

## Estruturas de Decisão

### E se...?

Você já percebeu que constantemente nos fazemos perguntas como:

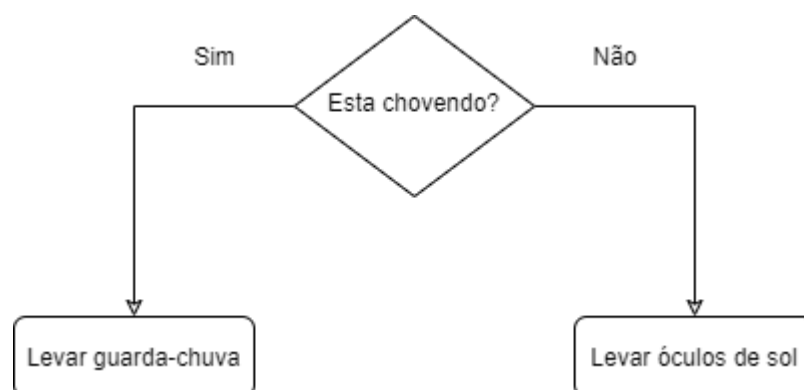
- Será que vai chover hoje?
- O que vou comer no almoço?
- Será que esse caminho é o mais curto?

O mais importante é que dependendo da resposta, tomamos ações diferentes!

Como no caso de chuva, se for chover, com certeza você não vai querer se molhar, e irá levar um guarda-chuva ou uma capa para se proteger. Mas, se porventura fizer aquele solzão, você não vai querer ficar andando por aí com o guarda chuva à toa!

Esses questionamentos que fazemos constantemente, geralmente de forma tão natural, são a base para nosso próximo passo rumo ao aprendizado de programação: compreender as estruturas de decisão!

Você vai perceber que na programação, quando fazemos alguma indagação, dependendo da resposta obtida, as ações que tomamos podem ser distintas. Chamamos isso de **estrutura condicional**, e com ela podemos criar fluxos incríveis dentro do nosso sistema. Eles podem ser interpretados assim:



Estudar as estruturas condicionais será importante para desenvolver nossos futuros códigos, pois essas estruturas permitem que o nosso código esteja preparado para executar ações diferentes, com base na escolha do usuário!

A partir desse módulo, iremos te apresentar como criar em condições com Javascript, e adicionar mais um recurso às nossas habilidades.

## Estruturas de Decisão

# Condicional IF - simples

### Vamos ver como funciona esse “se...”

O conceito de estruturas condicionais é muito útil na programação. Sem ele nosso código executa sempre da mesma forma, tem sempre o mesmo resultado, e dessa maneira, nós não conseguimos criar muitas soluções. Com as estruturas condicionais, podemos escrever trechos de código que só serão executados caso passem por uma verificação.

Em Javascript, temos várias formas de utilizar as estruturas condicionais. Uma delas é o comando **IF** (SE em inglês) que pode ser utilizada para verificar várias informações, como:

- Se a variável tem o valor que esperamos;
- Se o valor é maior, menor, diferente, etc.

Na sintaxe do IF , precisamos prestar atenção para não esquecer nenhum detalhe!

- Sempre teremos o comando **IF** seguido do conjunto de parênteses que engloba a condição;
- Depois do fechamento dos parênteses, temos abertura e fechamento de chaves, que engloba o código que deve ser executado caso a condição seja verdadeira.

Veja um exemplo dessa estrutura:

```
if ( condição ) {  
  
    // código que será executado caso a condição seja verdadeira  
  
}
```

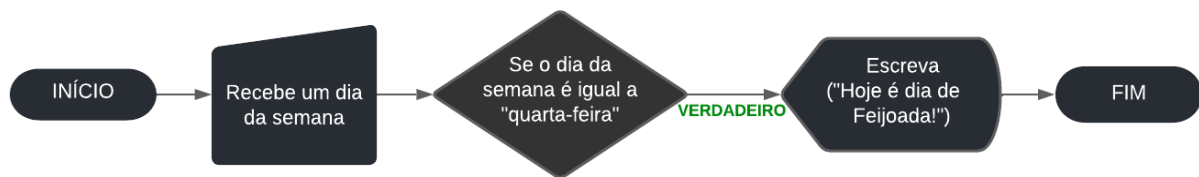
Muito confuso? Vamos ver uma aplicação desses conceitos?

## Situação Problema 1:

Um grupo de amigos resolveu que toda quarta-feira é dia de feijoada! E para lembrar a todos desse momento festivo, resolveram criar um algoritmo que verifica em qual dia da semana estamos.

“Se hoje for quarta-feira”, significa que é dia de Feijoada! E o algoritmo precisa emitir uma mensagem informando isso.

Vamos ver esse exemplo em fluxograma:



Observe que o fluxo é bem simples, contínuo, e que para o programa conseguir ser executado completamente, a resposta para a pergunta feita precisa ser verdadeira!

Vamos reescrever esse fluxograma em Portugol para entender como se constrói essa estrutura condicional IF. Para isso funcionar, precisamos armazenar o dia da semana em uma variável do tipo caracter, e depois verificar se o dia corresponde ao que buscamos, como mostrado no exemplo a seguir:

```
1 programa
2 {
3     funcao inicio ()
4     {
5         caracter diaDaSemana = "quarta-feira"
6
7         se (diaDaSemana == "quarta-feira") {
8             escreva("Hoje é dia de feijoada!")
9         }
10    }
11 }
12
13
```

Hoje é dia de feijoada!  
Programa finalizado. Tempo de execução: 18 ms

Agora, vamos passar esse código para o Javascript! Aqui temos a variável **diaDaSemana** (que em JavaScript não precisa ter o seu tipo definido) e logo abaixo, um exemplo de condição utilizando **IF**.

```
var diaDaSemana = "quarta-feira"

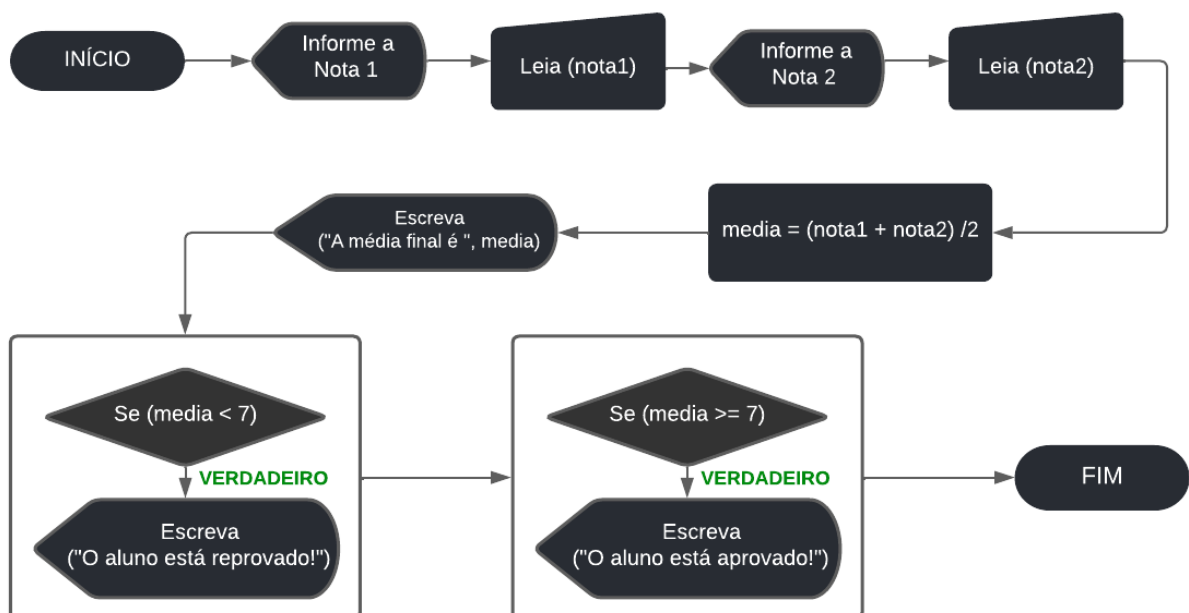
if (diaDaSemana == "quarta-feira") {
  console.log("Hoje é dia de feijoada!")
}
```

Nesse código, a condição verifica se a variável **diaDaSemana** possui o valor **igual** à “quarta-feira”. Em caso positivo, será executada a próxima linha com o comando **console.log**, que imprime a mensagem “**Hoje é dia de feijoada!**”.

Caso a variável tenha um valor diferente do esperado, o código que está entre as chaves será ignorado, e a execução continua na linha após o fechamento das chaves.

