### Оператори та вирази

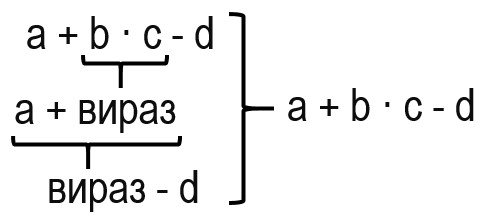
*Оператор* – це символ (або група символів), які представляють операцію, що повертає єдиний результат. *Операнд* – це елемент вхідних даних для операції. Оператор виконує наступне:

► Отримує операнди на вході.

► Виконує дію.

► Повертає значення результату.

*Вираз* – це рядок, який містить оператори та операнди і повертає результат певного типу. В ролі операндів можуть виступати літерали, константи, змінні, виклики методів, або інші вирази. Вирази за допомогою операторів можна поєднувати у складніші вирази:



Обчислення виразу полягає у застосуванню кожного оператора до його операндів у правильній послідовності, в результаті чого отримується результат. Він передається у те місце програми, де у коді розташовано вираз, і може бути використаний як операнд для іншого виразу.

#### Літерали

*Літерал* – це значення (числове, текстове чи інше), яке вписане безпосередньо у код програми. Наприклад, приведений код використовує текстовий та цілочисельний літерали:

Те

кстовий літерал

Console.WriteLine("Назва = {0}", "Телефон Nokia");

Console.WriteLine("Кількість = {0}", 28);

Цілочисельний літерал

Деякі типи даних мають свої власні літерали. Наприклад, тип bool має два літерали: true і false. Для змінних посилального типу використовують літерал null, який представляє значення null.

При позначенні літералів цілого типу можна вказувати суфікс, який задасть точний тип даних (таблиця 2.8).

Таблиця №2.8. Суфікси для літералів цілих типів

|  |  |  |
| --- | --- | --- |
| **Суфікс** | **Тип** | **Зауваження** |
| **Без суфікса** | int, uint, long, ulong |  |
| **U, u** | uint, ulong |  |
| **L, l** | long, ulong | Використовувати маленьку літеру "l" не рекомендовано, бо її можна сплутати з цифрою "1" |
| **ul, uL, Ul, UL, lu, lU, Lu, LU** | ulong |

Приклади цілих літералів із вказуванням типу:

123 // int

123L // long

123U // unsigned int

123UL // unsigned long

Суфікси для літералів, які представляють дійсні числа, приведено у таблиці 2.9.

Таблиця №2.9. Суфікси для літералів дійсних типів

|  |  |
| --- | --- |
| **Суфікс** | **Тип** |
| **Без суфікса** | double |
| **F, f** | float |
| **D, d** | double |
| **M, m** | decimal |

Символьні літерали містять символ в одинарних лапках. Вони можуть представляти: окремий символ, просту ESC-послідовність[[1]](#footnote-1), шістнадцяткову ESC-послідовність, або код символу Unicode. Найвживаніші символьні літерали приведено в таблиці 2.10.

Таблиця №2.10. Символьні літерали

|  |  |
| --- | --- |
| **Літерал** | **Опис** |
| **\0** | Null |
| **\a** | "Обережно" (звуковий сигнал) |
| **\b** | Backspace (стирає попередній символ) |
| **\t** | Горизонтальна табуляція |
| **\n** | Новий рядок |
| **\r** | Перехід на початок рядка |
| **\"** | Подвійна лапка |
| **\'** | Одинарна лапка |
| **\\** | Символ "\" |

Рядкові літерали є двох типів: звичайні та дослівні. Такі літерали оточують подвійними лапками. Вони можуть містити символи, символьні літерали (див. табл. 2.10) та ESC-послідовності. Перед дослівними літералами ставлять символ "@". У них символьні літерали та ESC-послідовності не обчислюються. Приклад використання рядкових літералів:

Console.WriteLine("Add \t \'Interest\'");

Console.WriteLine(@"Add \t \'Interest\'"); В результаті виконання цих інструкцій отримаємо:

Add 'Interest'

Add \t \'Interest\'

У першому рядку використано табуляцію (\t), в другому відповідний літерал відображується, ніби він є складовою частиною тексту.

1. ESC-послідовність – це керуюча послідовність символів, яка використовується для керування виводом тексту на екран чи принтер. [↑](#footnote-ref-1)