NaN → code on invalid wrong data types 100% chances.

Page No. 22/11/23
Date Wed

→ priority.

Precedence :- ( ), ., [ ], ++ --, - - (pre)
** , * , / , %
+ , -
> < >= <=
== != === !==.
++ -- (post)
=

Eg - <body>
<h2> Demo on precedene of operator </h2>
<script>
document.write (10+20*5-160%30, "
");
document.write (10-20/4+10*5, "
");
let x=15, y=6;
document.write (x+y); // 15+6 =21.
invalid// document.write ("Sum is" + x+y); //Sum is 156
or
document.write ("Sum is" + (x+y)); //Sum is 21
or
document.write ("Sum is", x+y); //Sum is 21
or
invalid // document.write ("sub is" + x-y); // NAN (not a numbers)
or
valid ~~If Nan~~ document.write ("sub is" + (x-y); // bracket laga dege to operation shi se execute hoga. because of precedance ++ got confuse.

Similarly.

document.write ("product is" + x*y) (+, -)
valid. // product is 90.
becj precedence goes to * than +.

means it is not in real string like "ramkam" ← it will be like int in string quote like "10".

Parsing:

⇒ Changing the data from string format to no format.

Types of parsing :-

1) implicit parsing.
2) explicit parsing.

1) Implicit parsing :-

⇒ System converting data into number format.

⇒ Eg :- Substraction's, product's & division's

- Substraction's ⇓ All type of subtn -, --, -=, ;
- product's ⇓ All types of multn *, **, etc..
- division's ⇓ All types of division. /, %,

2) Explicit parsing :-

⇒ manual conversion.

i) using fun".
ii) using operators.

i) using function :-

a) ParseInt()
- Converting data from string format to no (int) format.

Synt :- number.ParseInt("value")

Eg :- "10" → 10
Eg :- "10.56" → 10 → it will not come. only int format will come.
Eg :- "ram" → Nan

b) ParseFloat():-

converting data from String format to floating format.

Synt:- window.parseFloat("value")

Eg:- "10" → 10

Eg:- 10.56" → 10.56 // accepting the decimal values, because of that.

Eg:- "Ram" → NaN.

ii) using operators:-

→ + is a parse operator

→ uranay operator, it should use left side value/var (prefix)

Synt:- +variable OR +value

Eg:- +"10" → 10

Eg:- +"10.56" → 10.56

Eg:- +"Ram" → NaN.

Eg:-

```
<body>
<h2>Demo on Parsing</h2>
<script>
    let x = "50", y = "9";
    document.write(x+y, "<br>"); // no parse.
    document.write(x-y, '<br>'); // auto parse.
    document.write(x*y, '<br>'); // auto parse.
    document.write(x/y, '<br>'); // auto parse.
    document.write(x>y, '<br>'); // no parse.
```

Every input value comes as a string only.

let result = window.parseInt(x) + window.parseInt(y);
//manual passing or explicit passing.

document.write(result, '
');
document.write(parseInt(x) > parseInt(y), '
');
result = "25" + "45" //concetination → "2545".

If we want to add both we have to use parseInt() for both so, that it will change into addition operator.

document.write(parseInt(x) + parseInt(y));
result = "25" + + "45";
document.write(result, '
');
document.write(10 * "45", '
'); //auto parse.
document.write(+"100" + 45, '
');

+ will concetinate so, use + operator before string "100".

Eg: without using html tag.

```
<!DOCTYPE html>
<h3> Finding sum of two number </h3>
<Script>
    let x, y, res;
    x = Parseprompt ("enter 1st no");
    y = parseprompt (" " 2nd ".");
    res = x+y;
    document.write('Addition is' + res);
</Script>
</html>
```

Instead of parse we can directly use plus (+) before prompt so, that it will add the 2 values - not concetn.