## Situação Problema 2:

Um Professor precisa automatizar os resultados dos alunos referente às avaliações que foram realizadas no final do curso e pediu nossa ajuda para criar um algoritmo que calcule a média de duas notas e informe se o aluno foi aprovado ou não! Vamos analisar essa situação através da construção de um Fluxograma:

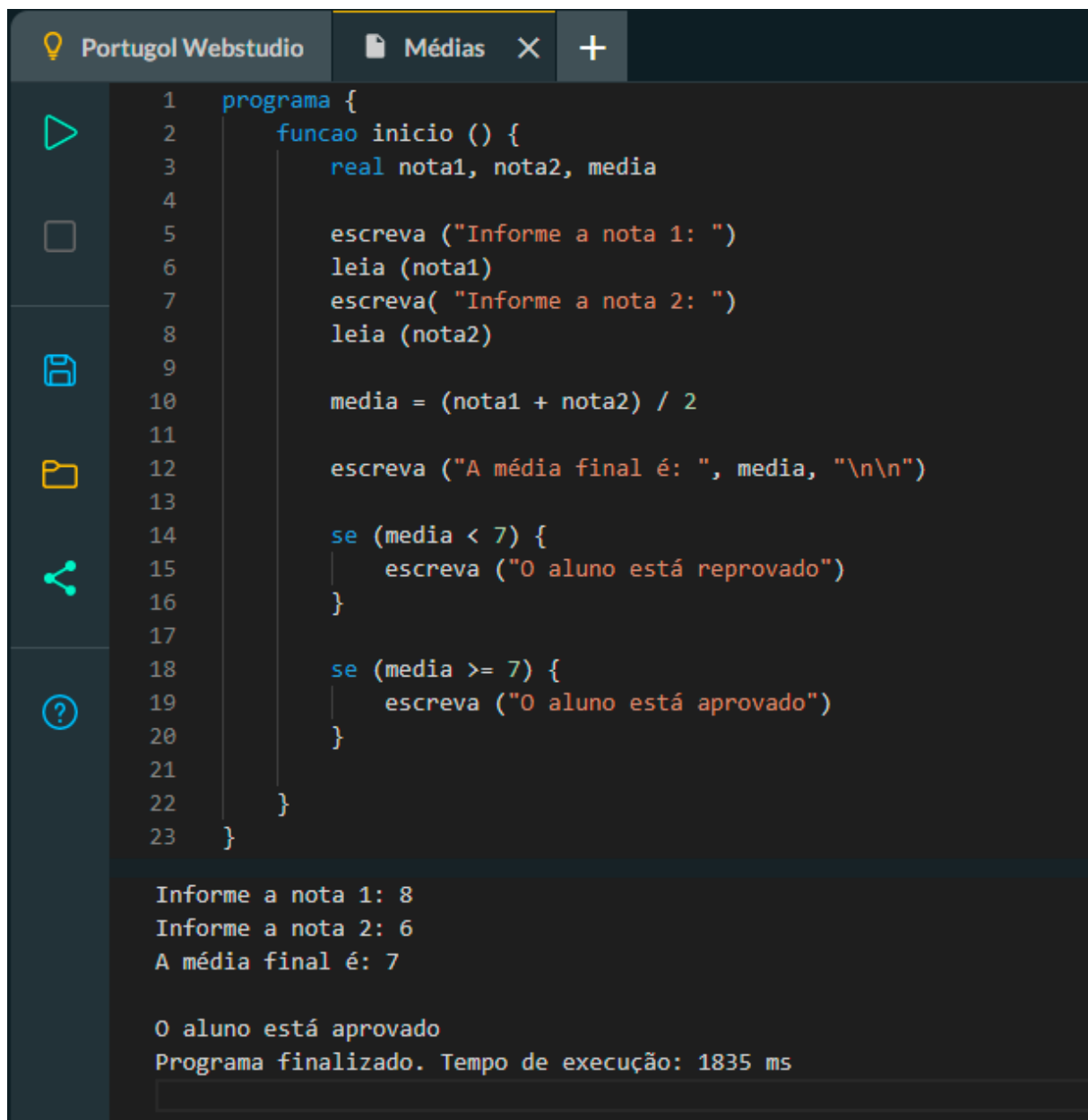


De acordo com o diagrama, vimos que o fluxo também é contínuo, mas nesse caso, temos duas verificações: Se média menor que sete ( $media < 7$ ) e se média maior ou igual a sete ( $media \geq 7$ ).

Observando o fluxograma, podemos entender que ao passar pelas condicionais, o programa deve verificar se as condições são verdadeiras, e caso sejam, deve executar o código correspondente.

Porém, em caso de a condição não ser atendida, o programa “pula” o trecho de código correspondente e o fluxo segue normalmente.

Vamos verificar como ficaria esse código utilizando o Portugol?



```
1  programa {
2      funcao inicio () {
3          real nota1, nota2, media
4
5          escreva ("Informe a nota 1: ")
6          leia (nota1)
7          escreva( "Informe a nota 2: ")
8          leia (nota2)
9
10         media = (nota1 + nota2) / 2
11
12         escreva ("A média final é: ", media, "\n\n")
13
14         se (media < 7) {
15             escreva ("O aluno está reprovado")
16         }
17
18         se (media >= 7) {
19             escreva ("O aluno está aprovado")
20         }
21
22     }
23 }
```

Informe a nota 1: 8  
Informe a nota 2: 6  
A média final é: 7

O aluno está aprovado  
Programa finalizado. Tempo de execução: 1835 ms

No código em Portugol, simulamos as notas 8 e 6, resultando na média 7. Nesse caso, o aluno foi “aprovado”. Observe que o programa “ignorou” o trecho de código da linha 15, pois a condição ( $media < 7$ ) não foi atendida!

E como será que escrevemos esse mesmo código utilizando o JavaScript? Será que é muito diferente do que já vimos? Vejamos:

```
script.js x
1
2 var nota1, nota2, media
3
4 nota1 = parseInt(prompt("Informe a nota 1: "));
5 nota2 = parseInt(prompt("Informe a nota 2: "));
6
7 media = (nota1 + nota2)/2
8
9 console.log("A média final é: " + media + "\n\n")
10
11 if (media < 7) {
12     console.log("O aluno está reprovado");
13 }
14
15 if (media >= 7) {
16     console.log("O aluno está aprovado");
17 }
18
```

Neste exemplo, utilizamos os métodos “**parseInt()**” e “**prompt()**” para conseguirmos ler as notas do usuário, mas não se preocupe com eles por enquanto, estudaremos os comandos de entrada e saída no javascript nos próximos módulos do curso.

Ao executar esse código, você provavelmente verá uma caixa de diálogo solicitando os dados de entrada (nota1 e nota2), em seguida, poderá ver o resultado no console.

# Exercícios

## IF - simples

1. Precisamos de um código que verifique a idade do usuário para ver se já tem idade mínima para adquirir carta de habilitação. Para isso, teremos uma variável **nome** que guarda um valor do tipo **string**, e uma variável **idade** que guarda um valor do tipo **numérico**.

Caso o usuário tenha mais de 18 anos ou mais, imprima a seguinte frase: "Parabéns, você já pode dirigir".

2. Você foi contratado por um parque de diversões para criar um sistema que mostre se seus usuários estão gostando da nova montanha russa. Crie uma variável chamada **notasUsuarios** que receberá um valor do tipo **numérico**.

Agora crie um código que imprima na tela os textos:

- "Excelente" para notas entre 9 e 10,
- "Muito bom" para notas entre 6 e 8
- "Não Gostei" para notas Menores que 5

Desenvolva esse código usando apenas a condicional **IF** (quantas vezes for necessário)



## Estruturas de Decisão

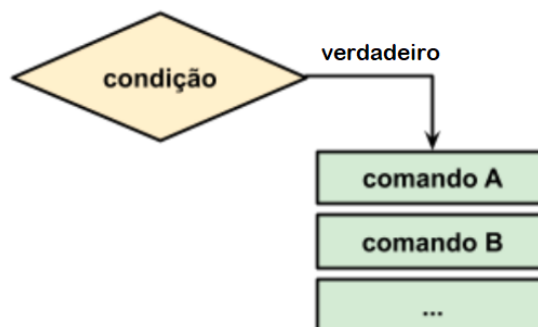
## Condicional IF ELSE - composto

### Vamos ver o que ele tem de diferente do IF?

Agora que já sabemos a importância da condicional e como utilizar o IF, vamos aprender uma variação dessa condicional, o IF ELSE.

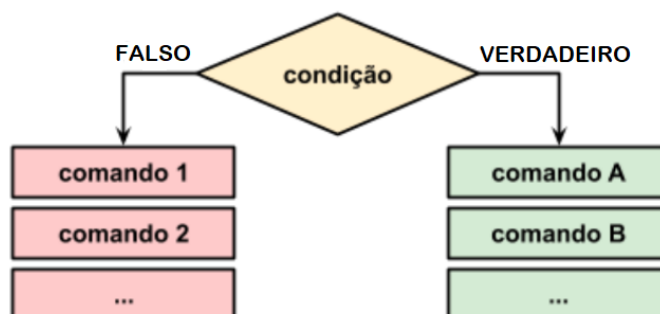
Você deve ter percebido que quando fazemos um **IF simples**, quando a condição retorna verdadeiro, é executado o bloco de código que está dentro das chaves do IF. Caso a condição retorne “falso”, o programa apenas “ignora” as linhas dentro do bloco do IF e segue o fluxo normal.

```
if (condição) {  
    comando A;  
    comando B;  
    ...  
}
```



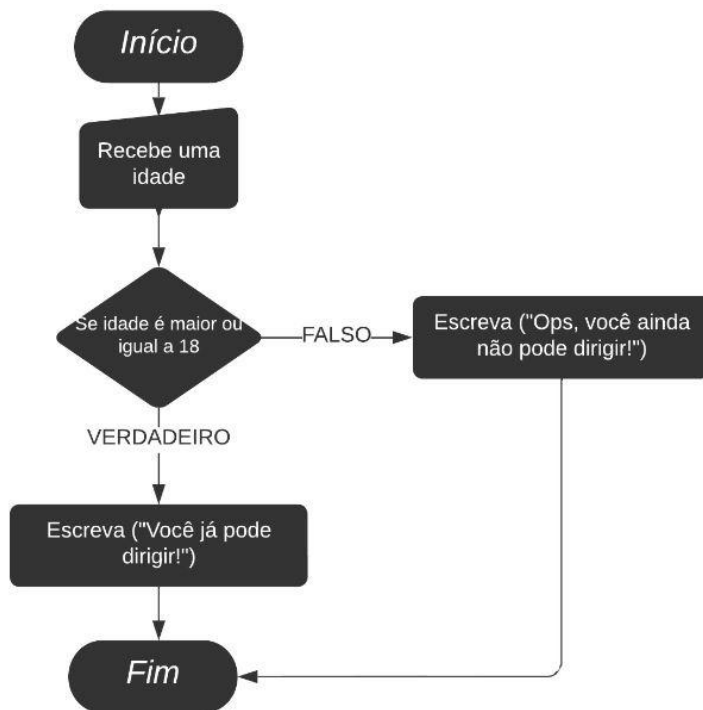
Mas, e se o resultado da condição for falso, podemos definir algum comando para essa situação também? Sim! Para isso usamos o comando **ELSE** (do inglês *senão*).

```
if (condição) {  
    comando A;  
    comando B;  
    ...  
} else {  
    comando 1;  
    comando 2;  
    ...  
}
```



### Situação Problema 3:

O Departamento de Trânsito da sua cidade procurou você para criar um algoritmo que receba do usuário a sua idade e retorne como resposta a informação se o mesmo pode ou não dirigir. Sabendo que a idade mínima para solicitar a carteira de motorista é 18 anos. Vejamos essa aplicação em um exemplo com um fluxograma:



Analisando o desenho ao lado, percebemos que o fluxo do algoritmo segue normalmente, caso a pessoa tenha idade maior ou igual a 18 anos, apresentando a mensagem **"Você já pode dirigir"** e depois encerra o algoritmo.

