

## Aula 31 - Estruturas condicionais.

### 1 - Estruturas condicionais.

Olá a todos, sejam muito bem-vindos a mais uma aula. Dando continuidade ao nosso curso de python, na aula anterior nós vimos um pouco sobre estruturas de dados, que são maneiras como podemos armazenar informações no nosso algoritmo. Nesta aula vamos entender o que é uma estrutura condicional e para quê vamos usá-la.

Quando nós vamos usar uma estrutura condicional? Basicamente quando nós temos uma tomada de decisão ou melhor dizendo, uma condição para que uma determinada ação seja executada.

Então vamos supor, que estamos atravessando uma rua, e temos duas condições para sabermos se devemos ou não atravessar esta rua:

Primeiro:

caso o semáforo esteja verde:

então atravessamos a rua

caso não esteja verde:

não atravessamos a rua;

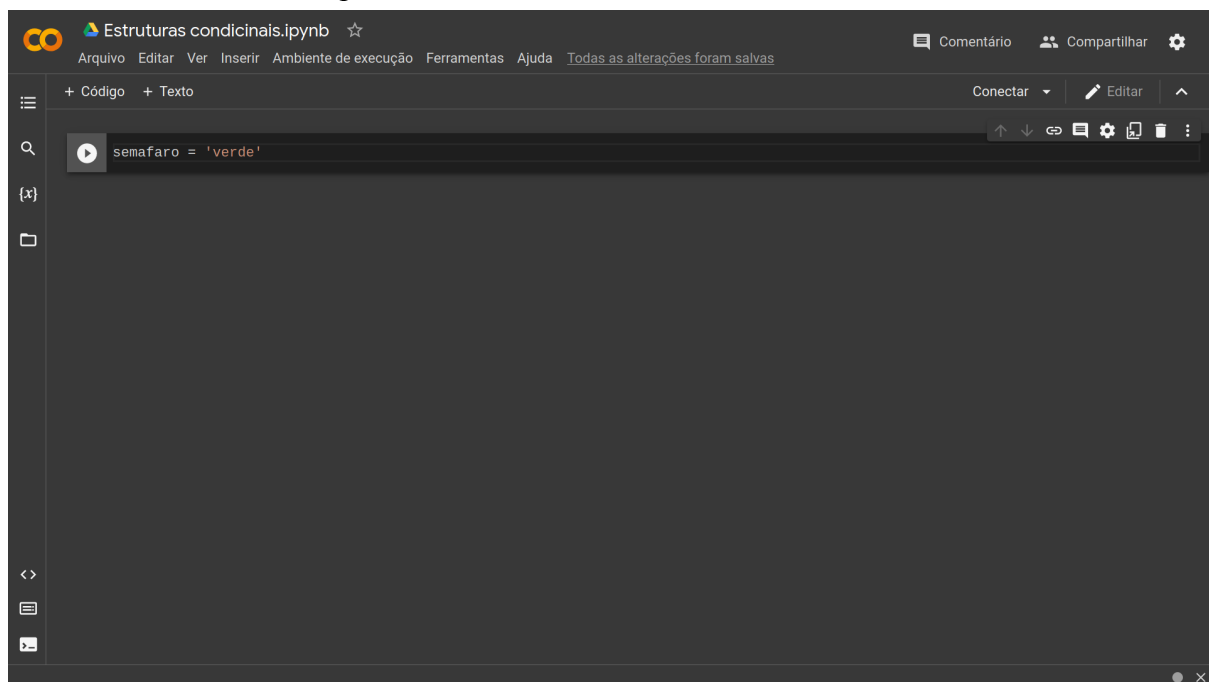
Podemos ver que é algo bem simples na verdade, uma tomada de decisão é algo que fazemos diariamente. Mas como podemos passar isso para o nosso algoritmo?

