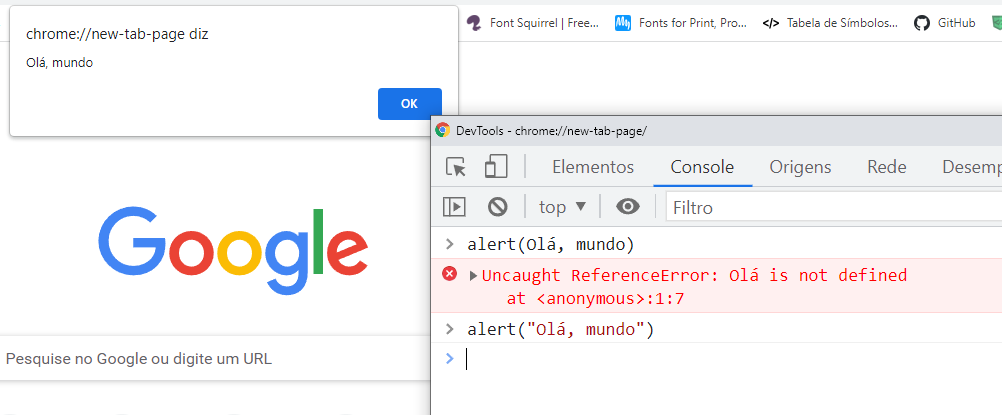
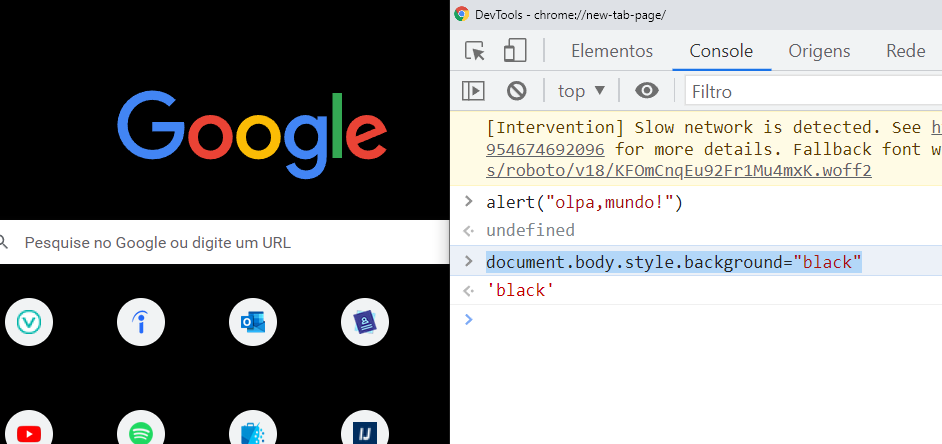
**Modulo A – Conhecendo o JavaScript**

Podemos programar mensagens de exibição no google Chrome usando a ferramenta **inspecionar** > **Devtools** > **Console,** inserimos a tag **alert (“**mensagem**”)** e damos **enter, exemplo abaixo:**



Usando essa declaração **document.body.style.background="black",** podemos modificar a cor de fundo da tela inicial do google chrome, **exemplo abaixo:**



Podemos inserir mensagens de alerta usando a tag **script** onde antes inserimos a tag **window.alert (‘mensagem’);** **exemplo abaixo:**

<body>

    <h1>Olá, Mundo!</h1><br>

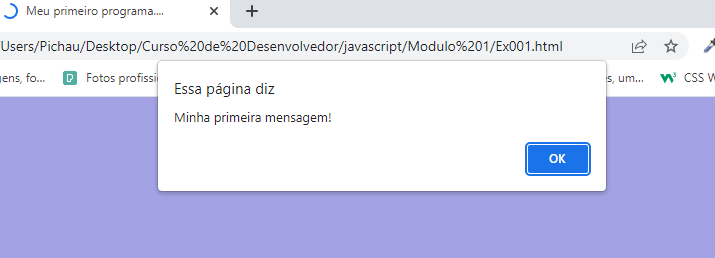
    <p>Já me livrei da maldição</p>

    <script>

        window.alert('Minha primeira mensagem!');

    </script>

</body>



Podemos inserir uma pergunta de confirmação para o usuário usando a tag **confirm** onde antes inserimos a **window.confirm(‘mensagem’);** **exemplo abaixo:**

<body>

    <h1>Olá, Mundo!</h1><br>

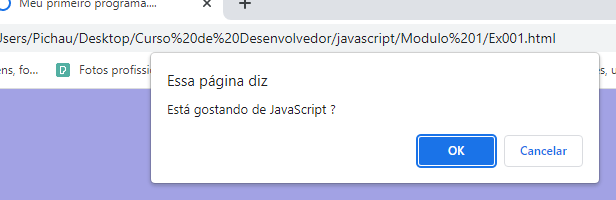
    <p>Já me livrei da maldição</p>

    <script>

        window.confirm('está gostando de JavaScript ?');