Todavia, se a pessoa responder que não possui a idade mínima, o fluxo sofre um pequeno desvio, alterando a resposta para **"Ops, você ainda não pode dirigir"**, e depois encerra o algoritmo.

Vamos traduzir esse fluxograma para o Português Estruturado?

```
1 programa {
2     funcao inicio () {
3         inteiro idade
4
5         escreva("Informe sua idade: ")
6         leia(idade)
7
8         se (idade >= 18) {
9             escreva("Você já pode dirigir")
10        }
11        senao {
12            escreva("Ops, você ainda não pode dirigir")
13        }
14    }
15 }
16
```

Vejamos como seria a saída caso o usuário digite 21 anos ou 15 anos:

```
Informe sua idade: 21
Você já pode dirigir
Programa finalizado. Tempo de execução: 5597 ms

Informe sua idade: 15
Ops, você ainda não pode dirigir
Programa finalizado. Tempo de execução: 1185 ms
```

A seguir, vamos reescrever esse código utilizando a linguagem JavaScript, e assumindo que a idade será 17:

```
1  var idade = 17
2
3  if (idade >= 18) {
4      console.log("Você já pode dirigir!")
5  } else {
6      console.log("Ops, você ainda não pode dirigir");
7  }
8
```

Como o valor da variável **idade** é 17, e a condição é que a idade seja **maior ou igual** a 18, a afirmação é falsa, ou seja, ele vai ignorar o primeiro bloco de código (linha 4) e entrar no **ELSE**, que tem outro bloco de código (linha 6).

## Sintaxe IF ELSE

O comando **ELSE** vem para complementar nossa estrutura do **IF**. Por via de regra, podemos assumir que NUNCA haverá um ELSE sem que haja um IF antes, pois eles se completam!

Veja que devemos colocá-lo logo após o fechamento de chaves do IF, e ele tem o próprio bloco de código, também dentro de chaves:

```
if ( condição ) {
    // código que será executada caso a condição seja verdadeira
} else {
    // código que será executado caso a condição seja falsa
}
```

A utilização do **ELSE** é opcional e depende muito da forma como o programador pensa a estrutura de resolução do problema. O ELSE vai sempre trazer um trecho de código que deve ser executado como padrão, no caso da condição do IF não ser verdadeira.

Quando criamos um código, precisamos estar preparados para todas as situações possíveis e prever as possíveis ações que o usuário pode tomar, por exemplo:

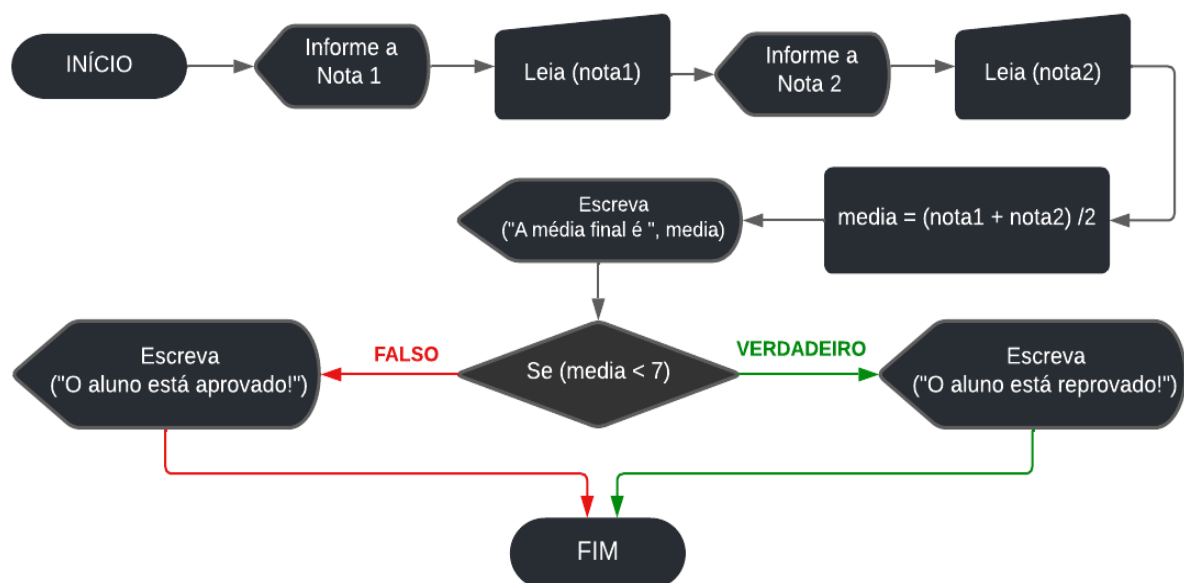
- e se um usuário colocar um número negativo ou zero?
- e se o usuário digitar um número maior do que o esperado?
- e se o número for menor que o esperado?
- e se o usuário colocar exatamente o número que buscamos?

Buscamos sempre desenvolver códigos que estejam preparados para cada uma das probabilidades, de forma que deve existir um trecho de código que trata cada situação possível.

Programar é criar soluções, o código é somente o meio, por isso, pense primeiro no problema que precisa resolver, depois pense numa solução e identifique quais ferramentas/comandos podem te ajudar no processo.

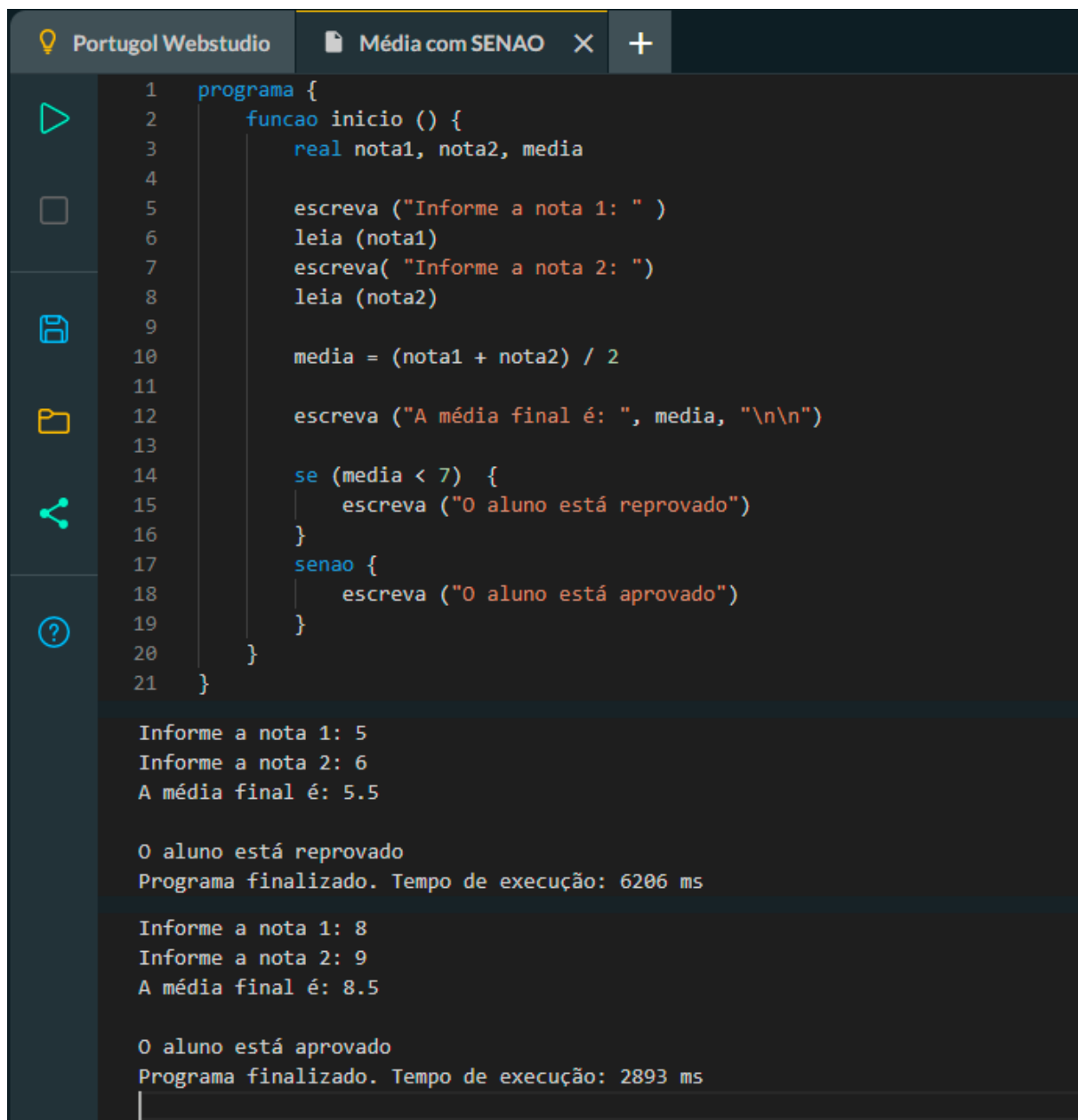
## Situação Problema 4:

Vamos resgatar a Situação Problema 2, que envolve a criação de um algoritmo para calcular a média de notas e informar a situação final do aluno. Porém, desta vez, iremos criar uma Condicional utilizando o IF ELSE. Vejamos como isso ficaria no fluxograma:



Observe que, neste caso, o fluxo não será contínuo pois em um determinado momento, quando tivermos o valor da média, o programa deve seguir por um caminho ou outro, nunca ocorrerá de executar as duas opções.

