



# NW01 - Lista Exercícios

## Introdução JS

---

### Enunciado do Projeto



### IMPORTANTE !

- 1) Antes de tudo: no github, vá até o Pull Request que você abriu anteriormente e verifique **se ele já foi aprovado**. Se sim, clique em **Merge Pull Request** para enviar todas as suas alterações para a branch `master`. Se não estiver aprovado **não tem problema!** Pode começar pelo passo 2 🙌🙌🙌
- 2) No seu computador, lembre-se de usar:
  - `git status` para ter certeza que não tem nada pendente
  - `git branch` para confirmar que você está na `master`
  - Caso não esteja, utilize `git checkout master`
  - `git pull`, garantindo que todas as informações remotas estão na sua branch local
- 3) Crie uma branch **a partir da branch master** para trabalhar no exercício de hoje. O nome da sua branch sempre deve seguir o formato: `quinzenaX-aulaY`
  - Utilize: `git checkout -b quinzena1-projeto`
- 4) Dentro da pasta `quinzena1` crie uma pasta chamada `lista-logica` e resolva todos os exercícios dentro dela
- 5) Realize a entrega através do formulário de entregas que você pode encontrar no canal de anúncios da sua turma

## Instruções Gerais

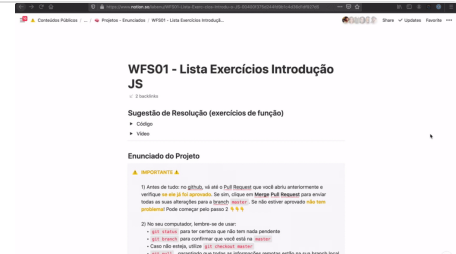
- Para fazer esses exercícios, **vocês devem usar o nosso boilerplate.**
  - ▼ Boilerplate

<https://s3-us-west-2.amazonaws.com/secure.notion-static.com/89df3f61-bbee-40c6-a712-77d7c67a2e97/lista-exercicios-js.zip>

## ▼ Explicação do boilerplate

### Explicação Lista Introdução JS

<https://www.loom.com/share/8073e0c7f25546c4aeb05e9fa92fddbfb>



- ⚠ Não modifiquem o arquivo `testes.js` ⚠
- ⚠ Não modifiquem os nomes das funções do arquivo `exercicios.js` ⚠
- Escrevam o código dentro da **função** correspondente ao número do exercício, no arquivo `exercicios.js`

## ▼ Dica

Você deve escrever o código dentro de uma função. Ainda vamos ver na semana que vem o que elas são e como funcionam certinho, mas para realizar a lista não precisamos saber disso! A única coisa que precisamos saber é onde colocar o nosso código, que é entre as duas chaves `{ }`, como as imagens abaixo indicam!



- Onde colocar o código:

```
function imprimeMensagem() {  
  // escreva seu código aqui!  
}
```

- Exemplo:

```
function imprimeMensagem() {  
  const mensagem = prompt('Digite uma mensagem!')  
  
  console.log(mensagem)  
}
```

- O local certo no arquivo para cada exercício será indicado com um comentário logo acima de cada espaço para escrever o código (que como vamos ver mais pra frente, chama-se função).
- Nos exercícios, você deve pedir dados ao usuário. Para isso, você deve usar a função `prompt()`. Você deve imprimir a resposta no console, usando o `console.log()`.
- Se vocês travarem em um exercício, pulem para o próximo. E, se mesmo assim continuarem muito travados/travadas, venham falar conosco.
- Os testes vão começar falando que vocês erraram a questão porque ainda não existe lógica dentro das funções! Não se desesperem com isso!
- Sintam-se livres para consultar os materiais das aulas anteriores.
- O projeto deve ser entregue até o final do dia 25/07, domingo.

 **Atenção:** para que os testes funcionem, você deve pedir as variáveis na mesma ordem que o enunciado indica 



## EXEMPLOS

Esses exemplos servem para você verificar o funcionamento dos testes automatizados que foram disponibilizados. Eles já estarão resolvidos.