    </script>

</body>



Podemos fazer uma pergunta ao usuário usando a tag **prompt** onde antes inserimos **window.prompt(‘mensagem’); exemplo abaixo:**

<body>

    <h1>Olá, Mundo!</h1><br>

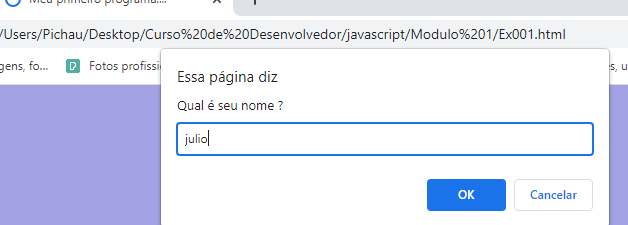
    <p>Já me livrei da maldição</p>

    <script>

        window.prompt('Qual é seu nome ?');

    </script>

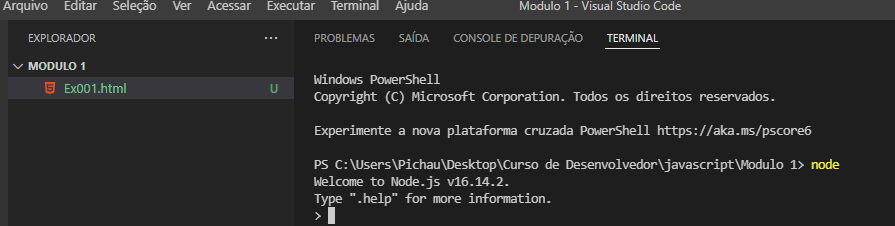
</body>



Para inserirmos comentários de até uma linha dentro do **Javascript** usamos a tag **//** e para inserirmos comentários com mais de 1 linha inserimos /\* \*/, lembrando que **/\* \*/** serve para o **CSS**, já para o **HMTL** usamos **<!—**

Para inserirmos variáveis usamos as tag **var** ou **let**, dentro do Javascript o sinal de **=** significa recebe.

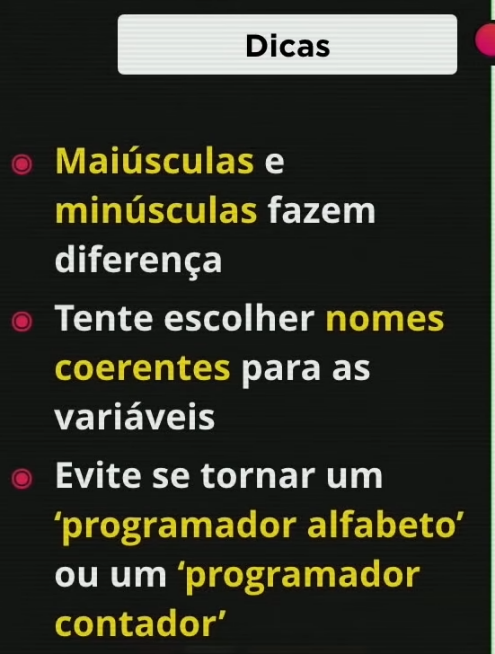
Podemos acessar o terminal dentro do **VScode** clicando em **terminal** > **novo terminal**, já dentro do terminal digitamos a palavra **node**, isso fará com que possamos abrir a tela do node dentro do **VScode**, **exemplo abaixo:**



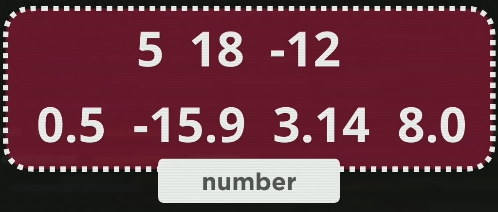
**MODULO B - Dados**



Abaixo estão algumas características que os **identificadores** em **Javascript** precisam e podem ter:

Abaixo temos 3 tipos de dados **primordiais (data types – tipos de dados)**:

Usamos a tag **window.alert**(‘mensagem, ‘ + resposta do usuário), abaixo temos um exemplo de **string de concatenação:**

<body>

    <h1>Olá, Mundo!</h1><br>

    <p>Já me livrei da maldição</p>

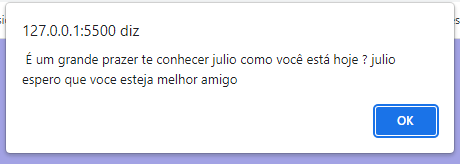
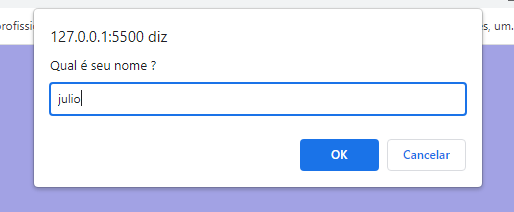
    <script>

       var nome = window.prompt('Qual é seu nome ?'); //vai ´perguntar o nome..

