

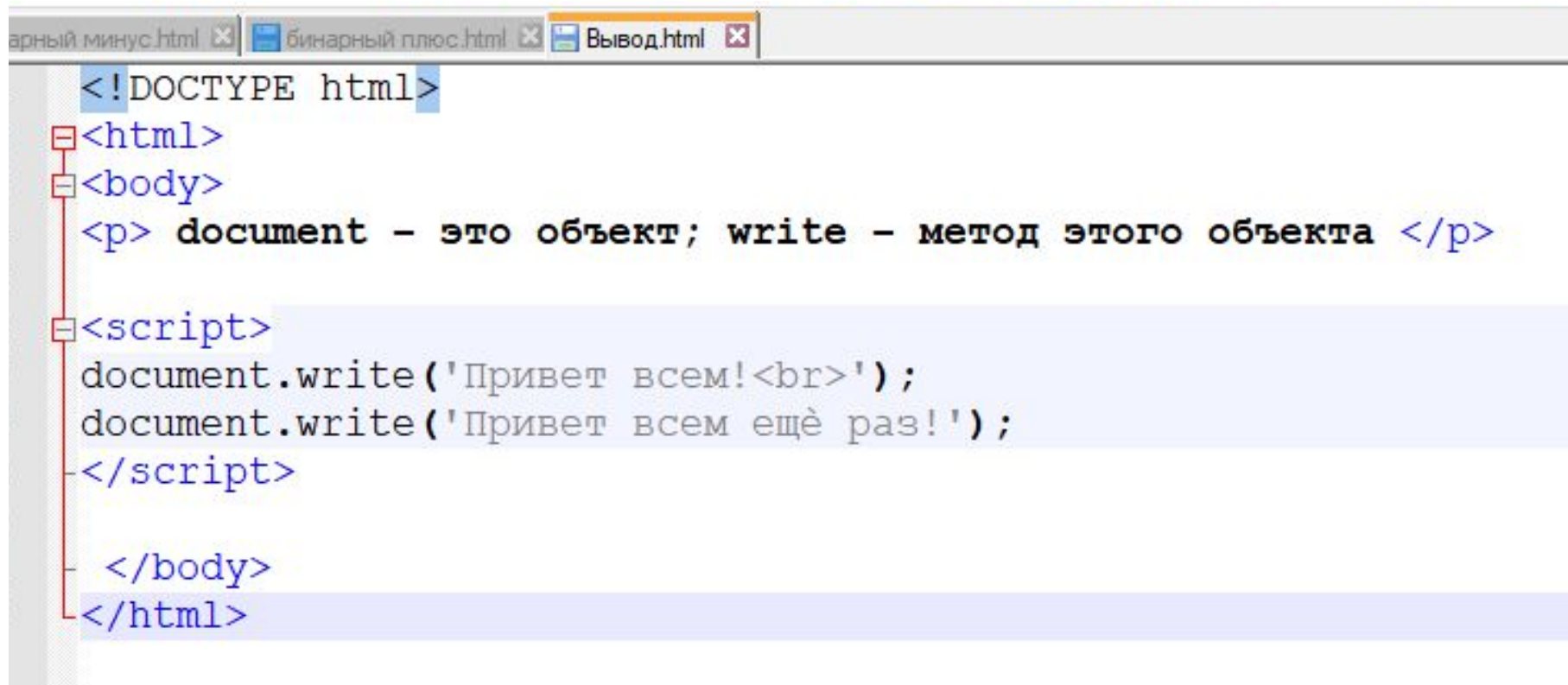
Основные операторы

Операнд

- ***Операнд*** – то, к чему применяется оператор.
- Например: $5 * 2$ – оператор умножения с левым и правым операндами. Другое название: «аргумент оператора».

Процедура document.write

Процедура document.write вставляет информацию именно в документ, а не в окно браузера



