Перевантаження операторів

Перевантаження операторів

LOGO

Категорія	Операції	Обмеження
Арифметичні бінарні	+, *, /, -, %	Немає
Арифметичні унарні	+, -, ++, --	Немає
Бітові бінарні	&, |, ^, <<, >>	Немає
Бітові унарні	!, ~ true, false	Операції true й false повинні перевантажуватися в парі
Порівняння	>=, <= >, <	Операції порівняння повинні перевантажуватися попарно
Присвоювання	+=, -=, *=, /=, >>=, <<=, %=, &=, |=, ^=	Ці операції не можна явно перевантажити; їхнє перевантаження здійснюється неявно при перевизначенні індивідуальних операцій +, -, % і т. д.
Індексація	[]	Операцію індексу не можна перевантажити явно. Тип члена, що індексує, дозволяє підтримувати операцію індексації в користувацьких класах і структурах
Приведення	()	Операцію приведення не можна перевантажити явно.

Перевантаження операторів

LOGO

```
// Загальний формат перевантаження для унарного оператора
public static тип_результату operator op (тип_параметра операнд)
{
    // операції
}

// Загальний формат перевантаження для бінарного оператора
public static тип_результату operator op (
            тип_параметра1 операнд1, тип_параметра2 операнд2)
{
    // операції
}
```


Правила перевантаження

LOGO

Для унарних операторів тип операнда повинен співпадати з класом, для якого визначений оператор.

Для бінарних операторів, тип хоча б одного операнда повинен співпадати з відповідним класом.

Таким чином, C#-оператори не можна перевантажувати для класів, що не створені Вами. Наприклад, не можна перевантажити оператор "+" для типів int або string.

Параметри операторів не повинні використовувати модифікатори ref або out.

Оператори ++ і --

```
1.  class ThreeD {
2.    int x, y, z; // 3-х-вимірні координати.
3.    public ThreeD () {
4.      x = y = z = 0;
5.    }
6.    public ThreeD (int i, int j, int k) {
7.      x = i;
8.      y = j;
9.      z = k;
10.   }
11.   public static ThreeD operator ++ (ThreeD op) {
12.     // Оператор "++" модифікує аргумент.
13.     op.x ++;
14.     op.y ++;
15.     op.z ++;
16.     return op;
17.   }
18.    // Відображаємо координати X, Y, Z.
19.   public void show () {
20.     Console.WriteLine (x + ", " + y + ", " + z);
21.   }
22. }
```

```
1.  class ThreeDDemo {
2.    public static void Main () {

3.      ThreeD a = new ThreeD (1, 2, 3);
4.
5.      a ++;        // Інкремент a.
6.      Console.Write ("Результат a ++:");
7.      a.show ();
8.    }
9.  }
```

Результат виконання програми

Результат a++: 2, 3, 4

Перевантаження операторів

```
1.  class ThreeD {
2.    int x, y, z; // 3-х-вимірні координати.
3.    public ThreeD () {
4.      x = y = z = 0;
5.    }
6.    public ThreeD (int i, int j, int k) {
7.      x = i;
8.      y = j;
9.      z = k;
10.   }
11.   public static ThreeD operator + (ThreeD op1, ThreeD op2)
12.   {
13.     ThreeD result = new ThreeD ();
14.     result.x = op1.x + op2.x;
15.     result.y = op1.y + op2.y;
16.     result.z = op1.z + op2.z;
17.     return result;
18.   }
19.   public void show () {
20.     Console.WriteLine (x + ", " + y + ", " + z);
21.   }
22. }
```

```
1.  class ThreeDDemo {
2.    public static void Main () {

3.      ThreeD a = new ThreeD (1, 2, 3);
4.      ThreeD b = new ThreeD(10, 10, 10);
5.      ThreeD c = new ThreeD();

6.      c = a + b;
7.      Console.Write ("Результат a + b:");
8.      c.show ();
9.      Console.WriteLine ();
10.   }
11. }
```

Результат виконання програми

Результат a + b: 11, 12, 13

Перевантаження операторів

```
1.  class ThreeD {
2.    int x, y, z; // 3-х-вимірні координати.
3.    public ThreeD () {
4.      x = y = z = 0;
5.    }
6.    public ThreeD (int i, int j, int k) {
7.      x = i;
8.      y = j;
9.      z = k;
10.   }
11.   public static ThreeD operator  –  (ThreeD op1, int op2)
12.   {
13.     ThreeD result = new ThreeD ();
14.     result.x = op1.x – op2;
15.     result.y = op1.y – op2;
16.     result.z = op1.z – op2;
17.     return result;
18.   }
19.   public void show () {
20.     Console.WriteLine (x + ", " + y + ", " + z);
21.   }
22. }
```

```
1.  class ThreeDDemo {
2.    public static void Main () {

3.      ThreeD a = new ThreeD (1, 2, 3);
4.      ThreeD b = new ThreeD(10, 15, 20);
5.      ThreeD c = new ThreeD();

6.      c = b – 5;
7.        // Об'єкт – int-значення
8.
9.      Console.Write ("Результат b – 5: ");
10.     c.show ();
11.   }
12. }
```