       window.alert(' É um grande prazer te conhecer ' +nome+ ' como você está hoje ? ' +nome+ ' espero que você esteja melhor amigo ') // usamos o sinal de + para juntar a resposta que o usuário vai nos fornecer, isso precedente com ''

    </script>

</body>



Abaixo temos um exemplo de **string template**:

<body>

    <h1>Olá, Mundo!</h1><br>

    <p>Já me livrei da maldição</p>

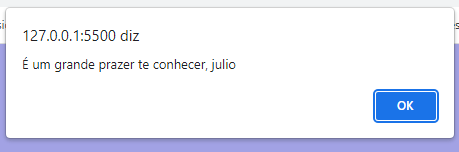
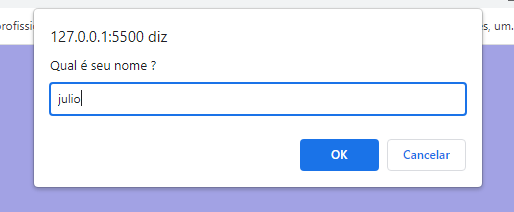
    <script>

       var nome = window.prompt('Qual é seu nome ?'); //vai ´perguntar o nome..

       window.alert(`É um grande prazer te conhecer, ${nome}`) // usamos o sinal de + para juntar a resposta que o usuário vai nos fornecer, isso precedente com ''

    </script>

</body>



Podemos converter **string** (nomes) para **numbers** (números) usando 2 tags, sendo elas:

**Number.parseInt(numero) –** usamos para ter retorno de números inteiros

**Number.parseFloat(numero) –** usamos para ter retorno de números quebrados

**Exemplo abaixo** usando **Number.parseInt – inteiros com string de concatenação**

<body>

    <h1>Olá, Mundo!</h1><br>

    <p>Já me livrei da maldição</p>

    <script>

        var n1 = Number.parseInt(window.prompt('Digite um número:')) //string

        var n2 = Number.parseInt(window.prompt('Digite outro número:')) //string

        var s = n1 + n2

        window.alert('A soma dos valores totais é ' + s)

        // (number+numer) server para adição

        // (string+string) serve para concatenação

    </script>

</body>

**Exemplo abaixo** usando **Number.parseFloat – números quebrados com string de concatenação**

<body>

    <h1>Olá, Mundo!</h1><br>

    <p>Já me livrei da maldição</p>

    <script>

        var n1 = Number.parseFloat(window.prompt('Digite um número:')) //string

        var n2 = Number.parseFloat(window.prompt('Digite outro número:')) //string

        var s = n1 + n2

        window.alert('A soma dos valores totais é ' + s)

        // (number+number) server para adição

        // (string+string) serve para concatenação

    </script>

</body>

Com a nova versão do **Javascript** usando a tag **Number(numero)** o próprio programa detecta se o **número** é **inteiro** ou **quebrado** e faz a soma**, exemplo abaixo com string de template:**

<body>

    <h1>Olá, Mundo!</h1><br>

    <p>Já me livrei da maldição</p>

    <script>

        var n1 = *Number*(window.prompt('Digite um número:')) //string

        var n2 = *Number*(window.prompt('Digite outro número:')) //string

        var s = n1 + n2

        window.alert(`A soma dos valores totais é ${s}`)

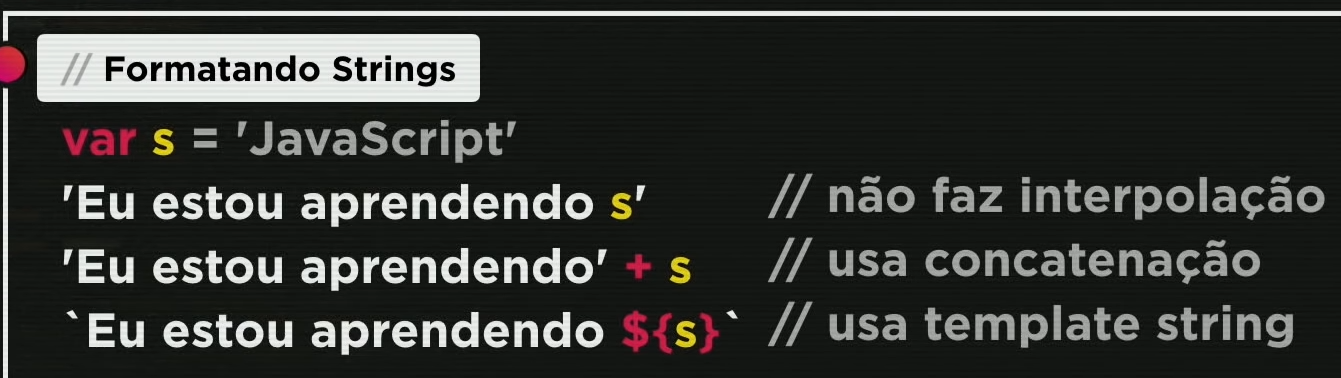
        // (number+number) server para adição