Ou seja, se a média for menor que 7, o programa seguirá para o comando que imprime “O aluno está reprovado!”, caso contrário, ele irá seguir para o comando que imprime “O aluno está aprovado!”. Vejamos como tratar essa informação em Portugol:



```
1 programa {
2     funcao inicio () {
3         real nota1, nota2, media
4
5         escreva ("Informe a nota 1: ")
6         leia (nota1)
7         escreva( "Informe a nota 2: ")
8         leia (nota2)
9
10        media = (nota1 + nota2) / 2
11
12        escreva ("A média final é: ", media, "\n\n")
13
14        se (media < 7) {
15            escreva ("O aluno está reprovado")
16        }
17        senao {
18            escreva ("O aluno está aprovado")
19        }
20    }
21 }
```

Informe a nota 1: 5  
Informe a nota 2: 6  
A média final é: 5.5

O aluno está reprovado  
Programa finalizado. Tempo de execução: 6206 ms

Informe a nota 1: 8  
Informe a nota 2: 9  
A média final é: 8.5

O aluno está aprovado  
Programa finalizado. Tempo de execução: 2893 ms

No exemplo acima, fizemos dois testes, o primeiro recebendo os valores 5 e 6 para gerar a média 5.5 e resultado reprovado e o segundo com os valores 8 e 9 para gerar média 8.5 e o resultado aprovado!

# Exercícios

## IF ELSE - composto

1. Desenvolva um código que receba uma variável alturaPessoa que guarda um valor do tipo numérico (inteiro).

Caso a altura do usuário seja maior ou igual a 160cm, imprima na tela: "Você poderá pilotar o carro de F1!"

Se a altura for inferior a 170cm, imprima: "Você não tem a altura mínima para pilotar esta máquina!".

2. Utilize o mesmo código do primeiro exercício e desta vez informe ao usuário com idades abaixo de 18 anos a seguinte frase: "Infelizmente você não tem autorização para dirigir"

## Estruturas de Decisão

# Condicional IF ELSE IF

O que isso quer dizer? Vamos ver com calma.

Quantas coisas aprendemos até aqui certo?

Vimos que podemos verificar condições e atribuir ações que deverão ser feitas usando o **IF** e, não só isso, caso essa primeira condição não seja verdadeira, podemos usar o **ELSE** para atribuir uma ação padrão, cobrindo assim boa parte das situações.

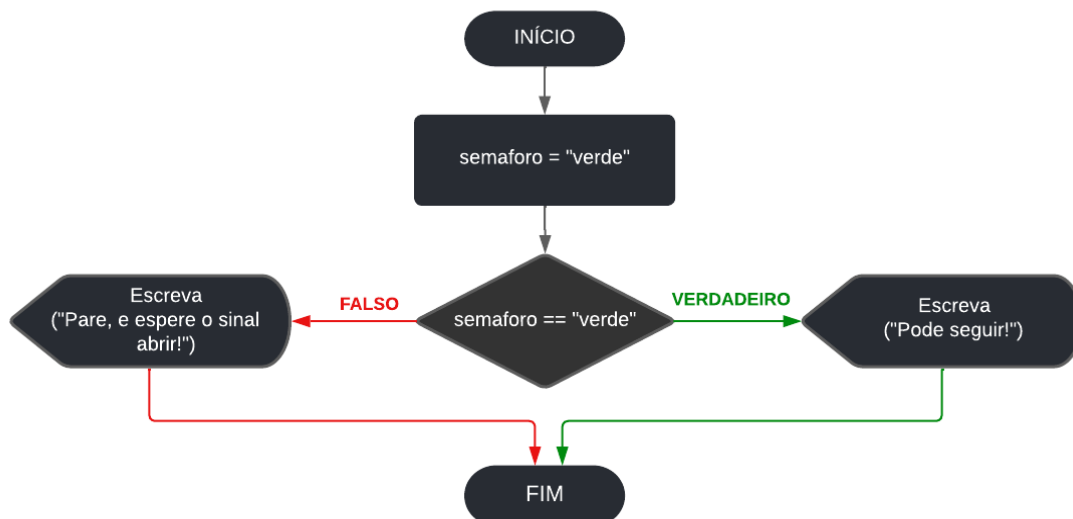
Entretanto, nem sempre podemos resumir nossos códigos em apenas duas possibilidades. O **IF ELSE IF** serve para criarmos mais uma condição, ou duas, talvez três... Na verdade não temos um limite aqui, então você pode avaliar diversas condições! E, em questão de sintaxe, ele se assemelha muito com a dupla **IF ELSE**, porém, incorpora mais condições após o else, e assim reforçar as regras de entrada no bloco:

```
if ( primeira condição ) {  
    // código que será executado caso a 1ª condição seja verdadeira  
} else if (segunda condição) {  
    // código que será executado caso  
    // a 1ª condição seja falsa e 2ª condição seja verdadeira  
} else {  
    // código que será executado caso a  
    // 1ª e 2ª condições ambas sejam falsas  
}
```

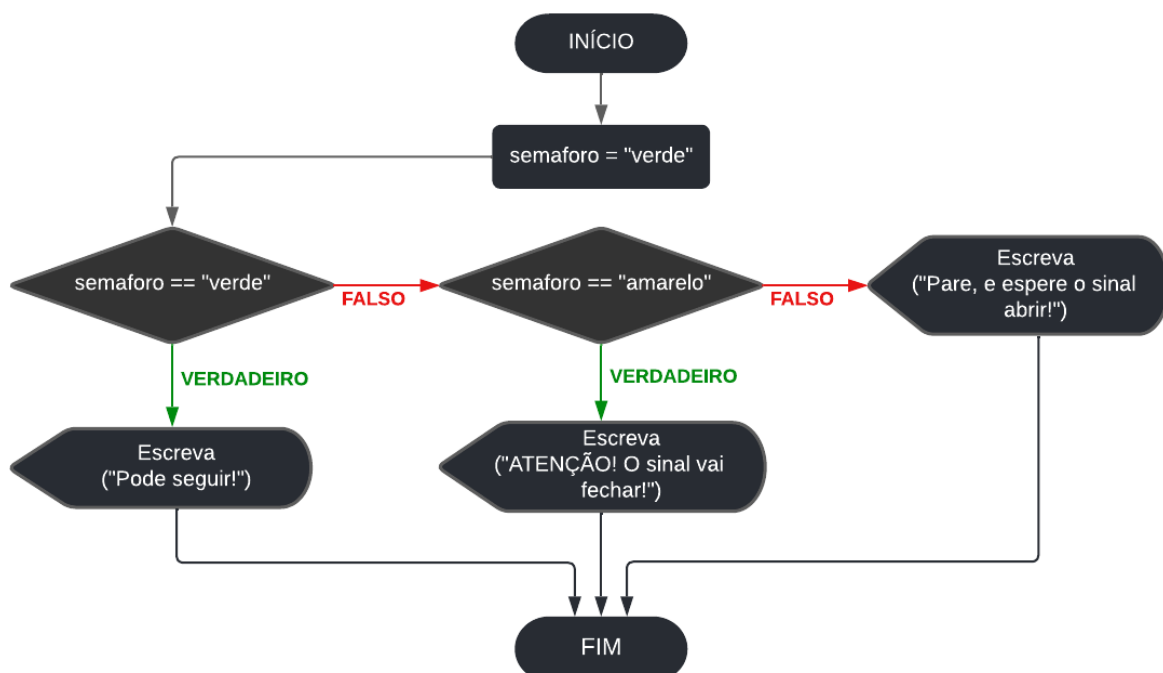
Vamos analisar o que acabamos de falar com mais uma situação problema.

## Situação Problema 5:

A Prefeitura da sua cidade ficou sabendo que você é programador e o procurou para ajudá-los com um novo projeto! Eles pretendem instalar um semáforo no cruzamento mais movimentado do seu bairro, mas precisam de algoritmo que faça os semáforos funcionarem! Bom, primeiro precisamos pensar nos passos que envolvem esse problema, e talvez, ao tentar simular um sistema usando as condicionais poderíamos ter algo assim:



Tudo parece bem, certo? Mas, espere um pouco, um semáforo tem 3 cores, ou seja, 3 possibilidades e não apenas duas! Precisamos pensar na ação para o sinal amarelo também, mas o que faremos? E é aqui que usamos o **IF ELSE IF**. Vamos ver como ficaria o fluxograma do semáforo, agora com as 3 possibilidades:



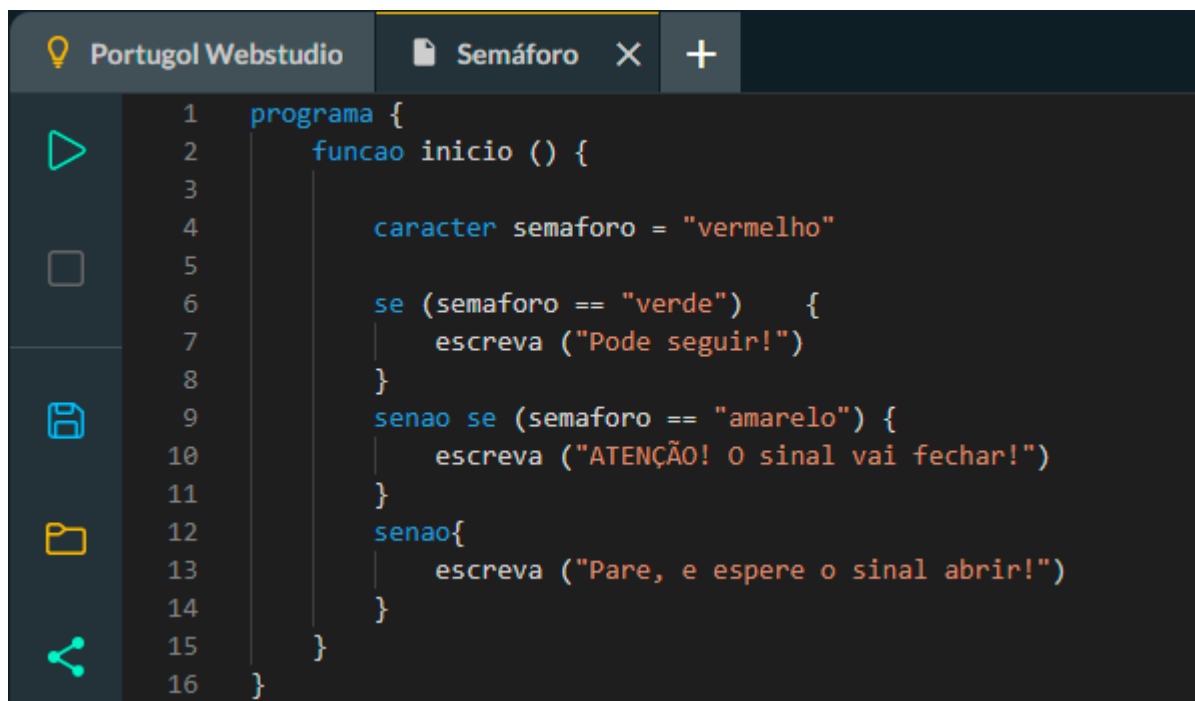


Observe que a primeira possibilidade verificada é se o **“sinal está verde”**, e se isso for verdadeiro, o programa imprimirá a frase **“Pode seguir!”**, caso contrário, ele irá para a próxima verificação.

A segunda possibilidade verifica se o **“sinal está amarelo”**, e se a resposta for verdadeira, o programa deverá imprimir a frase **“ATENÇÃO! O sinal vai fechar!”**, caso contrário, irá para o próximo passo.

E por último, não precisamos abrir uma terceira possibilidade, pois **“o sinal vermelho”** é a única alternativa possível depois de passar pelas outras verificações.

Em Portugol teríamos algo como:

A screenshot of the Portugol Webstudio interface. The window title is 'Semáforo'. The code is written in Portugol and implements a traffic light logic. It starts with a 'programa' block containing an 'inicio' function. Inside the function, a variable 'semaforo' is initialized to 'vermelho'. Then, there are three conditional blocks: 'se (semaforo == "verde")' which prints 'Pode seguir!'; 'senao se (semaforo == "amarelo")' which prints 'ATENÇÃO! O sinal vai fechar!'; and 'senao' which prints 'Pare, e espere o sinal abrir!'. The code is numbered from 1 to 16.

```
1 programa {
2     funcao inicio () {
3
4         caracter semaforo = "vermelho"
5
6         se (semaforo == "verde") {
7             escreva ("Pode seguir!")
8         }
9         senao se (semaforo == "amarelo") {
10             escreva ("ATENÇÃO! O sinal vai fechar!")
11         }
12         senao{
13             escreva ("Pare, e espere o sinal abrir!")
14         }
15     }
16 }
```

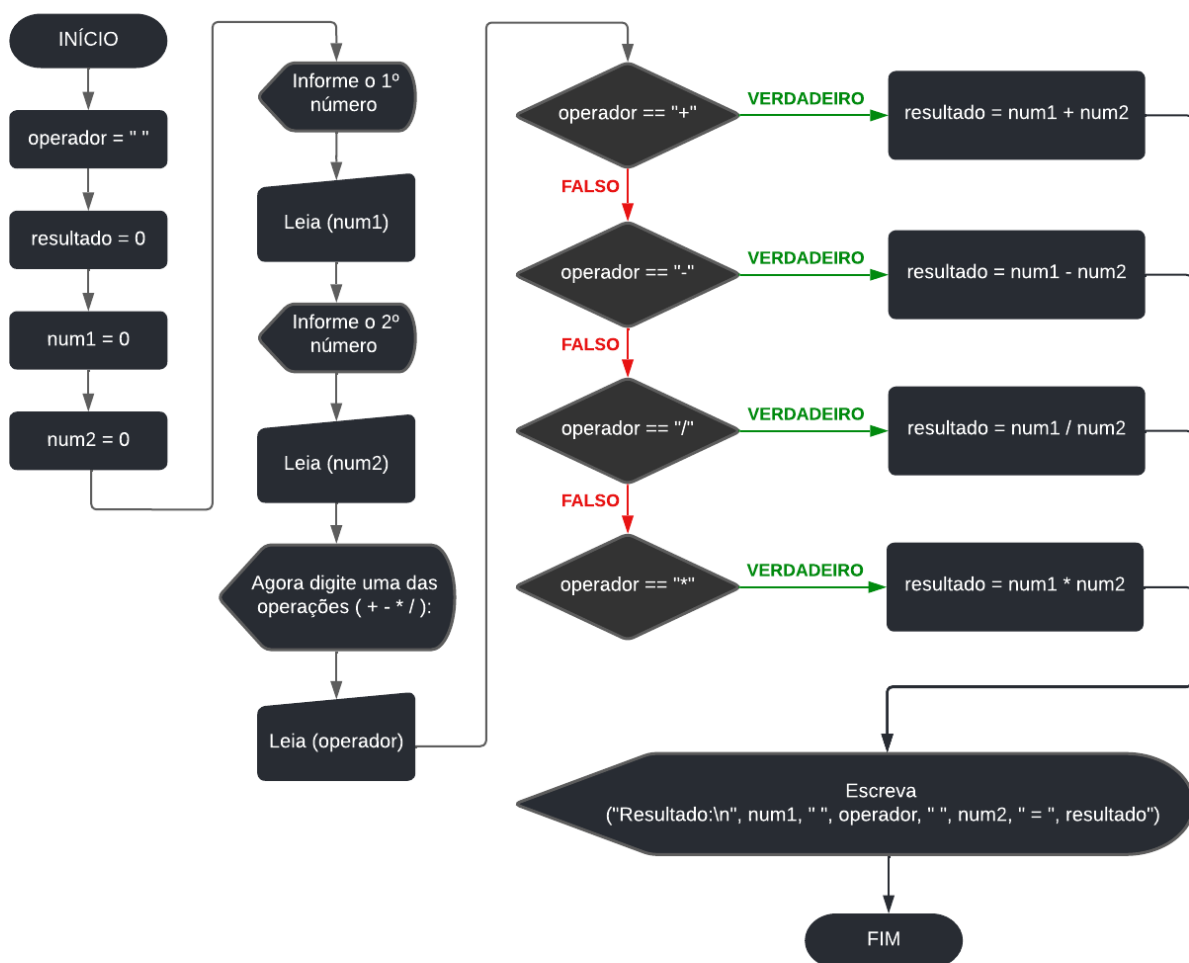
Já no JavaScript, o código ficaria mais ou menos assim:

```
if (semaforo == 'Verde') {
    console.log('Pode seguir!')
} else if (semaforo == 'amarelo') {
    console.log('Atenção, o sinal irá fechar!')
} else {
    console.log('Pare, e espere o sinal abrir')
}
```

**PARA VOCÊ PENSAR:** E se o usuário digitasse ou definisse qualquer outro valor diferente “vermelho”, como nosso programa iria reagir? Existe alguma forma mais eficiente de montarmos esse programa? Crie suas hipóteses e discuta com seus colegas. Enquanto isso, vamos resolver outra situação juntos?