Результат виконання програми

Результат b – 5: 5, 10, 15

Перевантаження операторів

```
1.  class ThreeD {
2.    int x, y, z; // 3-х-вимірні координати.
3.    … … … … … … … … … … … … …

4.    public static ThreeD operator  –  (ThreeD op1, int op2)
5.      {
6.        ThreeD result = new ThreeD ();
7.        result.x = op1.x – op2;
8.        result.y = op1.y – op2;
9.        result.z = op1.z – op2;
10.       return result;
11.     }

12.   public static ThreeD operator  –  (int op1, ThreeD op2)
13.     {
14.       ThreeD result = new ThreeD ();
15.       result.x = op1 – op2.x;
16.       result.y = op1 – op2.y;
17.       result.z = op1 – op2.z;
18.       return result;
19.     }
20.   public void show () {
21.     Console.WriteLine (x + ", " + y + ", " + z);
22.   }
23. }
```

```
1.  class ThreeDDemo {
2.    public static void Main () {

3.      ThreeD a = new ThreeD (1, 2, 3);
4.      ThreeD b = new ThreeD(10, 15, 20);
5.      ThreeD c = new ThreeD();

6.      c = b – 5;
7.      Console.Write ("Результат b – 5: ");
8.      c.show ();

9.      c = 30 – b;
10.     Console.Write ("Результат 30 – b: ");
11.     c.show ();
12.   }
13. }
```

Результат виконання програми

Результат b – 5: 5, 10, 15

Результат 30 – b: 20, 15, 10

Оператори > і <

```
1.  class ThreeD {
2.    int x, y, z; // 3-х-вимірні координати.
3.    public ThreeD () {
4.      x = y = z = 0;
5.    }
6.    public ThreeD (int i, int j, int k) {
7.      x = i;
8.      y = j;
9.      z = k;
10.   }
11.  public static bool operator <(ThreeD op1, ThreeD op2) {
12.    if((op1.x < op2.x) && (op1.y < op2.y) && (op1.z < op2.z))
13.      return true;
14.    else
15.      return false;
16.  }
17.  public static bool operator >(ThreeD op1, ThreeD op2) {
18.    if((op1.x > op2.x) && (op1.y > op2.y) && (op1.z > op2.z))
19.     return true;
20.   else
21.     return false;
22.  }
23.  public void show () {
24.    Console.WriteLine (x + ", " + y + ", " + z);
25.  }
26. }
```

```
1.  class ThreeDDemo {
2.    public static void Main () {

3.      ThreeD a = new ThreeD (11, 17, 23);
4.      ThreeD b = new ThreeD(10, 15, 20);
5.      ThreeD c = new ThreeD(20, 20, 20);
6.
7.      if (a > b)
8.      Console.WriteLine("a > b - TRUE");
9.      else
10.     Console.WriteLine("a > b - FALSE");

11.     if (b < c)
12.     Console.WriteLine("b < c - TRUE");
13.     else
14.     Console.WriteLine("b < c - FALSE");
15.   }
16. }
```

Результат виконання програми

a > b - TRUE

b < c - FALSE

Правила перевантаження

На перевантаження операторів відношення накладається серйозне обмеження: їх потрібно перевантажувати парами.

Наприклад, перевантажуючи оператор "<", Ви також повинні перевантажити оператор ">", і навпаки. Ось що мається на увазі під парами операторів відношень:

==	! =
<	>
<=	> =

Перевантажуючи оператори "==" і "!=", потрібно перевантажити також методи Object.Equals() і Object.GetHashCode().