        // (string+string) serve para concatenação

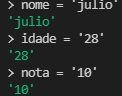
    </script>

</body>

**Usando strings a baixo:**

****

**Propriedades das variáveis:**

****

**Um exemplo de template string abaixo:**

****

**Um exemplo de string usando a concatenação (soma) abaixo:**

****

Abaixo um exemplo de declaração usando a **string template:**

<body>

    <h1>Olá, Mundo!</h1><br>

    <p>Já me livrei da maldição</p>

    <script>

        var n1 = *Number*(window.prompt('Digite um número:')) //string

        var n2 = *Number*(window.prompt('Digite outro número:')) //string

        var s = n1 + n2

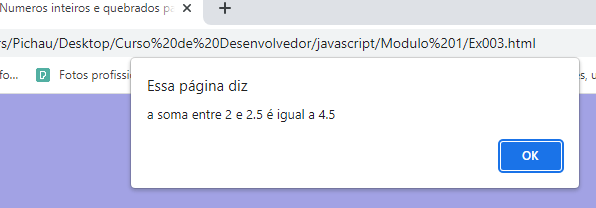
        window.alert(`A soma dos valores totais é ${s}`)

        // (number+number) server para adição

        // (string+string) serve para concatenação

    </script>

</body>

****

Abaixo um exemplo de declaração usando a **string de concatenação:**

<body>

    <h1>Olá, Mundo!</h1><br>

    <p>Já me livrei da maldição</p>

    <script>

        var n1 = *Number*(window.prompt('Digite um número:')) //string

        var n2 = *Number*(window.prompt('Digite outro número:')) //string

        var s = n1 + n2

        window.alert('A soma dos valores totais é ' + s)

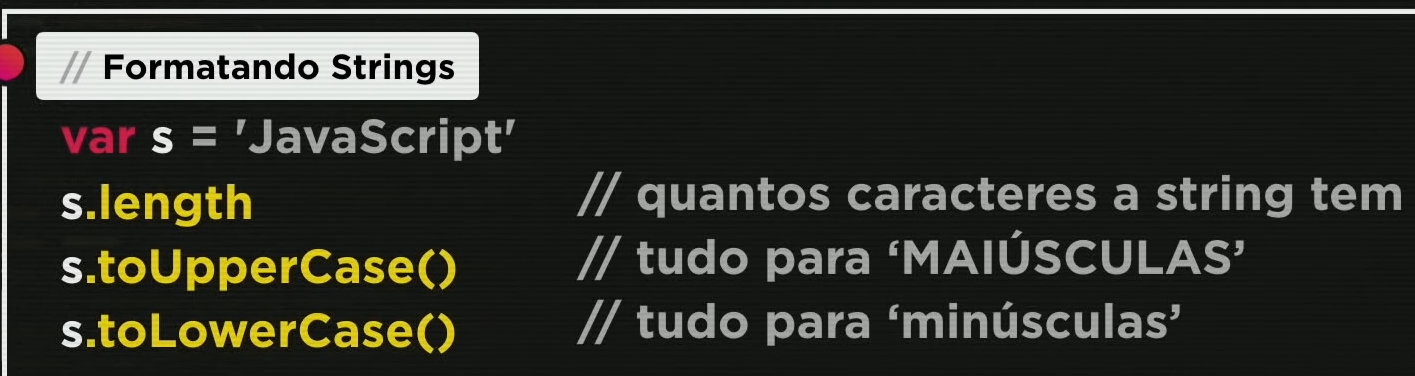
        // (number+number) server para adição

        // (string+string) serve para concatenação

    </script>

</body>

Abaixo mais variações de **strings** que podemos usar**:**



Podemos atribuir **nome à variável**, assim quando digitamos o cliente nos dar um nome a mesma vai retorna com a **resposta digitada** em texto dentro do site,além disso podemos **atribuir um estilo em CSS**, adicionarmos **parágrafos**, colocar a **fonte** em **negrito** ou colocar as **letras** **maiúsculas** ou **minúsculas**, **exemplo abaixo:**

    <style>

        body{

            font: normal 20pt arial;

        }

    </style>

</head>

<body>

    <script>

        var nome = window.prompt(`Qual e seu nome ?`)

        document.write(`Olá, <strong>${nome}!</strong> seu nome tem ${nome.length} letras <br>`)

        document.write(`Seu nome em maiusculas é ${nome.toUpperCase()}<br>`)

