**ინტერნეტ დაპროგრამება. ლექცია №8. ციკლები**

 for; for ... in

ციკლები (iteration)

ციკლები გამოიყენება განსაზღვრული მოქმედებების განმეორებით შესრულების­თვის - რამდენიმეჯერ ან მანამ, სანამ არ შესრულდება გარკვეული პირობა. არსებობს ციკლის ორგანიზების ოთხი ოპერატორი:

* for ოპერატორი;
* for ... in ოპერატორი;
* do ... while ოპერატორი;
* while ოპერატორი;

for ოპერატორი

JavaScript –ში ყველაზე ხშირად გამოყენებადი ციკლური სტრუქტურა არის for. მას აქვს შემდეგი სინტაქსისი:

for ([საწყისი მნიშვნელობა]; [პირობა]; [ცვლილების ბიჯი])

[{]ციკლის შიგნით შესასრულებელი ოპერატორები

[}]

კვადრატული ფრჩხილები მიუთითებს იმას, რომ მათში ჩაწერილი გამოსახულებების შეტანა არ არის აუცილებელი, ანუ ის შეძლება არც იყოს..

ოპერატორ for -ს უწოდებენ ასევე ციკლის ოპერატორს მთვლელით, თუმცა შეიძლე­ბა მოხდეს ისე, რომ მთვლელი მასში არც იყოს.

ის რაც მოთავსებულია მრგვალ ფრჩხილებში, მოსამსახურე სიტყვა for -ის შემდეგ წარმოადგენს ციკლის ოპერატორის *სათაურს*; ხოლო იმას, რაც ციკლის ოპერატორის სათაურის შემდეგაა, როგორც წესი, ფიგურულ ფრჩხილებში, უწოდებებ ციკლის ტანს.

 ციკლის ოპერატორის *სათაური* - ([საწყისი მნიშვნელობა]; [პირობა]; [ბიჯის ცვლილება])

საწყისი მნიშვნელობა- განსაზღვრავს ციკლის მთვლელის საწყის მნიშვნელობას, და იგი შესრულდება მხოლოდ ერთხელ, როცა დაიწყება ციკლოს ოპერატორის მოქმე­დება; საწყისი მნიშვნელობა ჩვეულებრივ მოიცემა მინიჭების ოპერატორის სახით, მაგალითად j=3 ან var j=3 . ცვლადის იდენტიფიკატორი, სახელი შეიძლება იყოს ნებისმიერი, ხოლო ცვლადის მნიშვნელობა არის მთელი რიცხვი. ამ ცვლადს ეწოდება ციკლის მთვლელი.

პირობა- იგულისხმება ისეთივე გამოსახულება როგორც if ოპერატორში. იგი წარმო­ადგენს ციკლის მუშაობის გაგრძელების პირობას. მაგალითად j<nUdidesi .

ბიჯის ცვლილება - წარმოადგენს გამოსახულებას, ოპერატორს, რომელიც სრულდე­ბა ყოველჯერზე ციკლის შიგნით შესასრულებელი ოპერატორების - ციკლის ტანის შეს­რულების შემდეგ. მაგალითად j=J+1 ან j++ .

ციკლის ოპერატორი მუშაობს შემდეგნაირად:

1.თავდაპირველად შესრულდება საწ­ყისი პირობა, ისიც მხოლოდ ერთხელ;

2. შემოწმდება პირობა, თუ კი პირობა არ სრულდება (მცდარია - fasle), ხდება მეოთხე პუნქტზე გადასვლა - ციკლის ოპერატორი ასრულებს მუშაობას, წინააღმდეგ შემ­თხვევაში - ანუ, როცა პირობა სრულდება (ჭეშმარიტია - true), დაიწყება ციკლის ტანის შეს­რულება.

3. როცა დასრულდება ციკლის ტანი დაიწყებს შესრულებას ბიჯის ცვლილე­ბა და მოხდება გადასვლა მეორე პუნქტზე.

4. ციკლის ოპერატორის მუშაობის დასრულება.