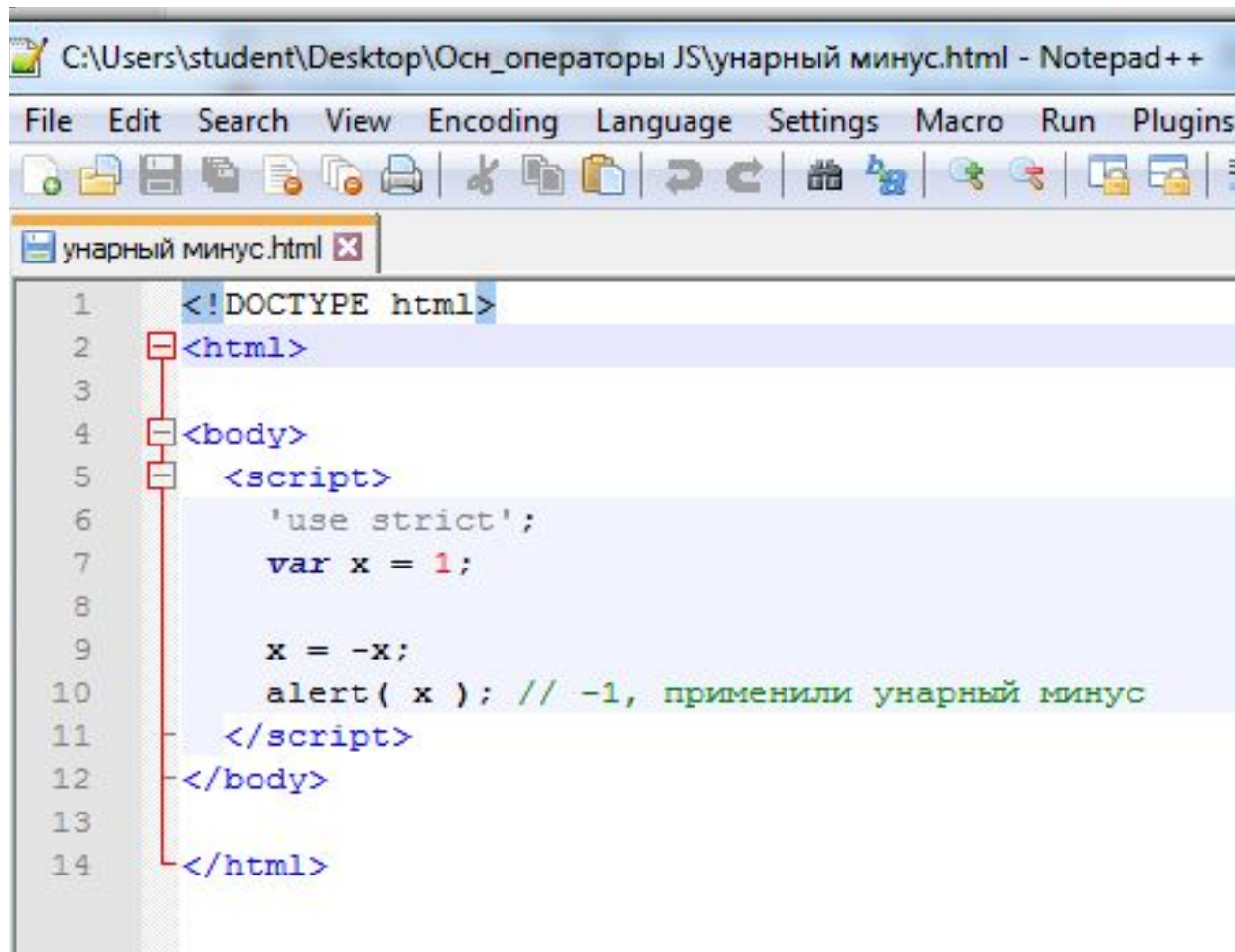
The screenshot shows a web browser window with three tabs: 'арный минус.html', 'бинарный плюс.html', and 'Вывод.html'. The 'Вывод.html' tab is active. The browser's content area displays the following HTML code:

```
<!DOCTYPE html>
<html>
<body>
  <p> document - это объект; write - метод этого объекта </p>
  <script>
    document.write('Привет всем!<br>');
    document.write('Привет всем ещё раз!');
  </script>
</body>
</html>
```

The code is displayed with syntax highlighting: HTML tags are in blue, the paragraph text is in black, and the JavaScript code is in black. The paragraph text is rendered in the browser window, and the JavaScript code is executed, resulting in two lines of text being displayed below the paragraph.

Пример1. **Унарный** оператор

- **Унарным** называется оператор, который применяется к одному выражению. Например, оператор унарный минус "-" меняет знак числа на противоположный:

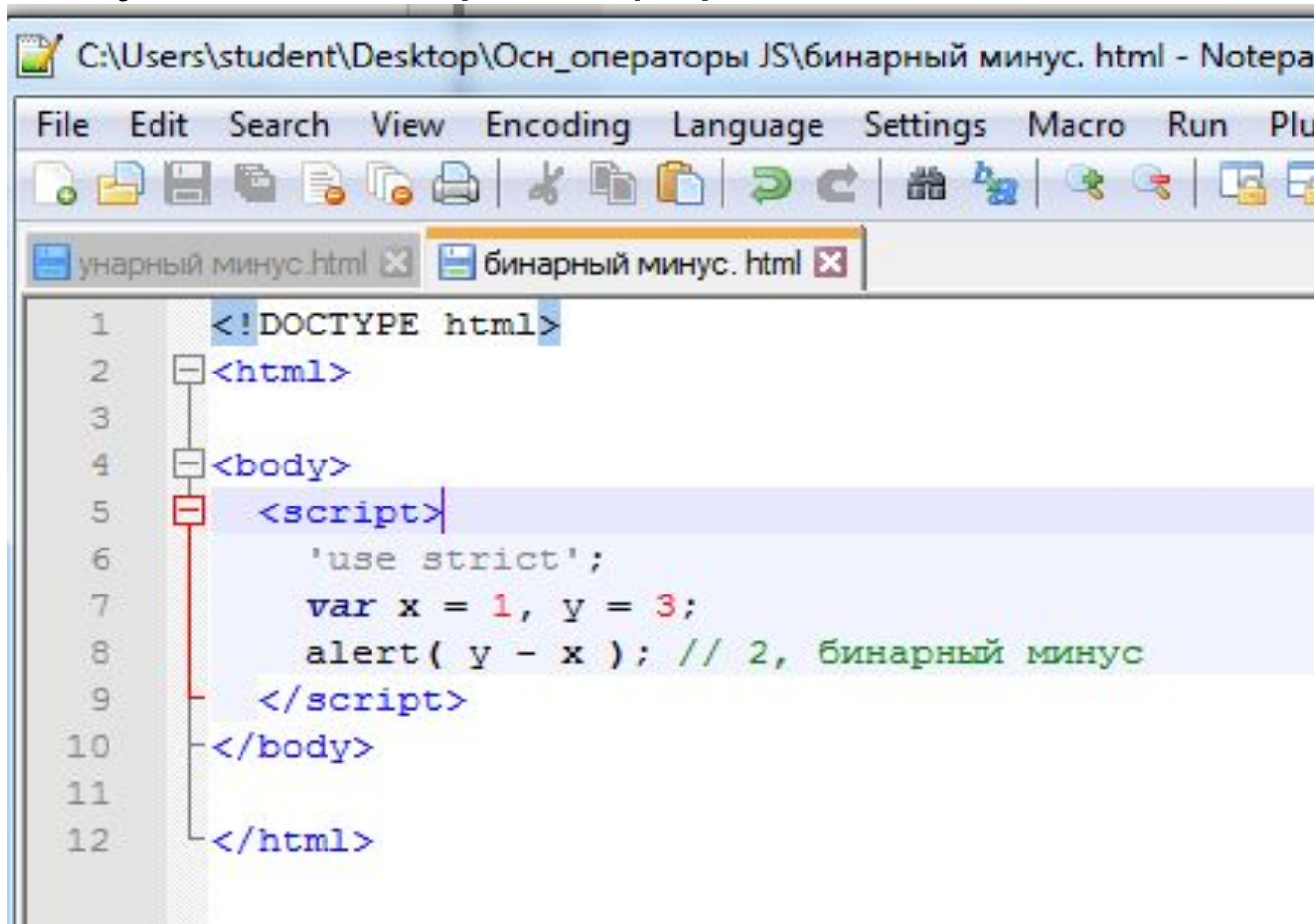


The screenshot shows a Notepad++ window titled "C:\Users\student\Desktop\Осн_операторы JS\унарный минус.html - Notepad++". The menu bar includes File, Edit, Search, View, Encoding, Language, Settings, Macro, Run, and Plugins. The toolbar contains various icons for file operations and editing. The active tab is "унарный минус.html". The code is as follows:

```
1  <!DOCTYPE html>
2  <html>
3
4  <body>
5    <script>
6      'use strict';
7      var x = 1;
8
9      x = -x;
10     alert( x ); // -1, применили унарный минус
11   </script>
12 </body>
13
14 </html>
```

Пример2. **Бинарный** оператор

- **Бинарным** называется оператор, который применяется к двум операндам. Тот же минус существует и в бинарной форме:



```
1 <!DOCTYPE html>
2 <html>
3
4 <body>
5 <script>
6     'use strict';
7     var x = 1, y = 3;
8     alert( y - x ); // 2, бинарный минус
9 </script>
10 </body>
11
12 </html>
```