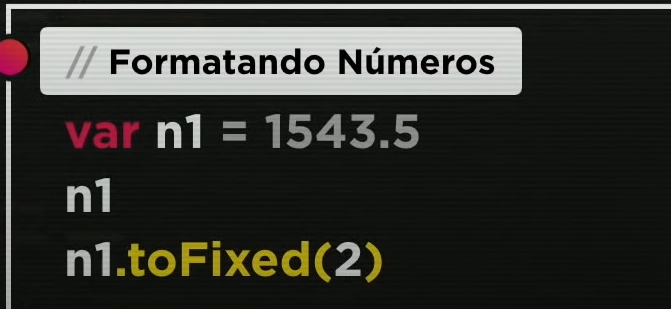
        document.write(`Seu nome em minusculas é ${nome.toLowerCase()}`)

    </script>

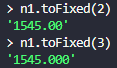
</body>

****

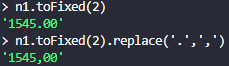
Podemos fazer a formatação de números seguindo as **strings** **abaixo:**



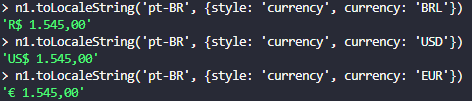
Com essa **string** estamos adicionando uma outra casa decimal usando o **toFixed(2)** em nossa **string**, podemos adicionar quantas casas decimais quisermos somente trocando o número 2.



Além disso podemos trocar a separação dos números de virgula por ponto ou vice e versa usando a declaração **toFixed(2).replace(‘.’,’,’)**

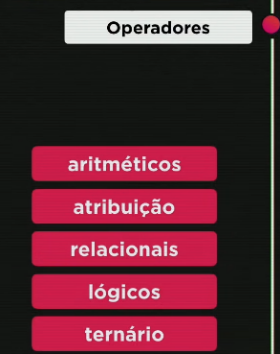


Além disso podemos colocar um estilo em nossa **string** como por exemplo o valor da moeda de cada país.



**Operadores em JS**

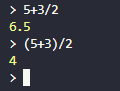
Abaixo temos alguns tipos de **operadores:**



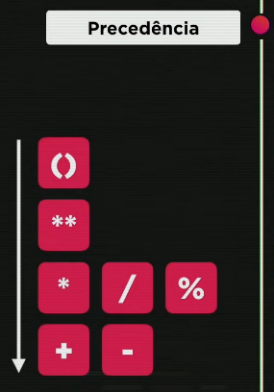
Iniciamos pelos operadores **aritméticos**, eles são usados para fazer **cálculos:**



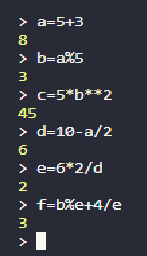
Devemos respeitar e estar atentos as formas de somar, multiplicar e dividir os números, pois existe uma lógica chamada **ordem de precedência:**



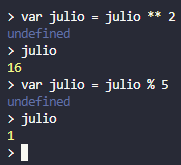
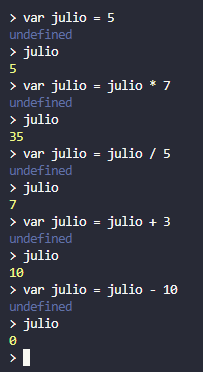
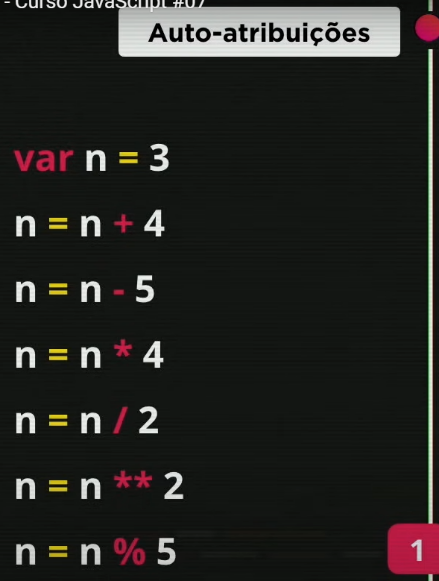
Abaixo está a ordem de **precedência**, o que significa quais logicas serão executas por **primeiro** e assim por sequência em **ordem decrescente**:



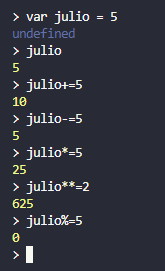
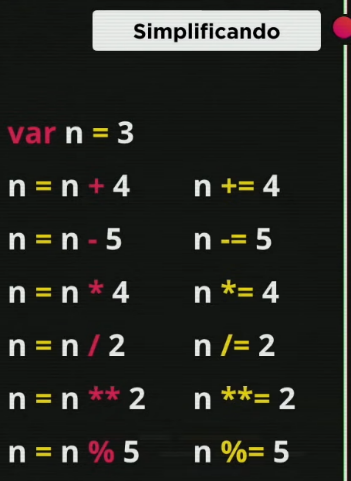
Abaixo algumas operações de **atribuições simples:**



