РОБОТА ЗІ РЯДКОВИМИ ФУНКЦІЯМИ

План:

1. Основи роботи зі рядковими функціями.
2. Метод toUpperCase
3. Метод toLowerCase
4. Метод substr
5. Метод substring
6. Метод slice
7. Метод indexO
8. Метод lastIndexOf
9. Метод replace
10. Метод split

Основи роботи зі рядковими функціями.

Властивість length дозволяє дізнатися, скільки символів міститься в рядку. Наприклад, якщо ми маємо рядок "Hello World!", то його довжина складається з 12 символів (включаючи пробіли).

Щоб дізнатися довжину рядка, потрібно використати властивість length. Наприклад, якщо у нас є змінна str зі значенням "abcdefg", ми можемо дізнатися її довжину, використовуючи наступний код:

```
var str = 'abcdefg';
console.log(str.length); // виведе 7
```

Тут ми використали метод length, який повертає кількість символів в рядку str. Результатом виконання цього коду буде виведення числа 7 в консоль.

Метод toUpperCase

Метод toUpperCase дозволяє перетворити рядок на верхній регістр, тобто змінити всі маленькі літери на великі. При цьому повертається новий рядок, а початковий рядок залишається без змін.

Щоб використати метод toUpperCase, потрібно викликати його від змінної, яка містить рядок, який потрібно перетворити.

Синтаксис: рядок.toUpperCase();

Наприклад, нехай маємо рядок str = "javascript+html".

Щоб перетворити його на верхній регістр, потрібно викликати метод toUpperCase від змінної str: str.toUpperCase().

Результат буде записаний в новий рядок, тому можна використовувати змінну для збереження результату або виводити результат безпосередньо у вікно браузера за допомогою команди document.write().

Приклад:

```
var str = "javascript+html";
var newStr = str.toUpperCase();
document.write(newStr); // Виведе: JAVASCRIPT+HTML
```

Також можна використовувати змінну, що містить початковий рядок, і зберігати результат в іншу змінну. При цьому початковий рядок залишиться без змін.

Приклад:

```
var str = "JavaScript";
var newStr = str.toUpperCase();
document.write('Новий рядок: ' + newStr + '<br>');
document.write('Початковий рядок: ' + str);
// Виведе:
// Новий рядок: JAVASCRIPT
// Початковий рядок: JavaScript
```

Метод toLowerCase

Метод toLowerCase преобразовує рядок до нижнього регістру (з великих букв робить маленькі). Результатом виконання методу є новий рядок, а початковий рядок не змінюється.

Синтаксис: рядок.toLowerCase();

Наприклад:

```
var str = "Мова JAVASCRIPT";
var newStr = str.toLowerCase();
console.log(newStr); // мова javascript
console.log(str); // Мова JAVASCRIPT
```

Або коротший варіант без збереження результату у змінну:

```
var str = "Мова JAVASCRIPT";
console.log(str.toLowerCase()); // мова javascript
console.log(str); // Мова JAVASCRIPT
```

Метод substr

Метод substr дозволяє отримати частину рядка (підрядок) без зміни початкового рядка.

Щоб скористатися методом substr, необхідно передати два параметри. *Перший параметр* – це позиція, з якої починається вирізання, а *другий параметр* – це кількість символів, які необхідно вирізати.

Якщо другий параметр не передано, будуть вирізані всі символи починаючи з позиції, переданої в першому параметрі.

Якщо передати в першому параметрі від'ємне значення, то позиція буде відраховуватися від кінця рядка. Останній символ рядка має номер -1, передостанній - номер -2, і так далі.

Приклад 1: Вирізання підрядка довжиною 4 символи з початку рядка:

```
var str = 'Мені дуже подобається JavaScript';
document.write(str.substr(0, 4));
```

Результат виконання коду: Мені

Приклад 2: Вирізання підрядка з 5-го символа до кінця рядка:

```
var str = 'Мені дуже подобається JavaScript';
document.write(str.substr(5));
```

Результат виконання коду: дуже подобається JavaScript

Приклад 3: Вирізання підрядка з 10-го символу з кінця і повернення 4 символи:

```
var str = 'Мені дуже подобається JavaScript';
document.write(str.substr(-10, 4));
```

Результат виконання коду: Java

Приклад 4: Вирізання підрядка з 10-го символу з кінця і повернення всієї решти рядка:

```
var str = 'Мені дуже подобається JavaScript';
document.write(str.substr(-10));
```

Результат виконання коду: JavaScript

Приклад 5: Повернення останнього символу рядка:

```
var str = 'Мені дуже подобається JavaScript';
document.write(str.substr(-1));
```

Результат виконання коду: t

Метод substring

Метод substring дозволяє вирізати підрядок з рядка без зміни вихідного рядка. Щоб використати цей метод, потрібно вказати два параметри: номер символу, з якого починається вирізання (починаючи з 0) та номер символу, на якому закінчується вирізання (не включаючи його в результат).