Оператори == і !=

```
class ind_matrix
{
    int rang = matrix.rang;
    public int[] ind_matr = new int[matrix.rang];
    … … … … … … … … … … … … … … … … …
public static bool operator == (ind_matrix MX1, ind_matrix MX2)
{
    for (int i=0; i<matrix.rang; i++)
    {
        if (MX1.ind_matr[i] != MX2.ind_matr[i])
            return false;
    }
    return true;
}
public static bool operator !=(ind_matrix MX1, ind_matrix MX2)
{
    return !(MX1 == MX2);
}
public override bool Equals (object obj)
{
    return (obj is ind_matrix) && (this == (ind_matrix)obj);
}
public override int GetHashCode()
{
    return ToString().GetHashCode();
}
```


Оператори true і false

Оператори true і false повинні бути перевантажені в парі. Не можна перевантажувати тільки один з них. Обидва вони виконують функцію унарних операторів і мають такий формат:

```
public static bool operator true (тип_параметра op)
{
        // Повернення значення true або false.
}

public static bool operator false (тип_параметра op)
{
        // Повернення значення true або false.
}
```

Оператори true і false

```
1.  class ThreeD {
2.    int x, y, z; // 3-х-вимірні координати.
3.    public ThreeD () {
4.      x = y = z = 0;
5.    }
6.    public ThreeD (int i, int j, int k) {
7.      x = i;
8.      y = j;
9.      z = k;
10.   }
11.  public static bool operator true(ThreeD op) {
12.    if((op.x !=0) || (op.y != 0) || (op.z != 0))
13.      return true; // Хоча б одна координата не дорівнює 0.
14.    else
15.      return false;
16.   }
17.  public static bool operator false(ThreeD op) {
18.    if((op.x == 0) && (op.y == 0) && (op.z == 0))
19.      return true; // Всі координати дорівнюють нулю.
20.    else
21.      return false;
22.   }
23.   public void show () {
24.    Console.WriteLine (x + ", " + y + ", " + z);
25.   }
26. }
```

```
1.  class ThreeDDemo {
2.    public static void Main () {

3.      ThreeD a = new ThreeD (1, 2, 3);
4.      ThreeD b = new ThreeD (10, 15, 20);
5.      ThreeD c = new ThreeD (0, 0, 0);
6.
7.      if (b)
8.        Console.WriteLine ("b - TRUE");
9.      else
10.       Console.WriteLine ("b - FALSE");
11.
12.     if (c)
13.       Console.WriteLine ("c - TRUE ");
14.     else
15.       Console.WriteLine ("c - FALSE");
16.   }
17. }
```

Результат виконання програми

```
b  -  TRUE
c - FALSE
```


Оператори перетворення

Якщо оператор перетворення визначений з ключовим словом **implicit**, перетворення виконується автоматично, тобто при використанні об'єкта у виразі, що включає дані типу тип_результату.

Якщо оператор перетворення визначений з ключовим словом **explicit**, перетворення виконується при використанні оператора приведення типів.

Для однієї і тієї ж пари типів, що беруть участь в перетворенні, не можна визначити як explicit-, так і implicit-оператори перетворення.

```
public static explicit operator тип_результату  (вихідний_тип v) {
    return значення;
}
public static implicit operator тип_результату  (вихідний_тип v) {
    return значення;
}
```


Оператори перетворення

```
1.  class ThreeD {
2.    int x, y, z; // 3-х-вимірні координати.
3.    public ThreeD () {
4.      x = y = z = 0;
5.    }
6.    public ThreeD (int i, int j, int k) {
7.      x = i;
8.      y = j;
9.      z = k;
10.   }

11.   public static implicit operator int (ThreeD op1)
12.   {
13.     return op1.x * op1.y * op1.z;
14.   }

15.   public void show () {
16.     Console.WriteLine (x + ", " + y + ", " + z);
17.   }
18. }
```

```
1.  class ThreeDDemo {
2.    public static void Main () {

3.      ThreeD a = new ThreeD (1, 2, 3);
4.      ThreeD b = new ThreeD (10, 15, 20);
5.      ThreeD c = new ThreeD (0, 0, 0);
6.      int i;
7.
8.      i = a;
9.        // Перетворимо в значення типу int
10.     Console.WriteLine (" i = " + i);
11.     Console.WriteLine ();
12.
13.     i = 2 * b;
14.       // Перетворимо в значення типу int
15.     Console.WriteLine (" (2 * b) = " + i);
16.   }
17. }
```

Результат виконання програми

```
i = 6
(2 * b) = 6000
```


Перевантаження операторів

Заборонено перевантажувати оператори:

```
&&          ||          []          ()          new         is
sizeof      typeof      ?           ->          .           =
```

Використання "+=" в програмі автоматично викликає вашу версію методу operator +(). Наприклад,

// при використанні фрагмента коду

ThreeD a = new ThreeD (1, 2, 3);

ThreeD b = new ThreeD (10, 10, 10);

b + = a; ***// підсумовуємо a і b***

Автоматично викликається метод operator + () класу ThreeD, в результаті чого об'єкт b буде містити координати 11,12,13.