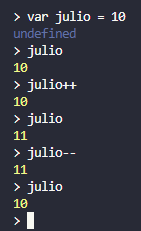
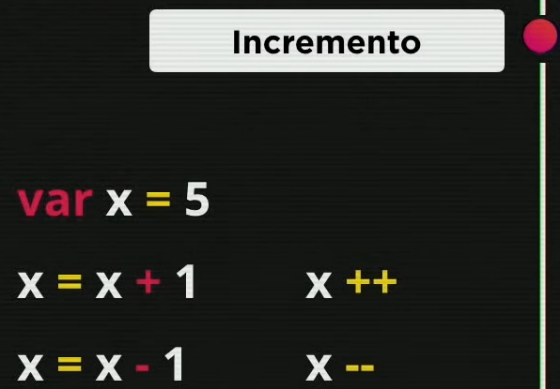
Também temos as **auto atribuições**, que são quando operações que podemos fazer com a **mesma variável**, mudando assim **seus valores:**



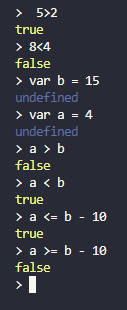
Temos uma maneira de **simplificar** estas **auto atribuições**, e estas **simplificações** são usadas por uma **grande variedade de linguagens:**

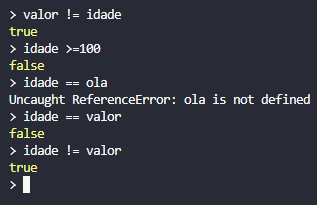
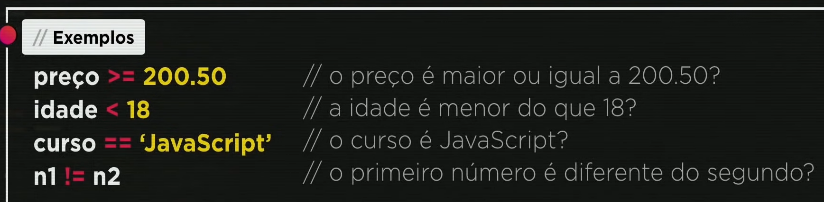


Também a uma outra forma chamada de **incremento** onde podemos realizar **somas** e **diminuições** atribuindo um valor para a **variável** e depois o nome da **variável** seguido de ++ ou -- :

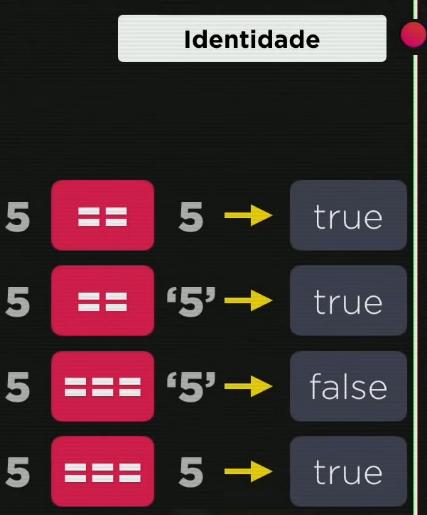
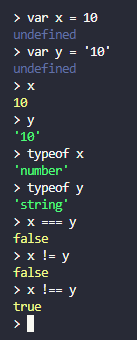


Abaixo veremos outros tipos de **operadores**, começando pelos **relacionais,** na operação dentro do **node.js** onde temos operadores **relacionais** e **aritméticos**, primeiro resolvemos os **aritméticos** e depois os **relacionais:**

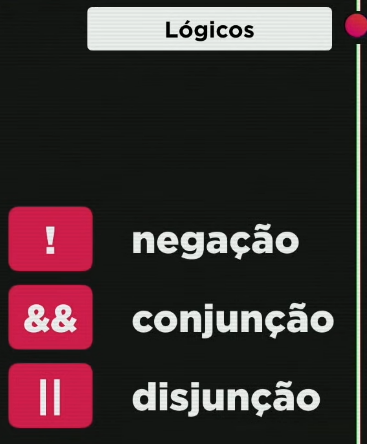


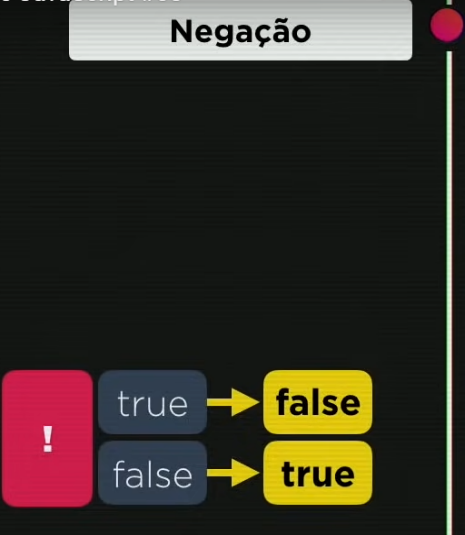
Podemos fazer outros testes lógicos, alguns **exemplos abaixo:**  