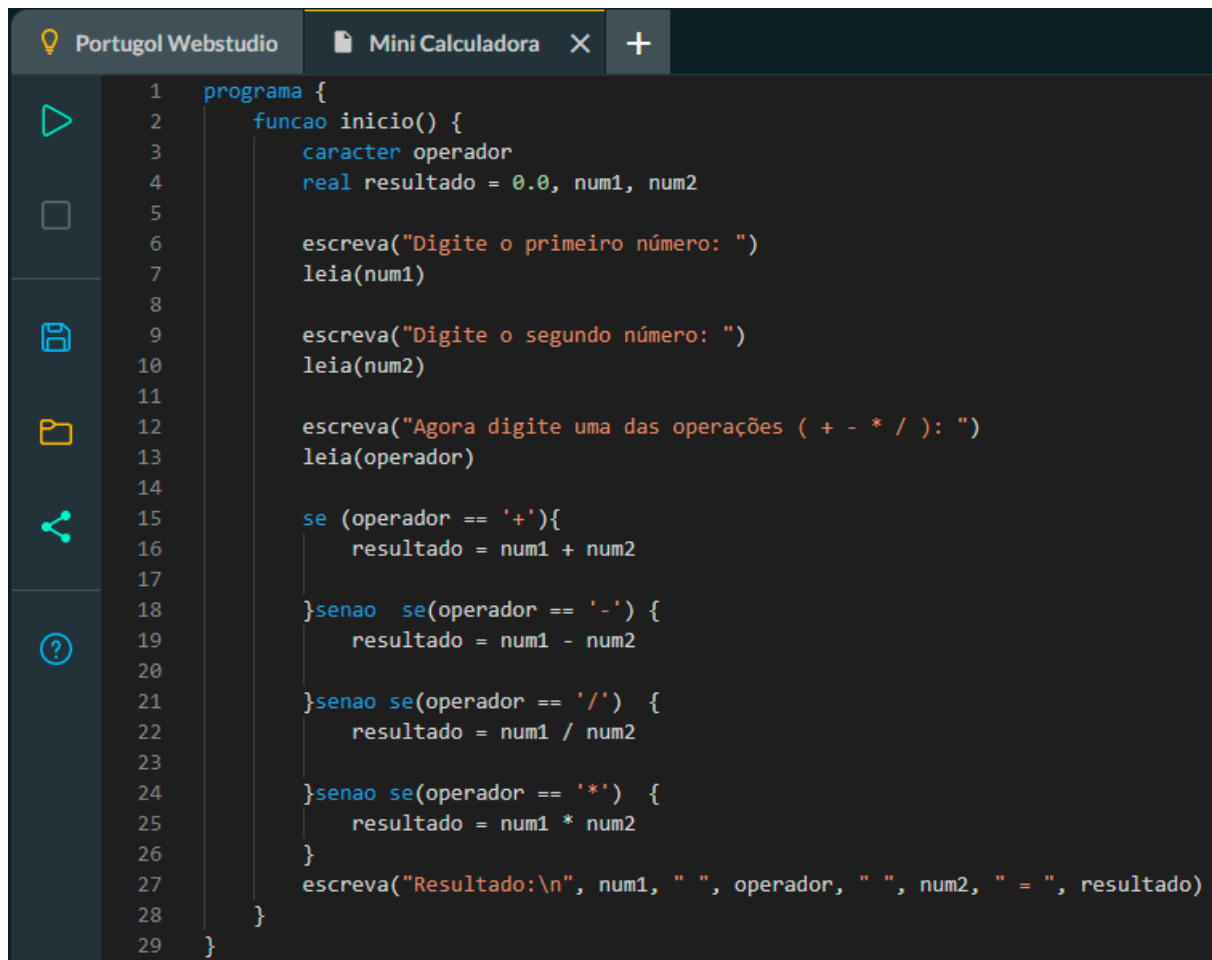
## Situação Problema 6:

Precisamos construir uma calculadora simples, que consiga efetuar as operações básicas de soma, subtração, multiplicação ou divisão, de acordo com a escolha do usuário. Para que a calculadora funcione, o usuário precisa obter os dois valores iniciais e a operação desejada. Vejamos como ilustrar essa situação em fluxograma:



Analisando o fluxograma, podemos perceber que existem 4 pontos onde o fluxo pode sofrer desvios, dependendo de qual operador o usuário vai escolher. Podemos deduzir também que, a cada execução do código, apenas um dos cálculos será executado, “ignorando” os demais.

Vejamos esse mesmo código em Portugal:

The image shows the Portugol Webstudio interface with a file named 'Mini Calculadora'. The code is written in Portugol and implements a simple calculator. It starts with a 'programa' block containing a 'funcao inicio()' block. Inside the function, it declares a 'caracter operador' and two 'real' numbers, 'num1' and 'num2', with 'num1' and 'num2' initialized to 0.0. The program then prompts the user to enter the first number, the second number, and the operator. It uses a series of 'se' (if) and 'senao' (else) statements to perform addition, subtraction, division, and multiplication based on the operator entered. Finally, it prints the result using 'escreva'.

Esse fluxograma ficou um pouco maior que os outros não é mesmo? Mas não se preocupe, pois utilizaremos fluxogramas e portugol somente nesta fase inicial, enquanto você aprende as estruturas. Agora, o mesmo código em JavaScript:

The image shows a snippet of JavaScript code that implements the same calculator logic as the Portugol code. It uses 'var' to declare and initialize 'operador', 'num1', 'num2', and 'operador' (re-declared). The logic is implemented using 'if', 'else if', and 'else' statements to handle the different operators. The result is logged to the console using 'console.log'.

# Exercícios

## IF ELSE IF - encadeamento

1. Você vai implementar uma melhoria em um sistema que consiste em dar dicas aos usuários sobre o que fazer durante cada tipo de clima.

Para isso cria uma variável chamada temperaturaAgora que receberá um dado do tipo numérico.

Caso a temperatura esteja acima de 20 graus ou mais, imprima a frase: “Não esqueça do filtro solar”

Para temperaturas entre 19 e 26, imprima: “Não esqueça seu óculos de sol”

para temperaturas abaixo de 19 imprima: “Já pegou seu casaco?”.

2. Precisamos criar um código que ajude uma escola a dizer se o aluno conseguiu atingir os objetivos na prova. E para isso eles deixaram para você as notas que eles usam para fazer essa classificação:

7 a 10 = “Aprovado”

5 a 6 = “Reforço”

Menos que 5 = “Recuperação”

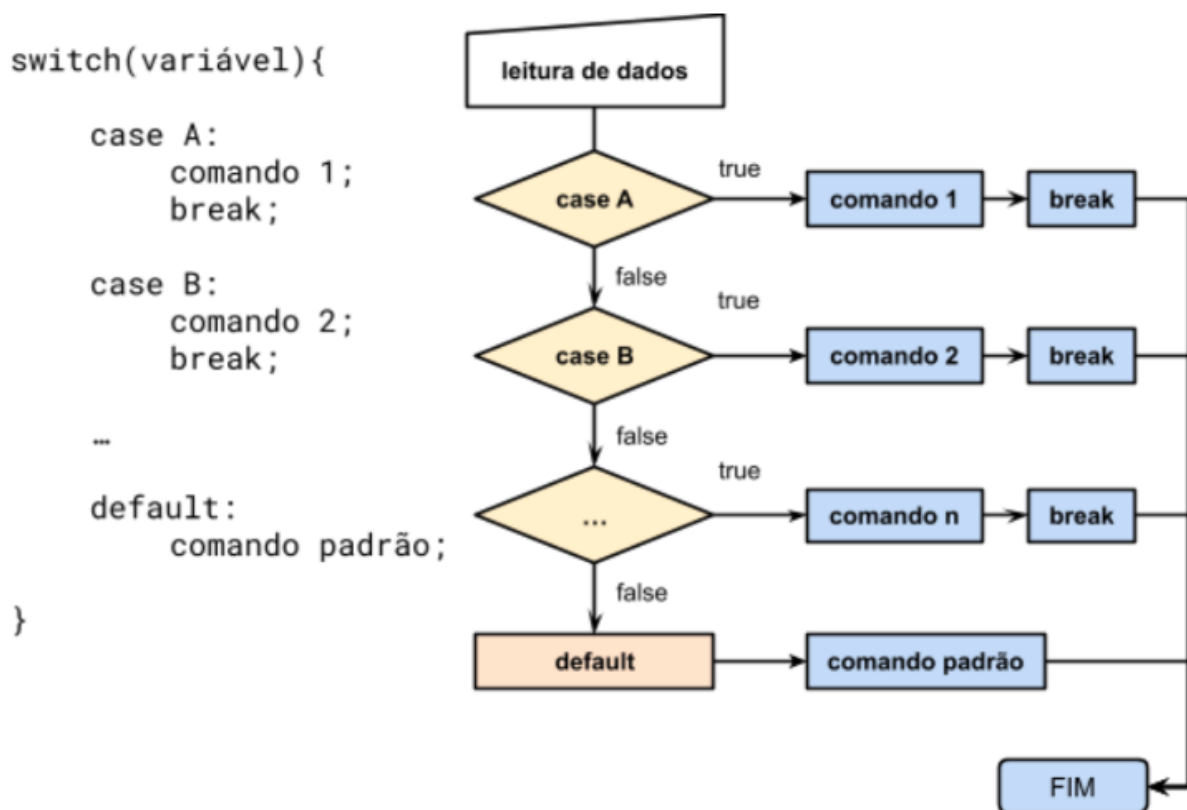
## Estruturas de Decisão

## Switch

### E se tivéssemos várias possibilidades?

Sabemos que não podemos sempre resumir nossos códigos em apenas duas possibilidades, e fazer isso utilizando as estruturas encadeadas IF ELSE IF, pode tornar a codificação um pouco complexa. Por isso temos o Switch que irá executar diferentes ações com base em diferentes condições (case).

Podemos ter uma ideia de como esse fluxo funciona em nossos sistemas com a figura a seguir:



O Switch possui um parâmetro que recebe a variável de referência e uma sequência de **cases**, onde cada um avalia se o seu valor é igual ao da variável passada. Se uma das

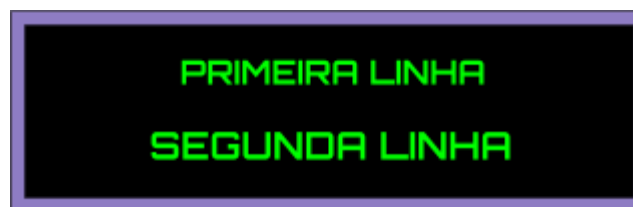
condições for correspondida, o resultado será verdadeiro e o código que está dentro dela será executado, mas caso o resultado seja falso, ele “pula” para o próximo case e analisa novamente a correspondência de valores. Pode acontecer de nenhum dos cases retornar o resultado verdadeiro, levando o fluxo do programa para a cláusula **Default (padrão)**, que é o último case a ser executado e depois encerra o fluxo do switch.

Um ponto importante: A instrução **break** associada a cada caso garante que o programa saia da condicional switch assim que a instrução correspondente for executada. Caso o break seja omitido, o programa continua a execução para a próxima instrução dentro do switch.

Vamos ver esse switch sendo utilizado para resolver um problema?