Na maioria das linguagens de programação usa-se o if e o else, que em uma tradução livre para o português, fica: se e se não:

Vamos ao google collab para entender melhor como representar isso na forma de algoritmo:

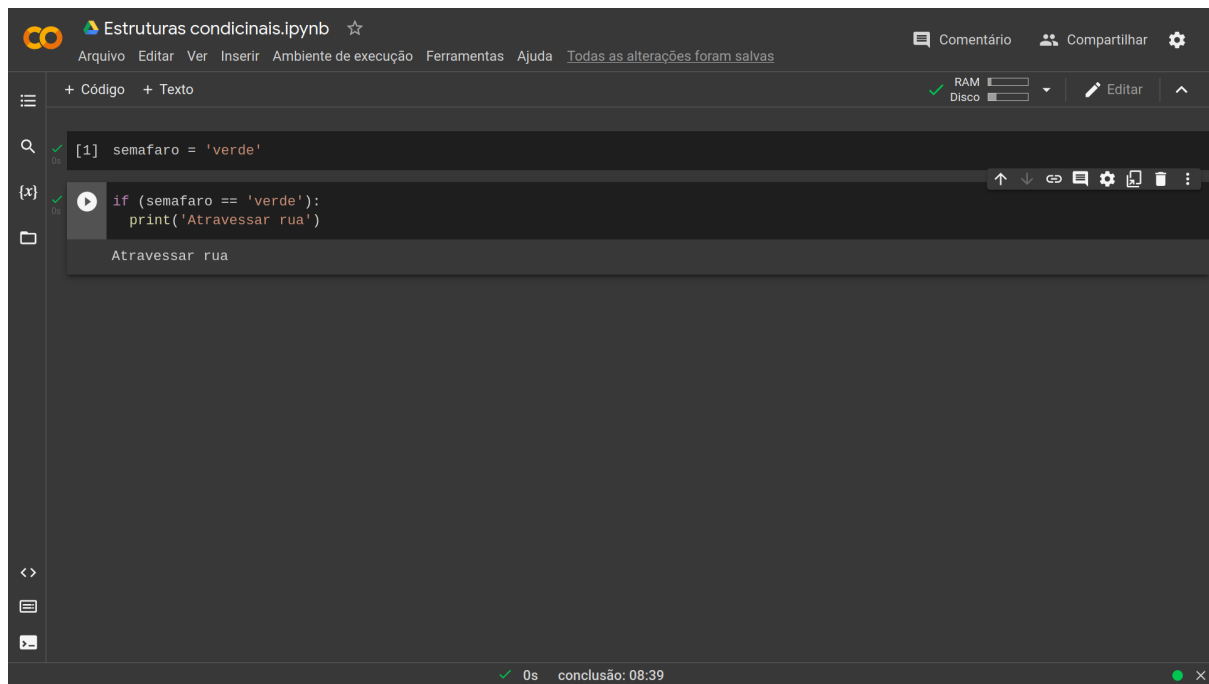
Vamos fazer o nosso algoritmo de atravessar a rua da seguinte maneira: Caso o farol esteja verde: então vamos exibir uma mensagem: Atravessando a rua... e caso não esteja verde: Esperando o farol abrir...

Vamos criar uma variável para armazenar o estado do nosso farol: semaforo.



```
semaforo = 'verde'
```

Agora vamos para a nossa estrutura do if e else:



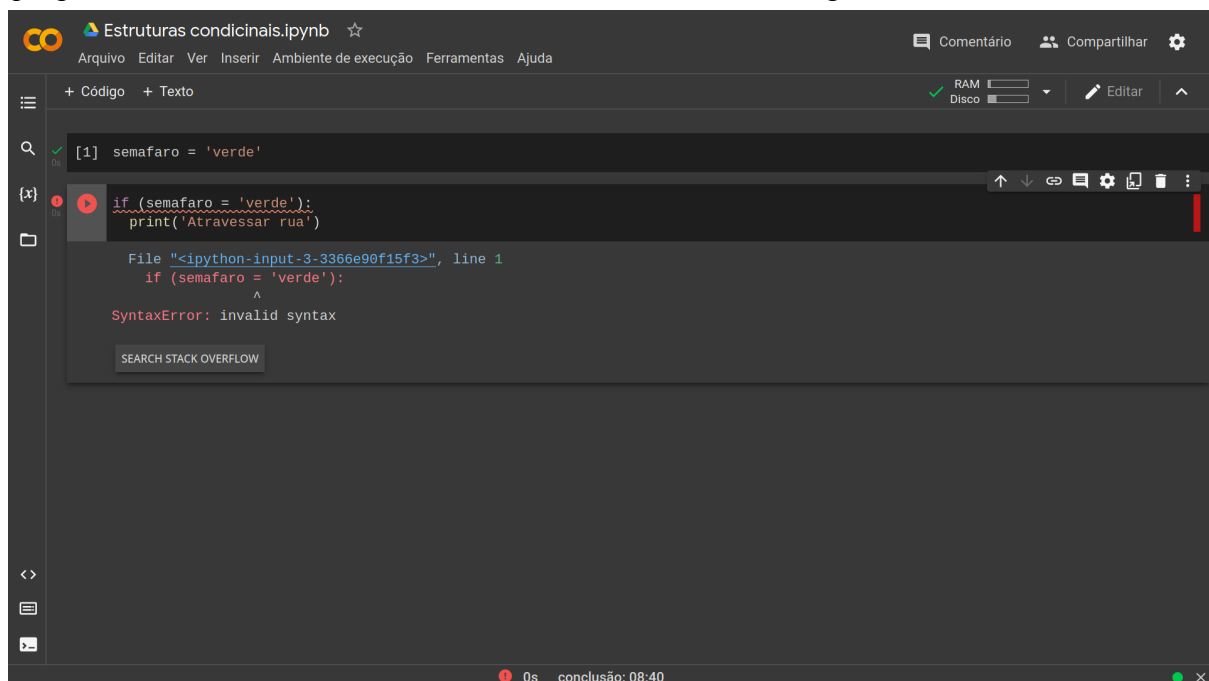
The screenshot shows a Jupyter Notebook interface with the title 'Estruturas condicionais.ipynb'. The top bar includes a menu with 'Arquivo', 'Editar', 'Ver', 'Inserir', 'Ambiente de execução', 'Ferramentas', and 'Ajuda', along with a status message 'Todas as alterações foram salvas'. On the right, there are buttons for 'Comentário', 'Compartilhar', and a settings icon. Below the menu, there are tabs for '+ Código' and '+ Texto'. The main area displays a code cell with the following Python code: 

```
[1] semafaro = 'verde'

if (semafaro == 'verde'):
    print('Atravessar rua')
```

