# Тема 2. Основи C#

**Лекція №4:** Оператори та операції у C#.

**План**

1. Основні поняття про оператори, операції, види та приорітет операцій;
2. Оператори умовного та безумовного переходів;
3. Оператори циклу;
4. Операції присвоєння, математичні операції, операції інкременту/декременту та логічні операції;

**Література**


1. Jeff Ferguson, Brian Patterson, Jason Beres, Pierre Boutquin, and Meeta Gupta C# Bible Wiley Publishing, Inc., 2002. — 590 с.: ил.
2. Рейли Д. Создание приложений Microsoft ASP.NET/Пер. с англ. — М.; Издательско-торговыи дом «Русская Редакция», 2002. — 480 с.: ил.
3. MSDN
4. Microsoft Corporation, C# Language Specification, Version 1.2, 392 с.


**Контрольні запитання**

1. Які відмінності між використанням "switch" у C++ та C#?
2. Коли використовується "if", а коли оператор "switch"?
3. Коли варто використовувати оператор "do...while", а коли "while"?
4. Коли використовується "foreach", а коли оператор "for"?
5. Коли використовується "while", а коли оператор "for"?
6. Коли використовується "while", а коли оператор "switch"?
7. Коли використовується префіксна форма операції, а коли постфіксна?
8. Що таке бінарні операції. Наведіть приклад?
9. Що таке унарні операції. Наведіть приклад?
10. Що таке тернарні операції. Наведіть приклад?
11. Чим відрізняється використання логічної операції "!" від операції відношення "!="?
12. Що є результатом операції відношення?

**Завдання для самостійної роботи**

- Дати письмову відповідь на контрольні запитання
- Знайти в Інтернет MSDN, ознайомитися зі змістом та скласти (запропонувати) стратегію вивчення .NET

**Основні поняття про оператори, операції, види та приорітет операцій**

Якщо програма складається з інструкцій, то оператор – це завершена інструкція програми. Оператори поділяються на:

- Оператори умовного та безумовного переходів (if і switch);
- Оператори безумовного переходу (goto);
- Оператори циклу (for, while, do та foreach);

Операції – це дія, що має у програмному коді певне символічне представлення. Серед операцій у C# розрізняють:

- Операції присвоєння (=);
- Математичні операції (+, -, /, *, та %);
- Операції інкременту/декременту;
- Логічні операції (&&, || та !);
- Операції відношення;
- Префіксні та постфіксні операції;

- Бінарні, унарні та тернарні операції.

**Оператори умовного та безумовного переходів**

Оператор умовного переходу використовуються для того, щоб залежно від деякої умови виконати або ж не виконати певний блок коду (оператор). Найпоширенішим представником даного виду є оператор "if":

```
if(умова1)
      оператор1;
else if(умова2)
      оператор2;
else
      оператор3;
```

У випадку, коли деяка змінна дорівнює деякому набору значень, а кожному значенню відповідає певний оператор, то доцільніше використовувати оператор switch:

```
switch(контрольнаЗмінна)
{
case умова1*:
      оператор1;
      break;
case умова2*:
      оператор2;
      break;
default:
      оператор3;
}
```

При цьому контрольнаЗмінна може бути типу "sbyte", "byte", "short", "ushort", "int", "uint", "long", "ulong", "char" або "string".

Оператор бузумовного переходу використовується для переходу до певної частини програмного коду.

**Оператори циклу**

Оператор "while" використовують у випадку необхідності виконати певну кількість разів (від 0 до якоїсь кількості) програмний код, що міститься у "while-body" та не має необхідності використовувати номер ітерації.

`while ( expr ) while-body`

Оператор "while" використовують у випадку необхідності виконати певну кількість разів (від 1 до якоїсь кількості) програмний код, що міститься у "do-body" та не має необхідності використовувати номер ітерації.

`do do-body while ( expr ) ;`

Оператор "for" використовують у випадку необхідності виконати певну кількість разів (від 1 до якоїсь кількості) програмний код, що міститься у "*embedded-statement*" та є необхідність використовувати номер ітерації.

`for ( for-initializer ; for-condition ; for-iterator ) embedded-statement`

Оператор "foreach" використовують у випадку необхідності для кожного елемента деякої колекції об'єктів виконати програмний код, що міститься у "*embedded-statement*" та є необхідність на кожній ітерації використовувати елемент цієї колекції, як екземпляр деякого класу.

foreach ( *type identifier* in *expr* ) *embedded-statement*

**Операції присвоєння, математичні операції, операції інкременту/декременту та логічні операції**

```
Операція присвоєння: a=в;
Операції  інкременту: a+=в;
          декременту: a-=в;
Логічні операції: (a && в), (a || в), (!a);
Операції відношення: (a>b), (a>=b), (a==b), (a<b), (a<=b);
Префіксні та постфіксні операції: --a; a--;
Бінарні, унарні та тернарні операції: (a+в), (a++), (a ? в:c);
```