ცხადია, რომ თუ მეორე პუნქტში ყოველთვის გვექნება პირობის შესრულება - ჭეშმარიტია - true, ციკლის ტანის ყოცელთვის შესრულდება - მოხდება ”ჩაციკვლა”. ”ჩაციკვლა” არასასურვრლია, რამდენადაც ალგორითმისთვის არსებითი თვისებაა ამოცანის გადაწყვეტა სასრული ბიჯის პირობებში.

აღსანიშნავია, რომ ციკლის ტანი შესდგება ერთი ოპერატორისგან, მაშინ ფიგურული ფჩხილების გამოყენება სავალდებულო არაა.

ოპერატორ for -ის მაგალითი

ლისტინგი 1. ოპერატორ for

მაგალითი იმისა, რომ შეიძლება ციკლის ტანი იყოს ფიგურული ფჩხილების გარეშე და ბიჯის ცვლილე­ბის შესრულება მაშინაც კი, როცა ციკლის ტანი ბოლოჯერ შესრულდა

ლისტინგი 2. ციკლის ტანი ფიგურული ფჩხილების გარეშე

<html>

 <head>

 <title> ............</title>

<script language="javascript">

<!-- ვინიღბებით!

for ( i = 1; i < 10; i ++ ) document.write ("გამოვიყვანოთ i= "+ i, "<BR>"); document.write ("გამოვიყვანოთ i= "+ i, "<BR>");

// ვიხსნით ნიღაბს. -->

</script>

</head>

 <body>

</body>

</html>

მაგალითები იმისა, როცა ციკლის სათაურში არაა რომელიმე კომპონენტი:

ლისტინგი 3\_1. არაა საწყისი მნიშვნელობა. საწყისი მნიშვნელობა გატანილია ციკლის თავში,

<html>

 <head>

 <title> ციკლი for</title>

<script language="javascript">

<!-- ვინიღბებით!

i=1;

for ( ; i<2 ; i ++ ) {

document.write ("გამოვიყვანოთ i= "+ i, "<BR>");

}

// ვიხსნით ნიღაბს. -->

</script>

</head>

 <body>

</body>

</html>

ლისტინგი 3\_2. არაა საწყისი მნიშვნელობა და პირობა. საწყისი მნიშვნელობა გატანილია ციკლის თავში, პირობის არარსებობა იწვევს ჩაციკვლას

<html>

 <head>

 <title> ციკლი for</title>

<script language="javascript">

<!-- ვინიღბებით!

i=1;

for ( ; ; i ++ ) {

document.write ("გამოვიყვანოთ i= "+ i, "<BR>");

}

// ვიხსნით ნიღაბს. -->

</script>

</head>

 <body>

</body>

</html>

განვიხილოთ არაუარყოფითი მთელი რიცხვის ფაქტო­რიალის გამოთვლის მაგალითი.

ვიცით, რომ 0!=1; 1!=1

ლისტინგი 4. ფაქტორიალი

<html>

 <head>

 <title> ფაქტორიალი, ციკლის პირობა ტანში</title>

<script language="javascript">

<!-- ვინიღბებით!

var z = 1 ;

n=prompt('შეიყვანეთ არაუარყოფითი მთელი რიცხვი','n');

if (n >1) {

for (i = 2 ; i<=n; i++) z=z\*i

 }

document.write('არაუარყოფითი მთელი რიცხვის ',n,'!='+z)

// ვიხსნით ნიღაბს. -->

</script>

</head>

 <body>

</body>

</html>

for (х = 1; x > 0; х+ + ) ასეთი ციკლი იწვევს ”ჩაციკვლა” - პირობა ყოველთვის იქნება ჭეშმარიტი.

როცა ციკლის სათაურში არ არის პირობა და ბიჯის ცვლილება, იმისათვის რომ ციკლმა კორექტულად იმუშაოს, მაშინ მის ტანში უნდა იყოს როგორც პირობა, ასევე ბიჯის ცვლილება.