Сложение строк. Бинарный

+

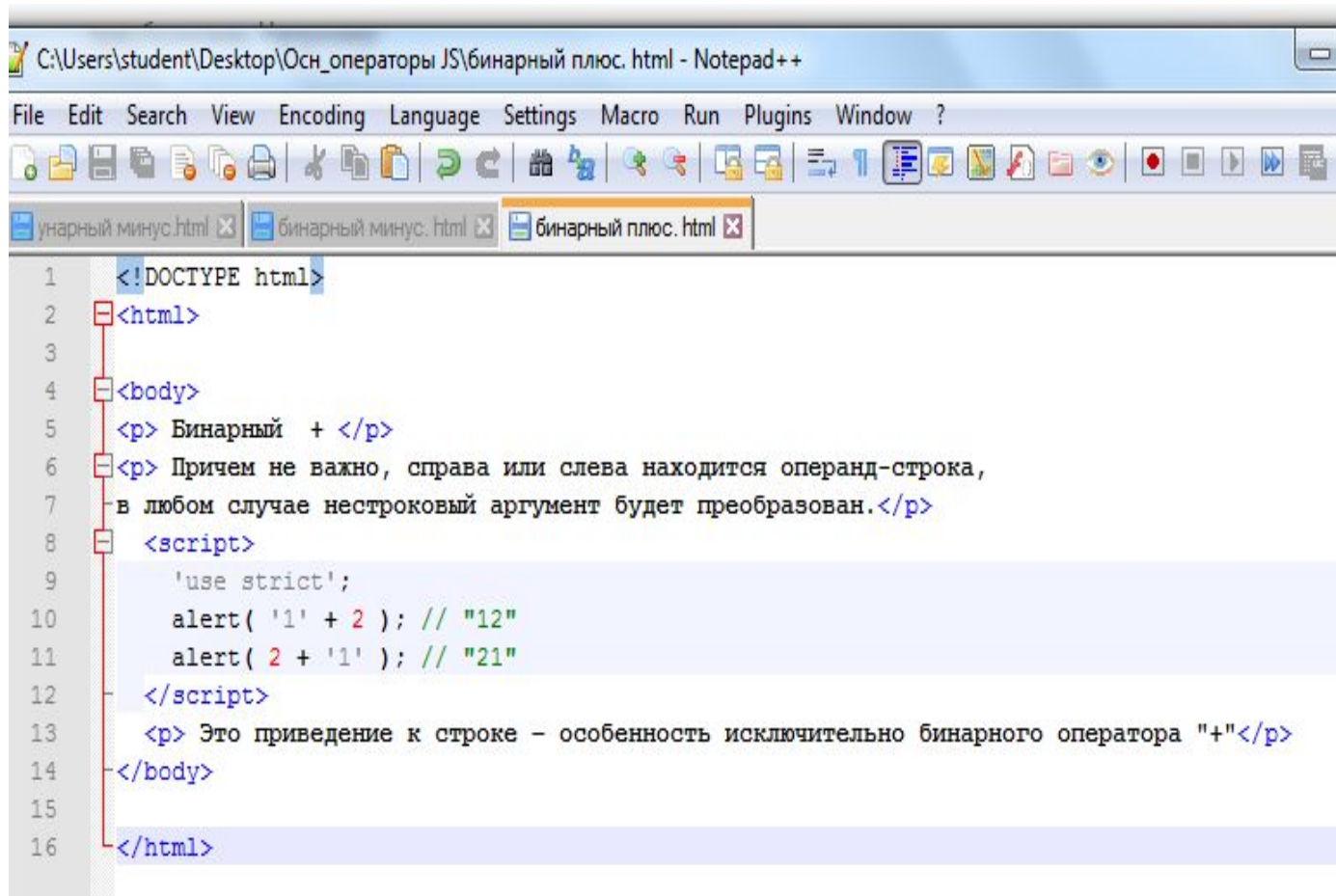
Если бинарный оператор '+' применить к строкам, то он их объединяет в одну:

```
var a = "моя" + "строка";  
alert( a ); // моястрока
```

Если хотя бы один аргумент является строкой, то второй будет также преобразован к строке!