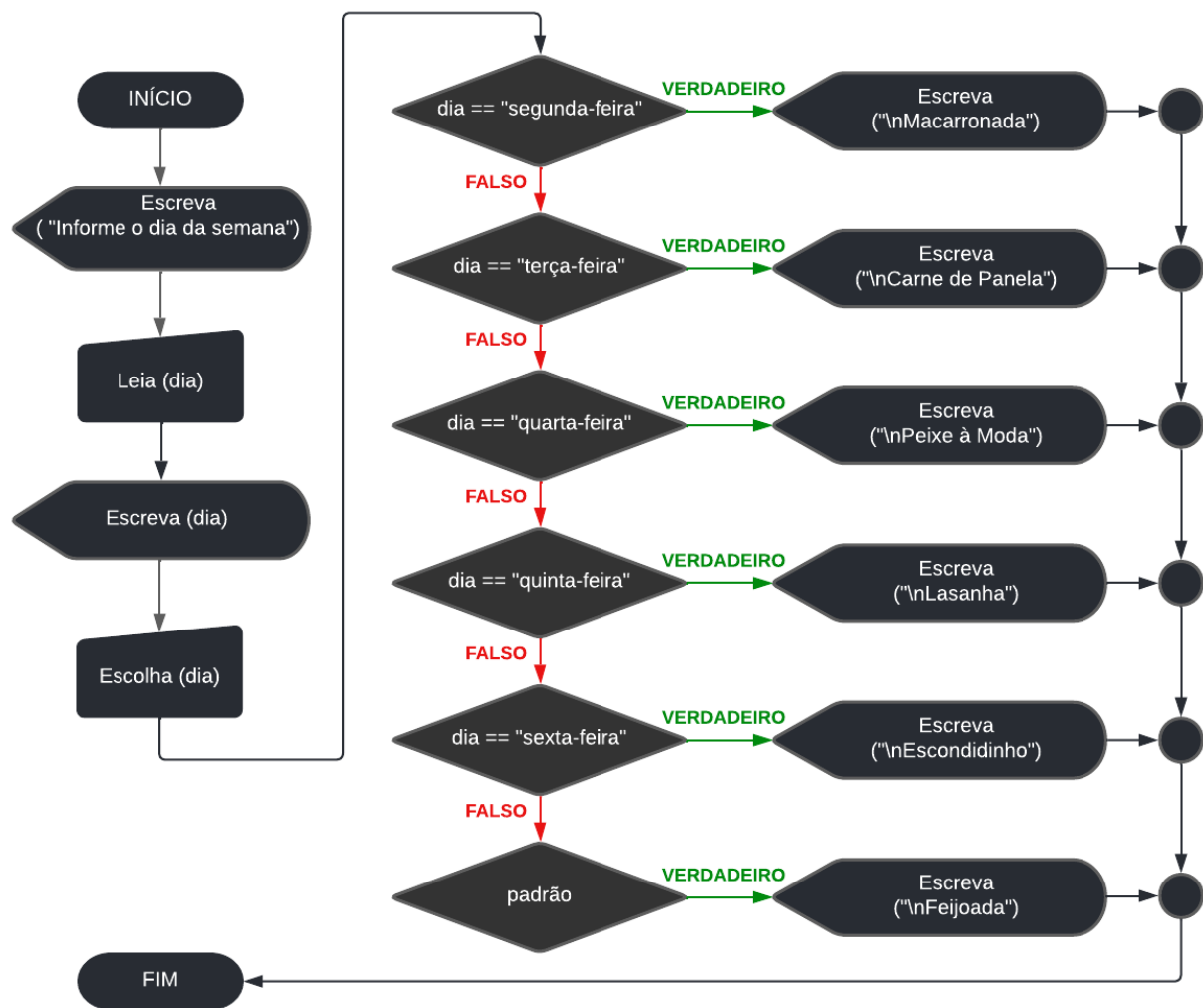
## Situação Problema 7:

Um restaurante da sua cidade comprou um painel digital para recepcionar seus clientes e informar qual o prato do dia, de acordo com o dia da semana! Todavia, o painel não veio programado e você foi contratado para criar um algoritmo que faça o painel funcionar de acordo com as regras a seguir:



- Na primeira linha deve ser exibido sempre o dia da semana;
- Na segunda linha, deve ser exibido o prato principal do dia como segue:
  - Toda **segunda-feira**, o prato principal será “**Macarronada**”;
  - Toda **terça-feira**, o prato principal será “**Carne de Panela**”;
  - Toda **quarta-feira**, o prato principal será “**Peixe à Moda**”;
  - Toda **quinta-feira**, o prato principal será “**Lasanha**”;
  - Toda **sexta-feira**, o prato principal será “**Escondidinho**”;
  - Aos **sábados** e **domingos** será “**Feijoada**”;

Montamos um fluxograma para ilustrar esse cenário:

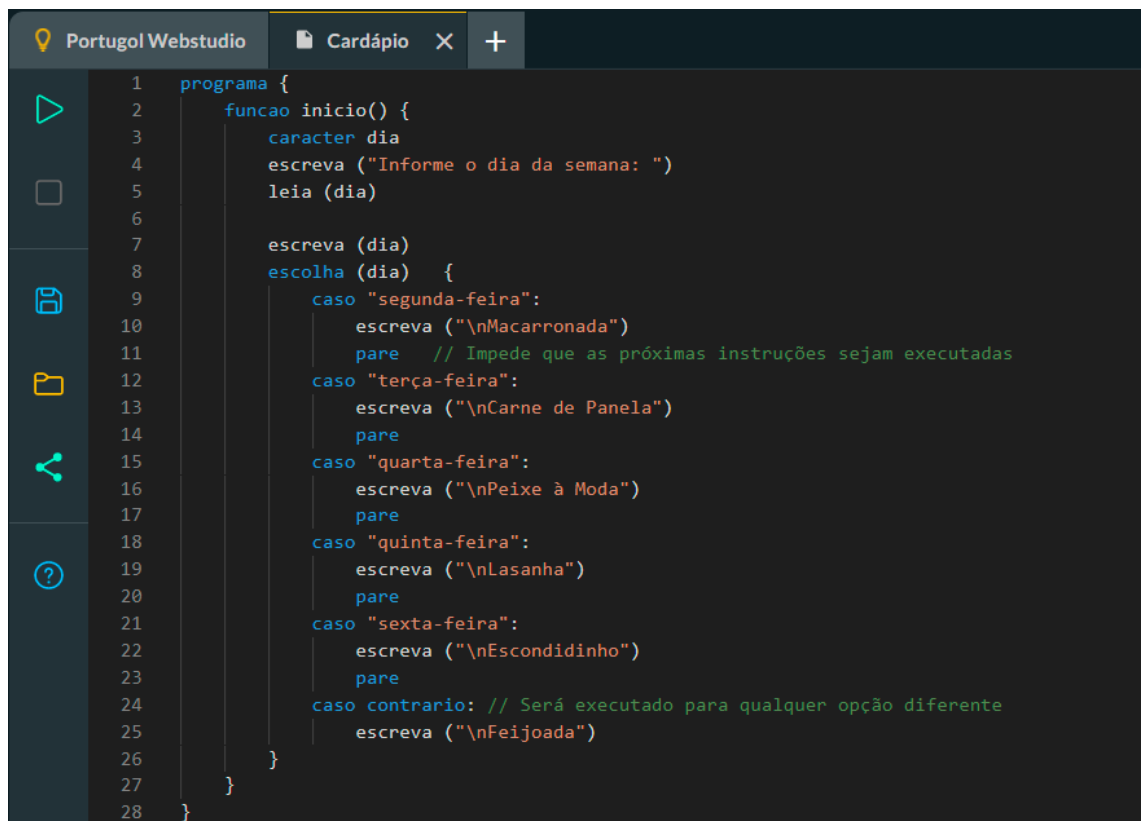


No modelo acima, vemos que após fazer a leitura do dia da semana, o programa imprime o dia na primeira linha e depois entra numa sequência de avaliações que, dependendo do dia escolhido, imprime o prato do dia!

Ao final de cada impressão, encontramos um conector que leva o fluxo para o final do programa.

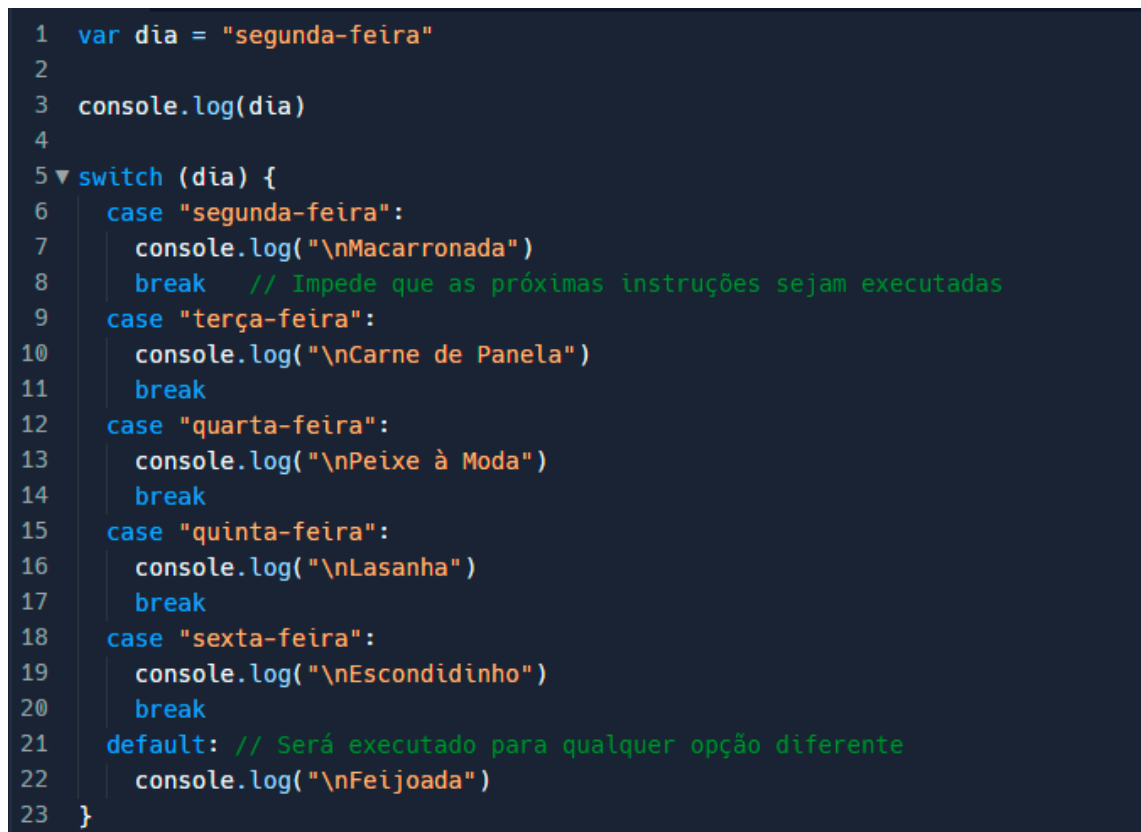
Quando trabalhamos com o Portugol, a estrutura é bem semelhante, porém, precisamos inserir ao final de cada caso um comando para “parar” o fluxo, caso contrário, o programa irá executar todas as opções disponíveis no código.