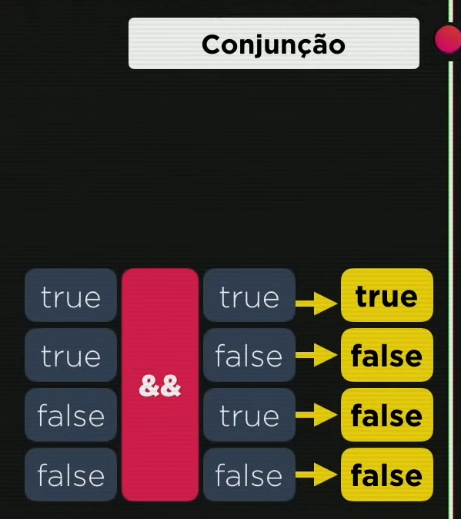
Outro exemplo de **operadores relacionais**, que chamamos de operadores de **identidade:**

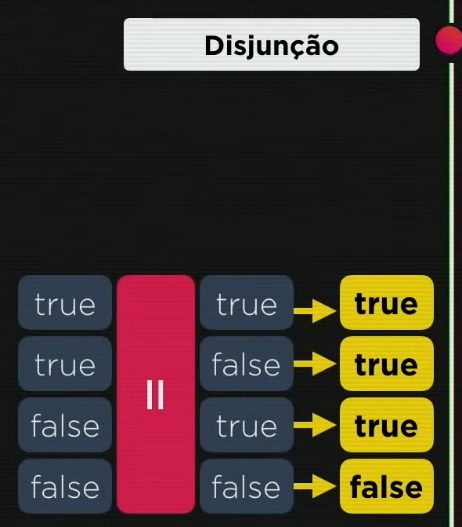


Abaixo temos alguns exemplos de operadores **lógicos:**

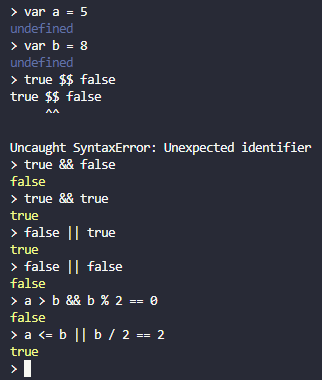


**Negação** – Informação que não é **verdadeira** o resultado e **false** e uma informação que não é **falsa** então será **true.**

**Conjunção** – retorna como **true** se a informação pedida for **totalmente encontrada**, caso não e voltado como **false**, só se satisfaz com as **2 informações verdadeiras**.

**Disjunção** – se um dos valores for **verdadeiro**, retorna como **verdadeiro**, só retorna como **falso** se os **2 valores forem falsos**.

Abaixo temos alguns exercícios usando os **3 operadores:**



No uso da **conjunção &&,** observamos que digitamos **true && true**, então o sistema nos retorna **true** pois os **2 valores digitados são iguais**.



Já no exemplo de cima o sistema nos retorna **false**, pois **true && false** são diferentes.



Já nos outros exemplos de baixo onde usamos a **conjunção ||** observamos que um dos valores sendo **true** ele nos retorna **true**,



já no exemplo de baixos os 2 valores são **false**, então o sistema nos retorna **false.**



Já abaixo temos temo uma **conjunção** onde resolvemos primeiro a parte **aritmética**, depois os **relacionais** e por último os **lógicos.** No caso abaixo o valor da direita será **true** e outro da esquerda **false** então nosso resultado será **false** pois na **conjunção** quando um dos valores for **false** e o outro valor ser **true** o resultado será **false.**



Já abaixo temos uma **disjunção** onde o valor da direita da **false** e o valor da esquerda da **true**, na **disjunção** quando um dos valores der **true** o resultado será **true**.