Пример 3

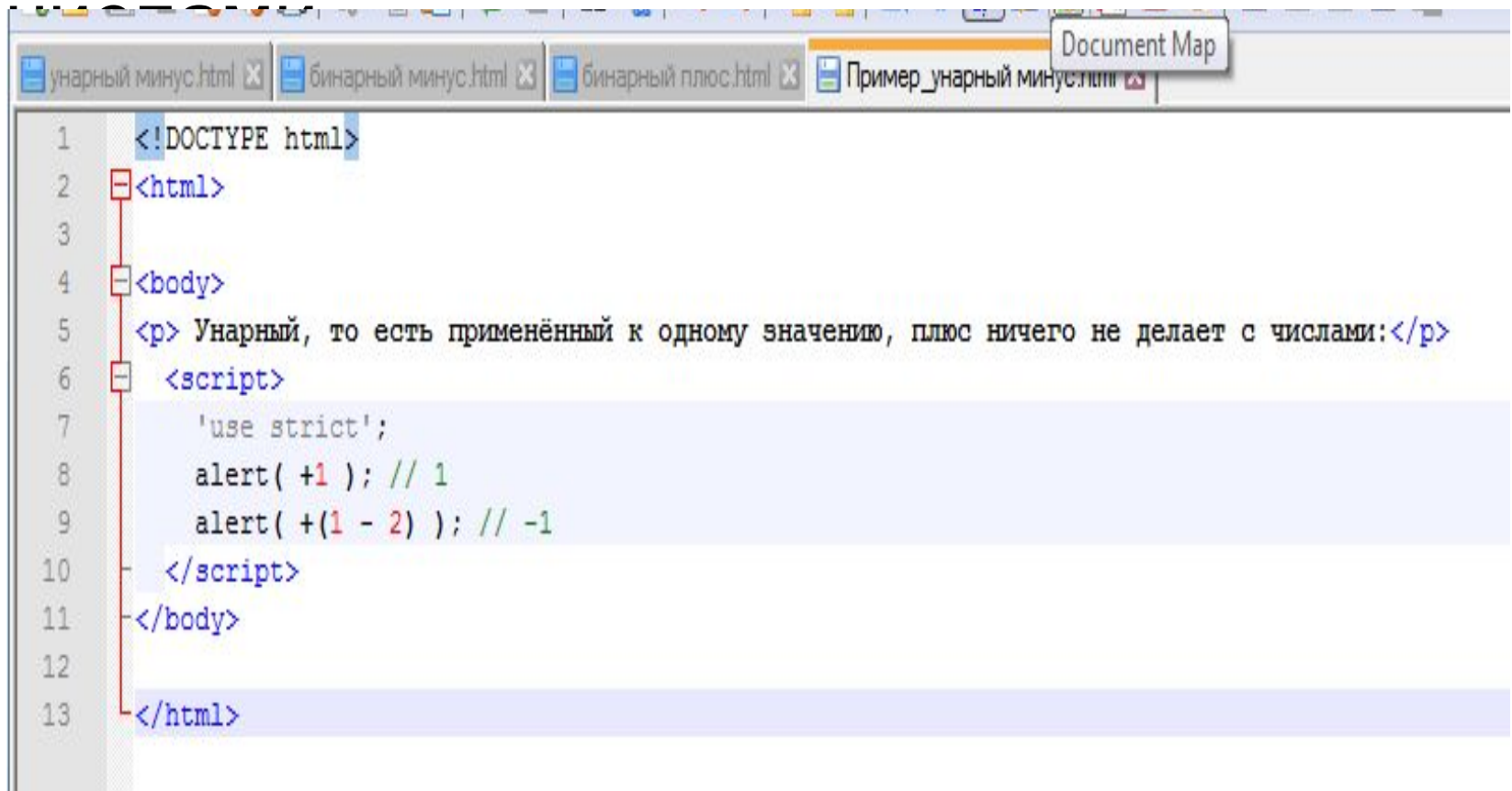
- Не важно, справа или слева находится операнд-строка, в любом случае нестроковый аргумент будет преобразован.



```
1 <!DOCTYPE html>
2 <html>
3
4 <body>
5   <p> Бинарный + </p>
6   <p> Причем не важно, справа или слева находится операнд-строка,
7   в любом случае нестроковый аргумент будет преобразован.</p>
8   <script>
9     'use strict';
10    alert( '1' + 2 ); // "12"
11    alert( 2 + '1' ); // "21"
12  </script>
13   <p> Это приведение к строке – особенность исключительно бинарного оператора "+"</p>
14 </body>
15
16 </html>
```