### ▼ Exercício 0A

Escreva um código que **pede ao usuário** dois números e **imprime** a soma deles no console.

#### ▼ Exemplo 1

##### Entradas

```
num1: "1" //Usando prompt colete esse input do usuário, que aqui digita 1  
num2: "2" //Usando prompt colete esse input do usuário, que aqui digita 2
```

##### Saída

```
3 //Usando console.log imprima o valor de resposta que aqui é 1+2 = 3
```

#### ▼ Exemplo 2

##### Entradas

```
num3: "4"  
num4: "5"
```

##### Saída

```
9
```

#### ▼ Exercício 0B

Escreva um código que **pede ao usuário** uma **mensagem** e a **imprime** no console.

#### ▼ Exemplo 1

##### Entradas

```
mensagem: "Olá Mundo!"
```

##### Saída

```
"Olá Mundo!"
```

### ▼ Exemplo 2

#### Entradas

```
mensagem: "bananinha"
```

#### Saída

```
"bananinha"
```



## EXERCÍCIOS

### ▼ Exercício 1

Escreva um código que pede ao usuário a **altura** e a **largura** (nessa ordem) de um retângulo e **imprime no console** a **área** do retângulo.

#### ▼ Dica

A área de um retângulo pode ser calculada pela **multiplicação** da altura pela largura.

### ▼ Exemplo 1

#### Entradas

```
altura: 3  
largura: 5
```

#### Saída

```
15
```

### ▼ Exemplo 2

#### Entradas

```
altura: 2  
largura: 6
```

#### Saída

```
12
```

### ▼ Exercício 2

Escreva um código que pede ao usuário o **ano atual** e seu **ano de nascimento** e **imprima no console** sua **idade**.

#### ▼ Dica

Você pode desconsiderar o mês de nascimento da pessoa.  
Ou seja, assuma que ela já fez aniversário no ano atual.

### ▼ Exemplo 1

#### Entradas

```
ano atual: 2020  
ano de nascimento: 1990
```

#### Saída

```
30
```

### ▼ Exemplo 2

#### Entradas

```
ano atual: 2021  
ano de nascimento: 1967
```

## Saída

```
54
```

### ▼ Exercício 3

Escreva um código que pede ao usuário o **peso** em kg e a **altura** em metros de uma pessoa e **imprima no console** o seu IMC (Índice de Massa Corpórea).

#### ▼ Dica

O IMC pode ser calculado de acordo com a fórmula a seguir:

```
IMC = Peso ÷ (Altura × Altura)
```

Como essa conta não dá números exatos, serão aceitas respostas com uma diferença de 0.1 do valor esperado. Por exemplo, se o esperado para uma entrada específica era `21.6`, serão aceitas respostas entre `21.5` e `21.7`.

### ▼ Exemplo 1

#### Entradas

```
peso: 85  
altura: 1.8
```

#### Saída (aproximada)

```
26.2
```

### ▼ Exemplo 2

#### Entradas



```
peso: 70  
altura: 1.65
```

### Saída

```
25.7
```

#### ▼ Exercício 4

Escreva um código que pede ao usuário seu **nome**, sua **idade** e seu **email** (nessa ordem), e imprime no console uma mensagem como a seguinte:

```
Meu nome é {nome}, tenho {idade} anos, e o meu email é {email}.
```

#### ▼ Exemplo 1

##### Entradas

```
nome: "Alice"  
idade: 28  
email: alice@gmail.com
```

##### Saída (aproximada)

```
Meu nome é Alice, tenho 28 anos, e o meu email é alice@gmail.com.
```

#### ▼ Exemplo 2

##### Entradas

```
nome: "João"  
idade: 25  
email: "joao@gmail.com"
```

##### Saída (aproximada)

```
Meu nome é João, tenho 25 anos, e o meu email é joao@gmail.com.
```

### ▼ Exercício 5

Escreva um código que pede ao usuário suas três cores favoritas e imprime no console um array que contenha essas três cores.