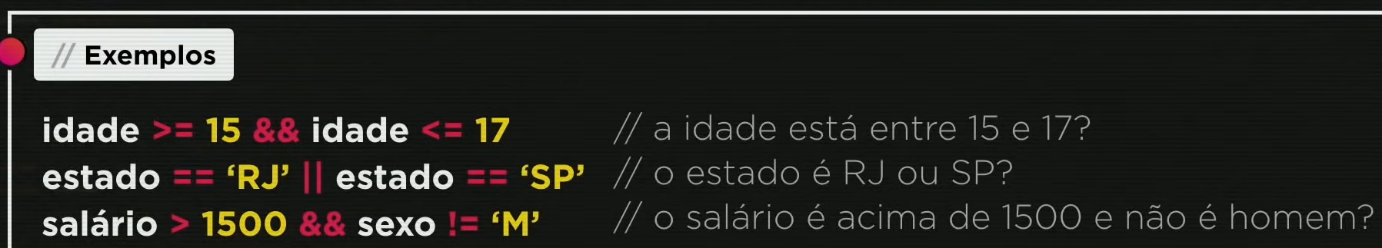


Se tivermos diversos **operadores** dentro de uma expressão, seguimos a **seguinte linha de** **raciocínio:**

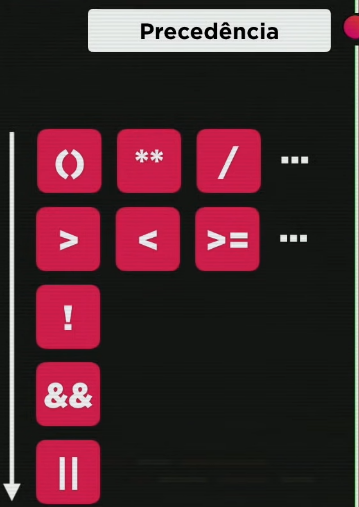
**=** (não)

**&&** (e)

**%** (ou)



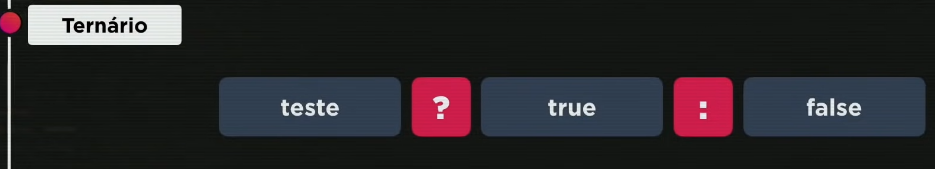
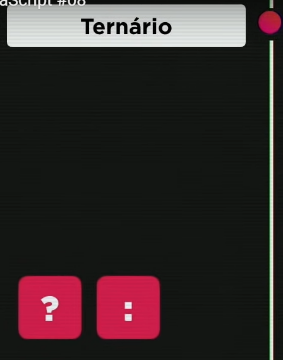
Abaixo temos uma tabela com a ordem de **precedência** dos **operadores:**

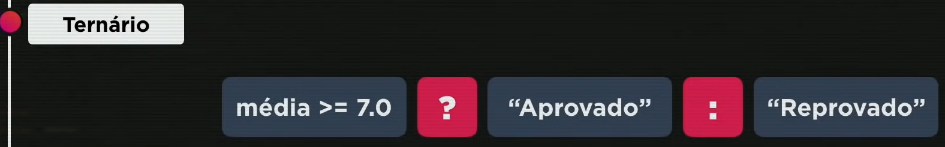
**1º Aritméticos**:

**2º Relacionais**: não tem ordem de precedência quem aparecer primeiro do será resolvido da esquerda para direita

**3º Lógicos**: !(não) &&(e) ||(ou)

Abaixo temos mais um operador, o operador **Ternário**, ele se chama **Ternário** por ter 3 testes operandos (operação):

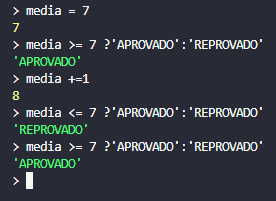
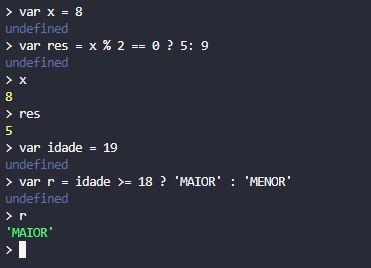




1º - faz um teste lógico.

2º - o que acontece quando o teste for **verdadeiro**.

3º - o que acontece quando o teste logico for **falso**.

Na primeira imagem damos a **variável media o valor 7**, depois perguntamos se a media e maior ou igual a 7, então o sistema nos retorna com **APROVADO**, pois a **média tem valor 7**, e **7 é igual a 7** então o valor e **true**.

Depois fazemos com que a **variável media** receba 1 acréscimo, digitando o comando **media+=1** sendo assim a **media passou a valer** 8, então abaixo estamos perguntando se a **média e menor ou igual a 7**, o sistema nos retornou como **REPROVADO** pois **8 não e igual e nem menor do que 7**, sendo assim o resultado **false.**

Na segunda imagem atribuímos a **variável idade o valor de 19**, depois atribuímos a **variável r** que seu valor o resultado da seguinte pergunta: **a variável idade é maior ou igual a 18?** a resposta e verdadeira **(true)** pois 19 é maior do que 18, sendo **true**, assim o sistema aplica o valor **MAIOR** para a **variável** **r**.

**MODULO C - DOM**