Преобразование к числу, унарный плюс +

- Унарный, то есть применённый к одному значению, плюс ничего не делает с



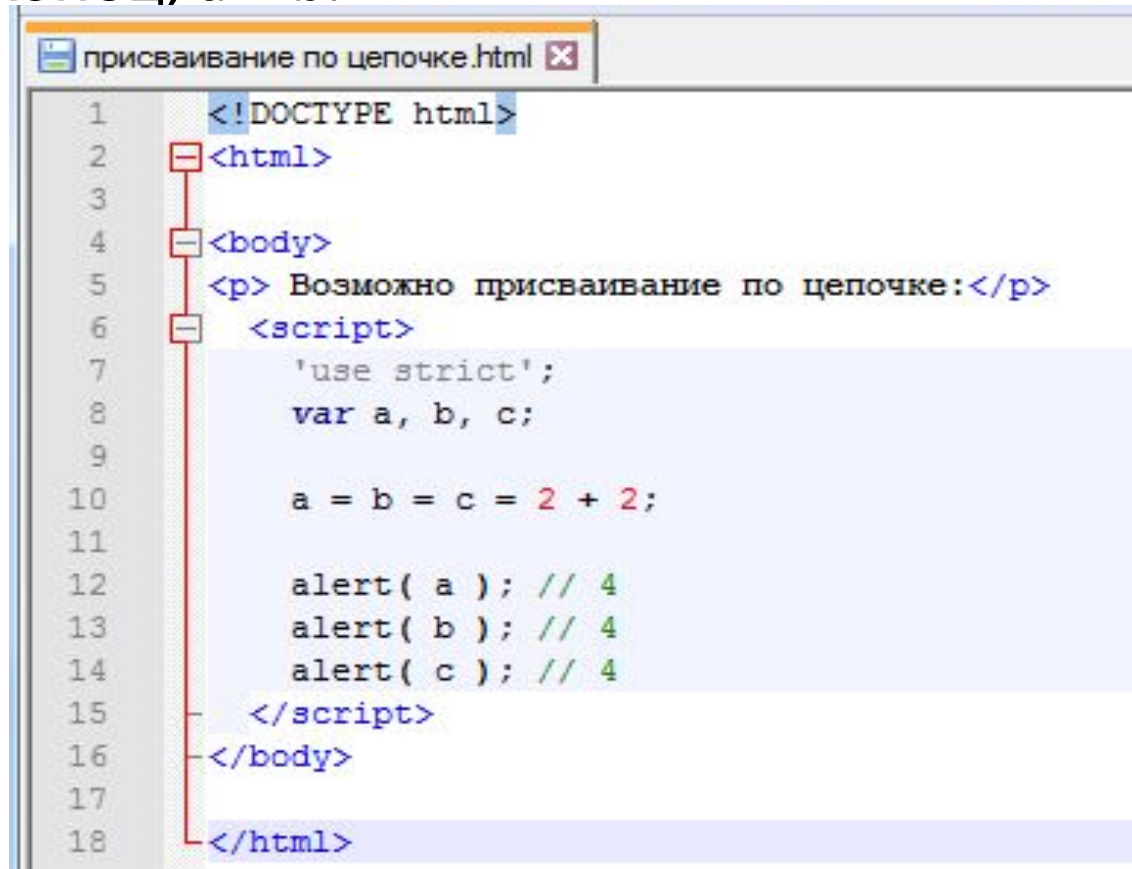
The screenshot shows a web browser window with four tabs: 'унарный минус.html', 'бинарный минус.html', 'бинарный плюс.html', and 'Пример_унарный минус.html'. The active tab is 'Пример_унарный минус.html'. The browser's address bar is empty. The page content is an HTML document with the following code:

```
1 <!DOCTYPE html>
2 <html>
3
4 <body>
5   <p> Унарный, то есть применённый к одному значению, плюс ничего не делает с числами:</p>
6   <script>
7     'use strict';
8     alert( +1 ); // 1
9     alert( +(1 - 2) ); // -1
10  </script>
11 </body>
12
13 </html>
```

The code is displayed in a light blue background with line numbers on the left. The script section is highlighted with a light blue background. The browser's Document Map is visible on the right side of the window.