 The output of the cell is 'Atravessar rua'. The status bar at the bottom indicates '0s' and 'conclusão: 08:39'.

Aqui está o nosso primeiro if. Aqui temos várias coisas a pontuar: Primeiro, é possível notar que para verificar se o sinal está verde, foi usado dois sinais de igual:



The screenshot shows the same Jupyter Notebook interface, but the code cell now contains a syntax error. The code is: 

```
[1] semafaro = 'verde'

if (semafaro = 'verde'):
    print('Atravessar rua')
```

 The output area shows an error message: 

```
File "<ipython-input-3-3366e90f15f3>", line 1
    if (semafaro = 'verde'):
                ^
SyntaxError: invalid syntax
```

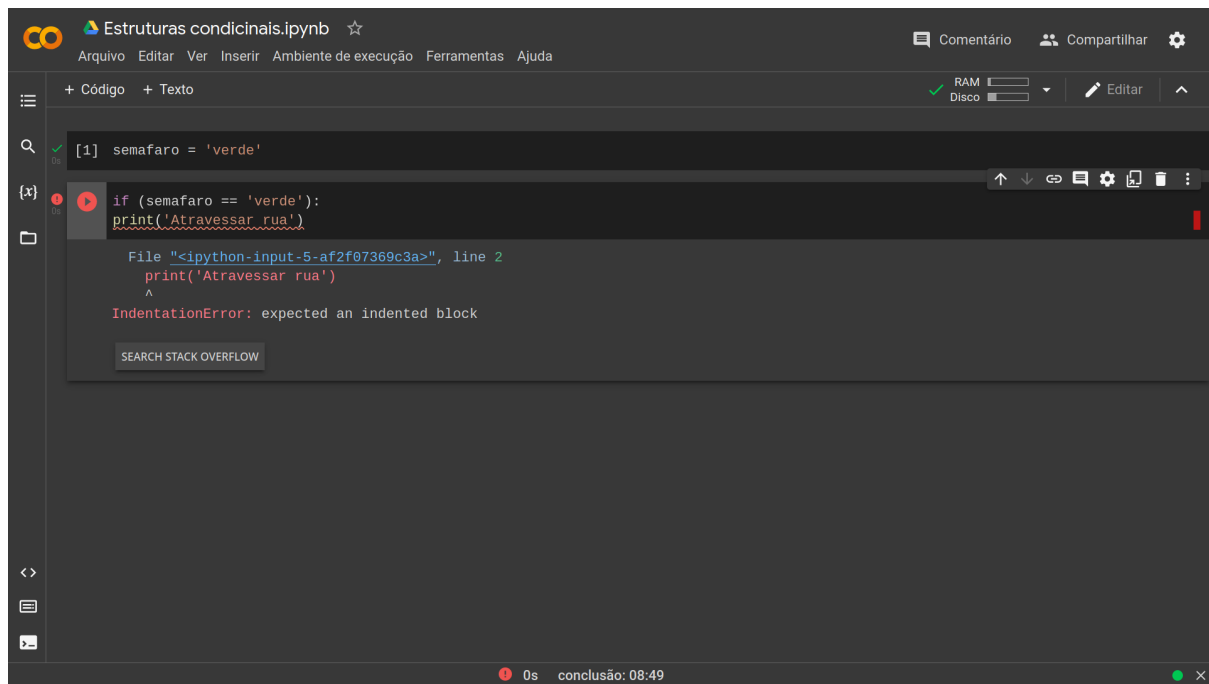
 Below the error message is a button that says 'SEARCH STACK OVERFLOW'. The status bar at the bottom indicates '0s' and 'conclusão: 08:40'.