Якщо другий параметр не вказаний, то будуть вирізані всі символи після першого параметра. Якщо перший параметр більший за другий, то substring поводиться так, ніби вони помінялися місцями.

Якщо якийсь з параметрів більше за довжину рядка, то він вважається рівним довжині рядка. Негативні значення прирівнюються до нуля.

Отже, можна використовувати substring для вирізання підрядка з рядка за його номерами символів.

Наприклад, якщо ми маємо рядок "*Мені дуже подобається JavaScript*", то можна використати substring для вирізання "*Мені дуже*" або *"дуже"*, а також можна вирізати підрядок з 5-тої позиції і до кінця рядка.

Ось кілька прикладів коду на JavaScript з використанням substring:

```
// Вирізаються слова "Мені дуже".
var str = 'Мені дуже подобається JavaScript';
document.write(str.substring(0, 9));

// Вирізається слово "дуже".
var str = 'Мені дуже подобається JavaScript';
document.write(str.substring(5, 9));

// Вирізається підрядок з 5-тої позиції і до кінця рядка.
var str = 'Мені дуже подобається JavaScript';
document.write(str.substr(4));
```

Метод slice

Метод slice допомагає вирізати частину рядка без зміни початкового рядка. Щоб використати цей метод, необхідно вказати номер символу, з якого почнеться вирізання та номер символу, на якому закінчиться вирізання.

Якщо другий параметр не вказано, вирізання буде здійснено від зазначеного в першому параметрі символу до кінця рядка. Якщо другий параметр має від'ємне значення, вирізання почнеться з кінця рядка.

Наприклад, якщо потрібно вирізати слово "*дуже*" з рядка "*Мені дуже подобається JavaScript*", можна використовувати метод slice (4, 9), оскільки перша літера слова "дуже" має номер 4, а остання - номер 8. Однак, щоб вирізати і останній символ, другий параметр можна не вказувати, і тоді вирізання здійсниться до кінця рядка.

У метода slice також є альтернативний синтаксис з використанням методу substr, який працює за аналогічним принципом, але відрізає підрядок з одного параметру до іншого, без можливості використовувати від'ємні значення. Ось кілька прикладів використання методу slice та substr:

Приклад 1: Вирізання підрядка "дуже" з рядка "Мені дуже подобається JavaScript"

```
var str = 'Мені дуже подобається JavaScript';
document.write(str.slice(4, 9));
```

Результат: дуже

Приклад 2: Вирізання підрядка з 5-тої позиції і до кінця рядка:

```
var str = 'Мені дуже подобається JavaScript';
document.write(str.slice(4));
```

Результат: дуже подобається JavaScript

Приклад 3: Виведення символів від 0 до позиції -1 не включно:

```
var str = '12345';
document.write(str.slice(0, -1));
```

Результат: 1234

Метод indexO

Метод indexOf використовується для пошуку підрядка у рядку. Він поверне позицію першого збігу підрядка в рядку, або -1, якщо такого збігу не знайдено.

Також можна вказати другий параметр, який вказує, з якої позиції слід починати пошук.

Метод чутливий до регістру символів, тобто він розрізняє великі і малі букви.

Ось приклади використання методу indexOf:

Приклад 1: У рядку 'Я вчу вчу Javascript' шукається слово 'вчу'.

Метод поверне 2, що є позицією першого збігу підрядка у рядку:

```
var str = 'Я вчу вчу Javascript';
document.write(str.indexOf('вчу'));
```

Результат: 2

Приклад 2: У рядку 'Я вчу вчу Javascript' також шукається слово 'вчу', але пошук починається з 5-ї позиції.

Метод поверне 6, що є позицією другого збігу підрядка у рядку:

```
var str = 'Я вчу вчу Javascript';
document.write(str.indexOf('вчу', 5));
```

Результат: 6

Приклад 3: У рядку 'Я вчу Javascript' шукається підрядок 'PHP', якого немає. Метод поверне -1:

```
var str = 'Я вчу Javascript';
document.write(str.indexOf('PHP', 5));
```

Результат: -1

Приклад 4: У рядку 'Я вчу Javascript' шукається підрядок 'JaVaScrIpt', але метод поверне -1, так як він чутливий до регістру символів:

```
var str = 'Я вчу Javascript';
document.write(str.indexOf('JaVaScrIpt'));
```

Результат: -1

Приклад 5: У рядку 'Я вчу Javascript' шукається підрядок 'вчу', починаючи з 8-ї позиції, але збігу підрядка після обраної позиції немає.

Метод поверне -1:

```
var str = 'Я вчу Javascript';
document.write(str.indexOf('вчу', 8));
```

Результат: -1

Метод lastIndexOf

Метод lastIndexOf шукає підрядок в рядку і повертає позицію першого збігу, рахуючи з кінця рядка. Якщо збіг не знайдено, повертається -1.