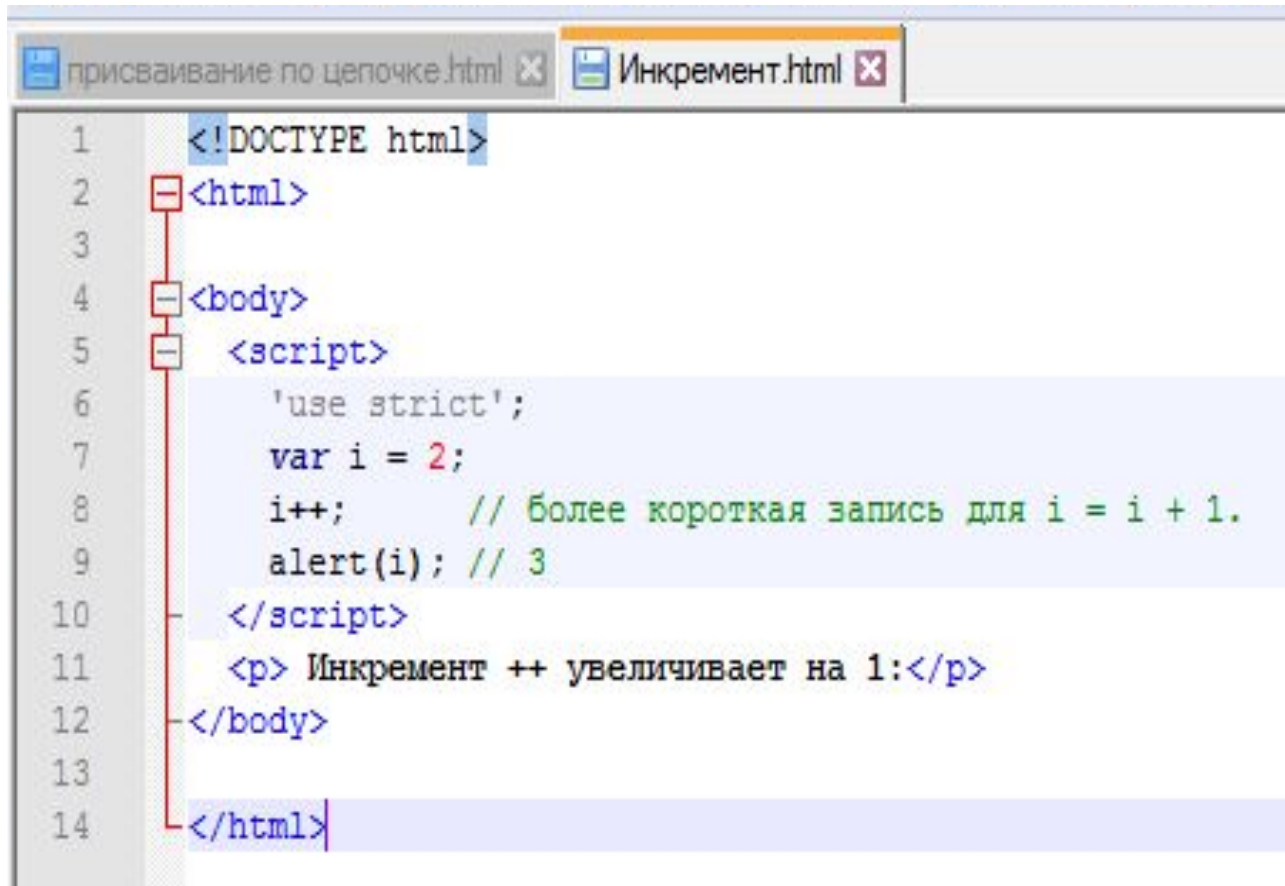
Пример4. Возможно присваивание по цепочке:

- Такое присваивание работает справа-налево, то есть сначала вычисляются самое правое выражение $2+2$, присвоится в c , затем выполнится $b = c$ и, наконец, $a = b$.



```
1 <!DOCTYPE html>
2 <html>
3
4 <body>
5 <p> Возможно присваивание по цепочке:</p>
6 <script>
7     'use strict';
8     var a, b, c;
9
10    a = b = c = 2 + 2;
11
12    alert( a ); // 4
13    alert( b ); // 4
14    alert( c ); // 4
15 </script>
16 </body>
17
18 </html>
```