Note que ocorrerá um erro caso coloque apenas um sinal: Isso acontece pelo seguinte. Na linguagem de programação python, apenas o sinal de igual é usado para atribuir um valor, igual quando fazemos para as nossas variáveis. Por exemplo, na nossa variável: semafaro, a ela está sendo atribuído o valor 'verde'. Já os dois sinais de igual servem para estabelecer uma comparação entre dois valores. Basicamente estamos perguntando se um valor é igual ao outro. Já vimos isso em aulas anteriores.

Outro aspecto muito importante que devemos fazer é a indentação do nosso código

Mas o que vem ser exatamente a indentação do código?

Vamos reescrever o código sem a indentação para ficar mais claro:



The screenshot shows a Jupyter Notebook interface with the title "Estruturas condicionais.ipynb". The code cell contains the following Python code:

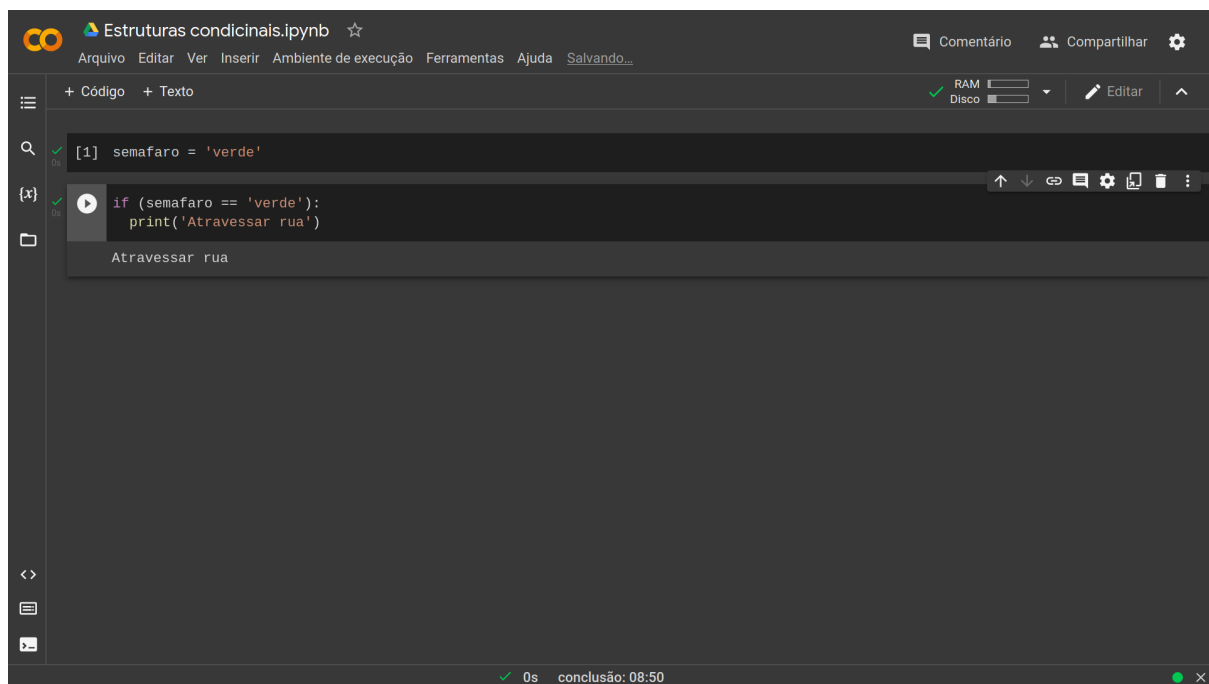
```
[1] semafaro = 'verde'

if (semaforo == 'verde'):
print('Atravessar rua')
```

The code is executed, and an error message is displayed:

```
File "<ipython-input-5-af2f07369c3a>", line 2
    print('Atravessar rua')
    ^
IndentationError: expected an indented block
```

The error message indicates that the code is missing an indentation for the print statement. The status bar at the bottom shows "0s" and "conclusão: 08:49".