#### ▼ Exemplo 1

##### Entradas

```
cor1: "Azul"  
cor2: "Amarelo"  
cor3: "Vermelho"
```

##### Saída

```
["Azul", "Amarelo", "Vermelho"]
```

#### ▼ Exemplo 2

##### Entradas

```
cor1: "Laranja"  
cor2: "Roxo"  
cor3: "Preto"
```

##### Saída

```
["Laranja", "Roxo", "Preto"]
```

### ▼ Exercício 6

Escreva um código que **pede ao usuário** uma **string** e **imprima no console** ela em letra maiúscula.

▼ Exemplo 1

**Entradas**

```
string: "oi"
```

**Saída**

```
"OI"
```

▼ Exemplo 2

**Entradas**

```
string: "bAnAnA"
```

**Saída**

```
"BANANA"
```

▼ Exercício 7

Escreva um código que **pede ao usuário** o **custo de um espetáculo de teatro** e o **valor de cada ingresso** (considere que todos os ingressos têm o mesmo preço) e **imprima no console** **quantos ingressos precisam ser vendidos** para que o espetáculo **não dê prejuízo**.

▼ Exemplo 1

**Entradas**

```
custo: 3000  
valor de cada ingresso: 100
```

### Saída

```
30
```

### ▼ Exemplo 2

#### Entradas

```
custo: 5500  
valor de cada ingresso: 50
```

### Saída

```
110
```

### ▼ Exercício 8

Escreva um código que **pede ao usuário duas strings** e **imprima no console** um booleano (true ou false) indicando se elas possuem o mesmo tamanho.

### ▼ Exemplo 1

#### Entradas

```
string1: "ola"  
string2: "abc"
```

### Saída

```
true
```

### ▼ Exemplo 2

#### Entradas

```
string1: "teste"  
string2: "porta"
```

#### Saída

```
true
```

### ▼ Exemplo 3

#### Entradas

```
string1: "abc"  
string2: "abcd"
```

#### Saída

```
false
```

### ▼ Exercício 9

Escreva um código que pede ao usuário duas strings e **imprima no console** um booleano (true ou false) indicando se elas são iguais, desconsiderando letras maiúsculas ou minúsculas.

### ▼ Exemplo 1

#### Entradas

```
string1: "Ola"  
string2: "oLA"
```

#### Saída

```
true
```

#### ▼ Exemplo 2

##### Entradas

```
string1: "bananinha"  
string2: "BANANINHA"
```

##### Saída

```
true
```

#### ▼ Exemplo 3

##### Entradas

```
string1: "banana"  
string2: "BANANINHA"
```

##### Saída

```
false
```

#### ▼ Exemplo 4

##### Entradas

```
string1: "cAsA"  
string2: "MeSA"
```

##### Saída

```
false
```

### ▼ Exercício 10

Escreva um código que **pede ao usuário o ano atual, seu ano de nascimento, e o ano em que sua carteira de identidade foi emitida** (nessa ordem). A função deve imprimir no console um booleano (true ou false) que indica se a carteira precisa ser renovada ou não. A carteira precisa ser renovada segundo os seguintes critérios:

- Para pessoas com menos de 20 anos, ou exatamente 20 anos, deve ser renovada de 5 em 5 anos (se for exatamente 5 anos, deve ser renovada).
- Para pessoas entre 20 e 50 anos, ou exatamente 50, deve ser renovada de 10 em 10 anos (se for exatamente 10 anos, deve ser renovada).
- Para pessoas acima dos 50 anos, deve ser renovada de 15 em 15 anos

### ▼ Dica

Você deve usar comparadores e operadores booleanos para avaliar as condições. Para te ajudar a organizar a lógica, tente criar 3 variáveis separadas, uma para cada condição, e depois compará-las.