Синтаксис: рядок.lastIndexOf(що шукаємо, [звідки починати пошук]);

Цей метод чутливий до регістру символів, тому, якщо ви шукаєте підрядок в нижньому регістрі, але в рядку є тільки верхній регістр, метод не знайде збіг.

Приклад: у рядку 'Б..Б..Б' шукається літера 'Б'. Метод поверне 6, тому що це позиція останньої літери 'Б' в рядку.

```
var str = 'Б..Б..Б';
document.write(str.lastIndexOf('Б'));
```

Результат виконання коду: 6

Можна передати другим параметром номер символу, звідки слід починати пошук. Наприклад, у рядку 'Б..Б..Б' знайдемо літеру 'Б', починаючи з п'ятої позиції (це передостанній символ рядка). Метод поверне 3, тому що це позиція першої літери 'Б', починаючи з 5-ої позиції.

```
var str = 'Б..Б..Б';
document.write(str.lastIndexOf('Б', 5));
```

Результат виконання коду: 3

Якщо метод не знайшов збіг, то він поверне -1. Наприклад, у рядку 'Б..Б..Б' шукається літера 'б'. Метод поверне -1, тому що в рядку немає маленької літери 'б', а тільки великі.

```
var str = 'Б..Б..Б';
document.write(str.lastIndexOf('б'));
```

Результат виконання коду: -1

Метод replace

Метод replace дозволяє замінити одну частину рядка іншою. Щоб замінити частину рядка, потрібно викликати метод replace() на рядку, який містить цю частину. Першим параметром метод приймає те, що потрібно замінити, а другим параметром – на що замінити.

Наприклад, у рядку "Я вчу PHP" можна замінити слово "PHP" на слово "JavaScript", використовуючи наступний код:

```
var str = 'Я вчу PHP';
var newStr = str.replace('PHP', 'JavaScript');
document.write(newStr);
```

Як результат, буде виведено рядок "Я вчу JavaScript".

Якщо в рядку є декілька входжень того, що потрібно замінити, метод replace() замінить тільки перше входження.

Наприклад, у рядку "Я вчу PHP PHP" перше слово "PHP" буде замінено на слово "JavaScript", а друге слово "PHP" залишиться незмінним:

```
var str = 'Я вчу PHP PHP';
var newStr = str.replace('PHP', 'JavaScript');
document.write(newStr);
```

Як результат, буде виведено рядок "Я вчу JavaScript PHP".

Щоб замінити всі входження того, що потрібно замінити, необхідно використовувати регулярні вирази та глобальний пошук.

Наприклад, у рядку "*Я вчу PHP PHP*" обидві входження слова "*PHP*" можна замінити на слово "*JavaScript*", використовуючи наступний код:

```
var str = 'Я вчу PHP PHP';
```

```
var newStr = str.replace(/PHP/g, 'JavaScript');
document.write(newStr);
```

Як результат, буде виведено рядок "Я вчу JavaScript JavaScript".

Метод split

Метод split використовується для розбиття рядка на масив елементів за заданим розділювачем. Розділювач вказується як перший необов'язковий параметр. Якщо розділювач не заданий – буде повернений весь рядок. Якщо розділювач заданий як порожні лапки "" – то кожен символ рядка буде записаний в окремий елемент масиву.

Другим необов'язковим параметром можна вказати максимальну кількість елементів у отриманому масиві.

Ось декілька прикладів використання методу split:

```
// Приклад 1: Рядок розбитий по роздільнику "-"
// В результаті вийде масив ['Мені', 'дуже', 'подобається', 'JavaScript']
var str = 'Мені-дуже-подобається-JavaScript';
var arr = str.split('-');
console.log(arr);
// Приклад 2: Рядок розбитий по роздільнику "-"
// Другим параметром вказано максимальну кількість елементів в отриманому масиві (2 елементи).
```

```
// Тому в новий масив запишеться тільки 2 елементи: ['Мені', 'дуже']
var str = 'Мені-дуже-подобається-JavaScript';
var arr = str.split('-', 2);
console.log(arr);
// Приклад 3: Кожен символ рядка записується в окремий елемент масиву
// В результаті вийде масив ['a', 'b', 'c', 'd', 'e']
var str = 'abcde';
var arr = str.split('');
console.log(arr);
// Приклад 4: Перші 3 символи рядка записуються в окремий елемент масиву
// В результаті вийде масив ['a', 'b', 'c']
var str = 'abcde';
var arr = str.split('', 3);
console.log(arr);
// Приклад 5: Рядок з числами розбитий на масив рядків
// В результаті вийде масив ['1', '2', '3', '4', '5'], а не масив чисел [1, 2, 3, 4, 5]
var str = '12345';
var arr = str.split('');
console.log(arr);
```