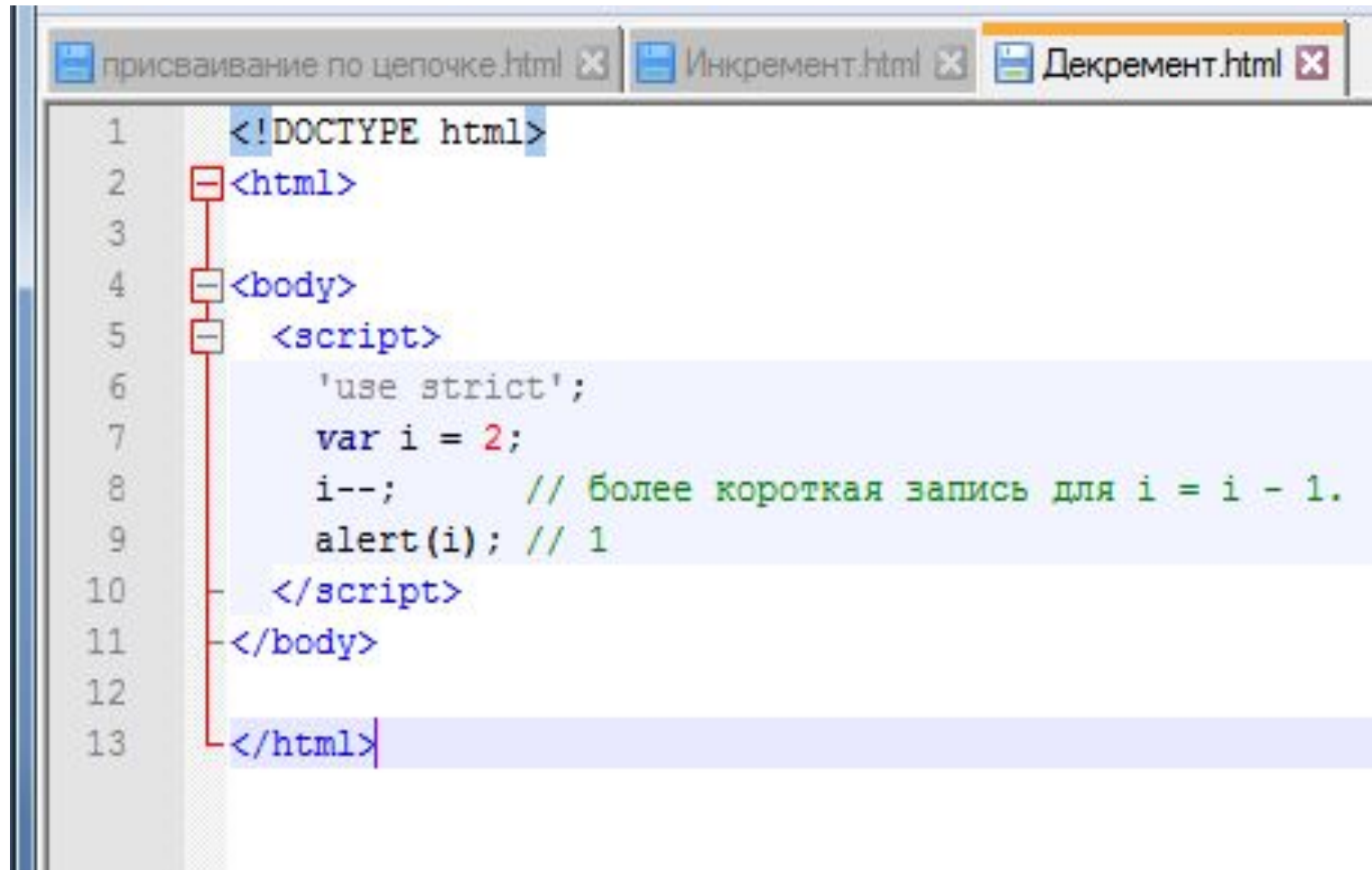
Пример5. **Инкремент ++** увеличивает на 1



```
1 <!DOCTYPE html>
2 <html>
3
4 <body>
5   <script>
6     'use strict';
7     var i = 2;
8     i++;      // более короткая запись для i = i + 1.
9     alert(i); // 3
10  </script>
11  <p> Инкремент ++ увеличивает на 1:</p>
12 </body>
13
14 </html>
```

- Одной из наиболее частых операций в JavaScript, как и во многих других языках программирования, является увеличение или уменьшение переменной на единицу.

Декремент -- уменьшает на 1:



```
1 <!DOCTYPE html>
2 <html>
3
4 <body>
5 <script>
6     'use strict';
7     var i = 2;
8     i--;      // более короткая запись для i = i - 1.
9     alert(i); // 1
10 </script>
11 </body>
12
13 </html>
```

Сокращённая арифметика с присваиванием

```
1 var n = 2;  
2 n += 5; // теперь n=7 (работает как n = n + 5)  
3 n *= 2; // теперь n=14 (работает как n = n * 2)  
4  
5 alert( n ); // 14
```

Задания

1. Преобразуйте код в программу

```
1 var apples = "2";  
2 var oranges = "3";  
3  
4 alert( apples + oranges ); // "23", так как бинарный плюс складывает строки
```

```
1 var apples = "2";  
2 var oranges = "3";  
3  
4 alert( +apples + +oranges ); // 5, число, оба операнда предварительно преобразованы в числа
```

2. Напишите код программы

а) "3" + 4 + 5 //результат –

б) 3 + 4 + «5» //результат –

Задания

3. Напишите код программы.

Значения переменным произвольно присвоить в коде программы. Также вывести на экран сам пример в том виде, как он выглядит в коде программы. Примечание: на данном этапе для возведения в степень можете переменную умножить саму на себя нужное количество раз.

$$65A^2 + 43B + C; \frac{A^3 + B^2 - 56}{C}.$$

4. Создайте программу с переменной `nim` и присвойте ей значение 5. Выведите на экран это значение. Прибавить к переменной `nim` 5 и вывести результат. Удвоить переменную и вывести результат. Выполнить деление на 10 и вывести результат. Выполнить вычитание числа 2 и вывести результат.

5. Создайте программу вычисления площади прямоугольного треугольника, ввод данных сделать с клавиатуры

6. Создайте программу вычисления площади трапеции, ввод данных сделать с клавиатуры