### ▼ Exemplo 1

#### Entradas

```
ano atual: 2020  
ano de nascimento: 2000  
ano de emissao do RG: 2015
```

#### Saída

```
true
```

### ▼ Exemplo 2

## Entradas

```
ano atual: 2020  
ano de nascimento: 2000  
ano de emissao do RG: 2016
```

## Saída

```
false
```

### ▼ Exemplo 3

## Entradas

```
ano atual: 2020  
ano de nascimento: 1990  
ano de emissao do RG: 2015
```

## Saída

```
false
```

### ▼ Exemplo 4

## Entradas

```
ano atual: 2020  
ano de nascimento: 1990  
ano de emissao do RG: 2009
```

## Saída

```
true
```

### ▼ Exemplo 5

## Entradas



```
ano atual: 2010  
ano de nascimento: 1959  
ano de emissao do RG: 2000
```

### Saída

```
false
```

### ▼ Exemplo 6

#### Entradas

```
ano atual: 2010  
ano de nascimento: 1959  
ano de emissao do RG: 1990
```

### Saída

```
true
```

### ▼ Exercício 11

Escreva um código que pede ao usuário um ano e **imprima no console** um booleano (true ou false) que indica se o ano é bissexto. Um ano é bissexto de acordo com as seguintes condições:

- **São bissextos** todos os anos múltiplos de 400.
- **São bissextos** todos os múltiplos de 4, exceto se for múltiplo de 100 mas não de 400.
- **Não são bissextos** todos os demais anos.

### ▼ Dica

Você deve usar comparadores e operadores booleanos para avaliar as condições. Para te ajudar a organizar a lógica,

tente criar 3 variáveis separadas, uma para cada condição, e depois compará-las.

#### ▼ Exemplo 1

##### **Entradas**

```
ano: 2000
```

##### **Saída**

```
true
```

#### ▼ Exemplo 2

##### **Entradas**

```
ano: 2012
```

##### **Saída**

```
true
```

#### ▼ Exemplo 3

##### **Entradas**

```
ano: 1900
```

##### **Saída**

```
false
```

#### ▼ Exemplo 4

##### **Entradas**

```
ano: 2001
```

### Saída

```
false
```

#### ▼ Exercício 12

Escreva um código que faz as seguintes perguntas ao usuário (condições para ser uma pessoa estudante da Labenu):

- Você tem mais de 18 anos?
- Você possui ensino médio completo?
- Você possui disponibilidade exclusiva durante os horários do curso?

O usuário deve responder essas perguntas com `"sim"` ou `"nao"`.

A função deve imprimir no console um booleano (`true` ou `false`) que indica se a inscrição para o curso é válida, ou seja, se o usuário pode ou não fazer o curso da Labenu. A inscrição é válida quando todas as respostas para todas as perguntas for positiva.

#### ▼ Exemplo 1

##### Entradas

```
tem mais de 18?: "sim"  
tem ensino médio completo?: "sim"  
tem disponibilidade de horários?: "sim"
```

### Saída

```
true
```

#### ▼ Exemplo 2

## Entradas

```
tem mais de 18?: "nao"  
tem ensino médio completo?: "sim"  
tem disponibilidade de horários?: "sim"
```

## Saída

```
false
```

### ▼ Exemplo 3

## Entradas

```
tem mais de 18?: "sim"  
tem ensino médio completo?: "nao"  
tem disponibilidade de horários?: "sim"
```

## Saída

```
false
```

### ▼ Exemplo 4

## Entradas

```
tem mais de 18?: "sim"  
tem ensino médio completo?: "sim"  
tem disponibilidade de horários?: "nao"
```

## Saída

```
false
```

### ▼ Exemplo 5

## Entradas

```
tem mais de 18?: "nao"  
tem ensino médio completo?: "nao"  
tem disponibilidade de horários?: "nao"
```

## Saída

```
false
```