modf(x,&amp;p)</b>	виділяє цілу й дробову частинні числа
Double atof(const char* _s);	<b>atof (s)</b>	перетворює рядок символів у число з плаваючою крапкою

Визначені константи: **M\_PI** = 3.1415... —  $\pi$ , **M\_E** = 2.71828... — e, **M\_SQRT2** = 1.4142... — **sqrt(2)**, **M\_LN2** = 0.6931... — **ln(2)** тощо.