განვიხილოთ შემთხვევა, როცა ციკლის სათაურში არ არის პირობა და ის ციკლის ტანშია მოთავსებული. ეს სიტუაცია ავხნათ 100 ნატურალური რიცხვის მონიტორზე გამოტანის მაგალითზე. პირობა if (i>100) break; ჩაწერილია ტანში და მასში გამოყენებულია ოპერატორი break , რომელიც იწვევს ციკლის მუშაობის შეწყვეტას.

ლისტინგი 5. პირობა ტანშია break

<html>

 <head>

 <title> 100 ნატურალური რიცხვის გამოტანა, ციკლის პირობა ტანში</title>

<script language="javascript">

<!-- ვინიღბებით!

var i=1;

for (; ; i++) {

 if (i>100) break;

 document.write('ნატურალური ',i,"<br>");

}

// ვიხსნით ნიღაბს. -->

</script>

</head>

 <body>

</body>

</html>

ეხლა განვიხილოთ ის შემთხვევა, როცა ბიჯის ცვლილებაც ტანშია, ისევ 100 ნატურალური რიცხვის გამოტანის მაგალითზე

ლისტინგი 5\_1 პირობა და ბიჯის ცვლილება ტანშია, break ოპერატორი

<html>

 <head>

 <title> 100 ნატურალური რიცხვის გამოტანა, ციკლის პირობა ტანში</title>

<script language="javascript">

<!-- ვინიღბებით!

var i=0;

for (; ; ) { i++;

 if (i>100) break;

 document.write('ნატურალური ',i,"<br>");

}

// ვიხსნით ნიღაბს. -->

</script>

</head>

 <body>

</body>

</html>

ოპერატორი continue იძლევა საშუალებას გადავიდეთ ციკლის შემდეგ იტერაციაზე, გამოტოვებს რა ტანის ოპერატორებს. ვანახოთ მისი გამოყენება ასი ნატურალური რიცხვიდან მხოლოდ იმათი გამოტანის მაგალითზე, რომელნიც [5,10] მონაკვეთს არ ეკუთვნიან:

ლისტინგი 5.2 ოპერატორი continue

<html>

 <head>

 <title> 100 ნატურალური რიცხვის გამოტანა, ოპერატორი continiu </title>

<script language="javascript">

<!-- ვინიღბებით!

for (var i=1; i<101; i++) {

 if (i>4 && i<11) continue;

 document.write(i + "<br>");

}

// ვიხსნით ნიღაბს. -->

</script>

</head>

 <body>

</body>

</html>

for ... in ოპერატორი , ციკლის ორგანიზაციის ეს ოპერატორი გამოიყენება მასივის ელემენტებზე ერთი ამ რამდენიმე ოპერატორის შესასრილებლად. ანუ ჩავწერათ რა ციკლის ტანში ერთ ბრძანებას, იგი იმოქმედებს მასივის ყველა ელემენტზე, და აღარაა საჭირო ეს ბრძანება ცალ-ცალკე ჩაიწეროს მასივის თითოეულ ელემენტისთვის.

განვიხილოთ მასივის ელემენტების ერთი ერთეულით გაზრდის მაგალითი:

ლისტინგი 6. for ... in ოპერატორი

<html>

 <head>

 <title> ×for...in ოპერატორი </title>

<script language="javascript">

<!-- ვინიღბებით!

var myarray=new Array(5);

myarray[0]=5;

myarray[1]=8;

myarray[2]=10;

myarray[3]=18;

myarray[4]=180;

for (x in myarray)

{

myarray[x] = ++myarray[x];

}

alert(myarray);

// ვიხსნით ნიღაბს. -->

</script>

</head>

 <body>

</body>

</html>

კიდევ ერთი მაგალითი, როცა მასივის ელემენტები ივსება სტრიქონული ცვლადებით

ლისტინგი 6.1. for ... in ოპერატორი

<html>

 <head>

 <title> ×for...in ოპერატორი </title>

<script language="javascript">

<!-- ვინიღბებით!

var myarray=new Array("","","","","","","","","","")

for (x in myarray)

{

myarray[x] = "img" + x + ".gif";

}

alert(myarray);

// ვიხსნით ნიღაბს. -->

</script>

</head>

 <body>

</body>

</html>