Vejamos um exemplo deste problema em Portugol:



```
1 programa {
2     funcao inicio() {
3         caracter dia
4         escreva ("Informe o dia da semana: ")
5         leia (dia)
6
7         escreva (dia)
8         escolha (dia) {
9             caso "segunda-feira":
10                escreva ("\nMacarronada")
11                pare // Impede que as próximas instruções sejam executadas
12            caso "terça-feira":
13                escreva ("\nCarne de Panela")
14                pare
15            caso "quarta-feira":
16                escreva ("\nPeixe à Moda")
17                pare
18            caso "quinta-feira":
19                escreva ("\nLasanha")
20                pare
21            caso "sexta-feira":
22                escreva ("\nEscondidinho")
23                pare
24            caso contrario: // Será executado para qualquer opção diferente
25                escreva ("\nFeijoadá")
26        }
27    }
28 }
```

Agora em Javascript, o algoritmo ficará assim:



```
1 var dia = "segunda-feira"
2
3 console.log(dia)
4
5 switch (dia) {
6     case "segunda-feira":
7         console.log("\nMacarronada")
8         break // Impede que as próximas instruções sejam executadas
9     case "terça-feira":
10        console.log("\nCarne de Panela")
11        break
12    case "quarta-feira":
13        console.log("\nPeixe à Moda")
14        break
15    case "quinta-feira":
16        console.log("\nLasanha")
17        break
18    case "sexta-feira":
19        console.log("\nEscondidinho")
20        break
21    default: // Será executado para qualquer opção diferente
22        console.log("\nFeijoadá")
23 }
```



# Exercícios

## SWITCH

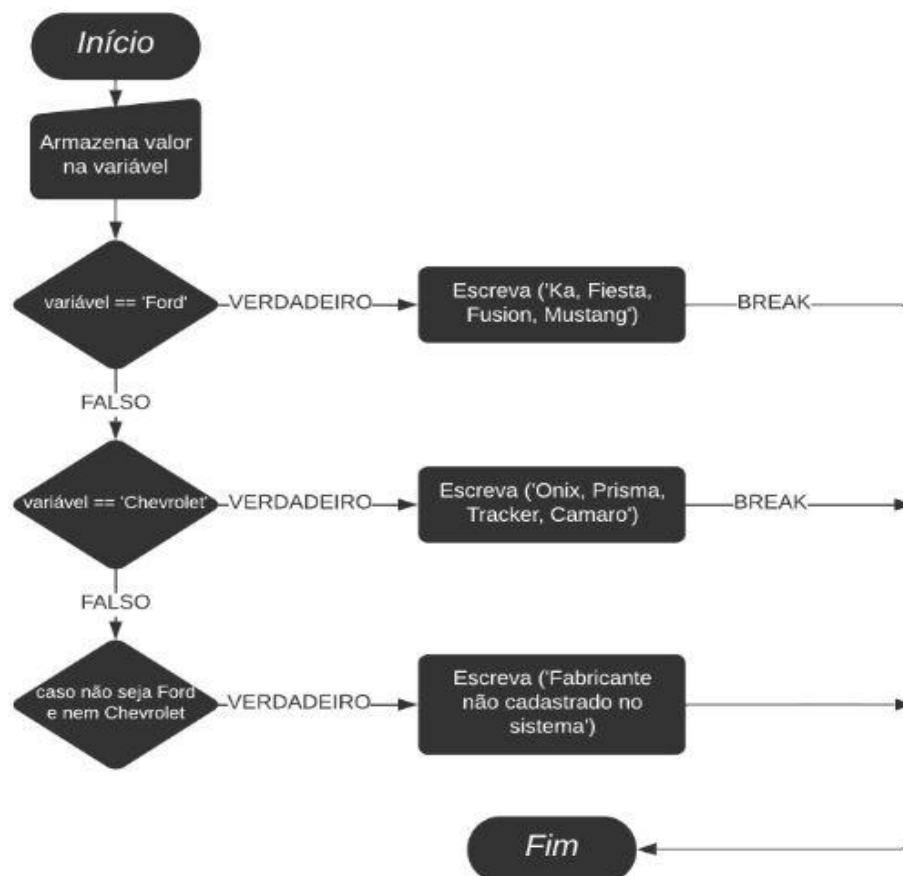
1. Vamos desenvolver um sistema onde aparecerá alguns carros de determinado fabricante à medida que o usuário o indica. Para isso crie uma variável chamada **fabricante** que receberá um dado do tipo “**string**” e sem seguida crie os seguintes casos:

**Ford** = imprima: “Ka, Fiesta, Fusion, Mustang”

**Chevrolet** = imprima: “Onix, Prisma, Tracker, Camaro”

**Qualquer outro fabricante** = imprima: “Fabricante não cadastrado no sistema”

Veremos primeiro a resolução dessa situação problema em algoritmo:



```
let fabricante = 'Ford'

switch (fabricante) {
  case 'Ford':
    console.log('Ka, Fiesta, Fusion, Mustang')
    break;
  case 'Chevrolet':
    console.log('Onix, Prisma, Tracker, Camaro')
    break;
  default:
    console.log('Fabricante não cadastrado no sistema')
}
```

2. Crie um código que mostre ao usuário diferentes frases ao longo da semana. Utilizando o Switch crie uma variável chamada **dia** que receberá um dado do tipo String referente aos dias da semana. Para o exercício , utilize os seguintes casos:

**segunda-feira** = “Tenha uma excelente semana!”

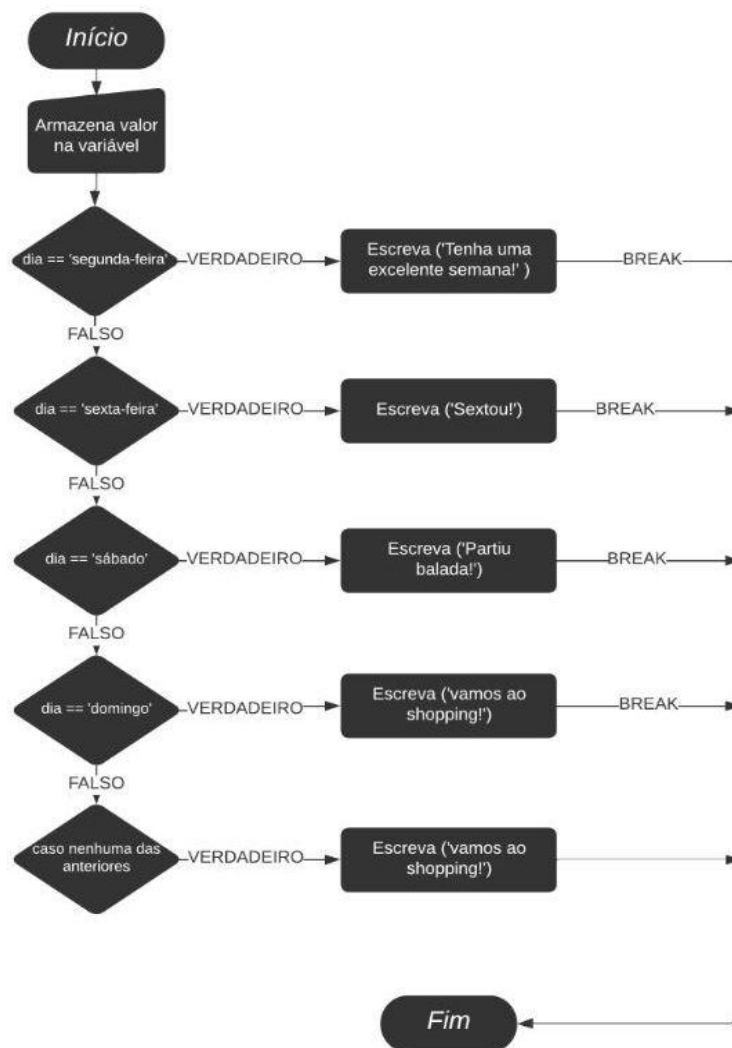
**sexta - feira** = “Sextou!”

**sábado** = “Partiu balada!”

**domingo** = “vamos ao shopping!”

**qualquer outro dia** = “ Bom dia!”

Em fluxograma a resolução ficará assim:



```

let dia = 'segunda-feira'
switch (dia) {
  case 'segunda-feira':
    console.log('Tenha uma excelente semana!')
    break;
  case 'sexta-feira':
    console.log('Sextou!')
    break;
  case 'sábado':
    console.log('Partiu balada!')
    break;
  case 'domingo':
    console.log('vamos ao shopping!')
    break;
  default:
    console.log('Bom dia!')
}

```