The screenshot shows the same Jupyter Notebook interface, but the code has been corrected to use proper indentation (4 spaces) for the print statement:

```
[1] semafaro = 'verde'

if (semaforo == 'verde'):
    print('Atravessar rua')
```

The code is executed, and the output "Atravessar rua" is displayed. The status bar at the bottom shows "0s" and "conclusão: 08:50".

É possível notar alguma diferença entre os códigos? Bem, no primeiro podemos notar que a linha onde está o print está com um espaço a mais. Isso é o que chamamos de indentação do código, não o espaço a mais, mas esse espaço que geralmente fazemos com a tecla Tab, mas isso serve para representar que a instrução print está dentro do bloco do if: Especialmente no Python, é fundamental que nos atentemos à indentação do nosso código, pois diferente de outras linguagens, o python não utiliza chaves para fazer a separação de seus blocos:

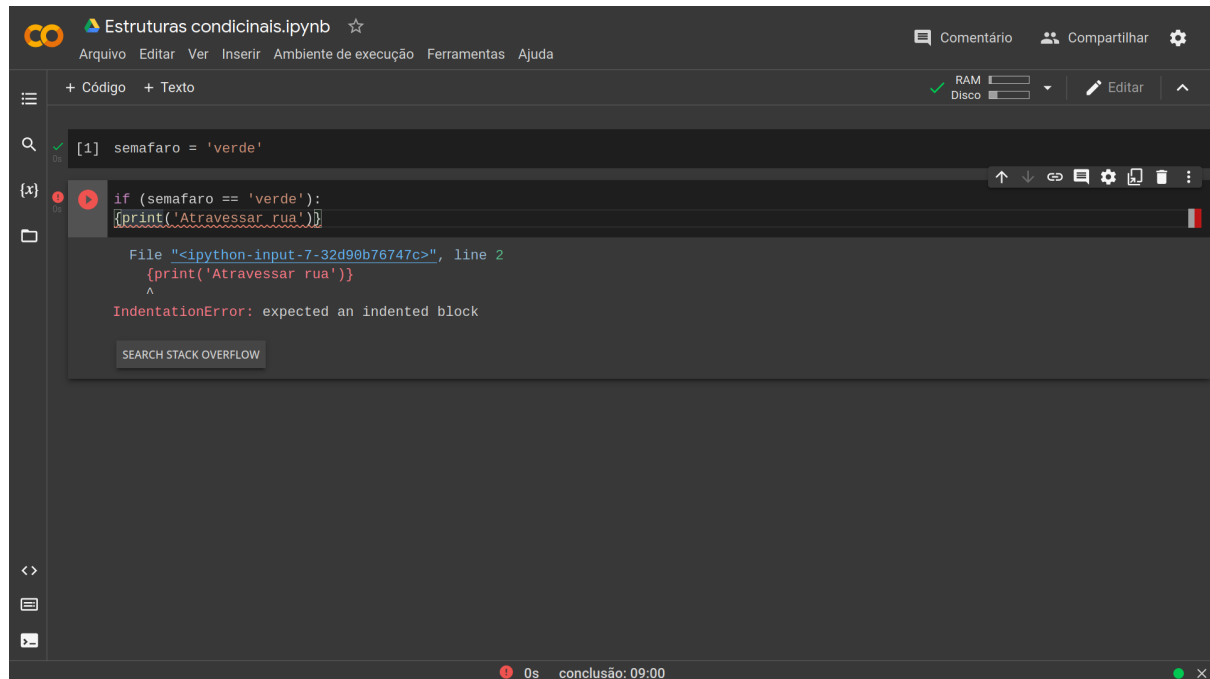
Em outras linguagens teríamos por exemplo

```
If (condição) {
    código
}
```

Ou seja, o que está dentro das chaves é o que será executado dentro da condição. Então em uma linguagem este mesmo código poderia ser escrito até desta maneira:

if (condição) {código}

Mas o Python, uma vez que não utiliza chaves para separação de código, é obrigatório que esteja bem indentado para que o código rode sem problemas.



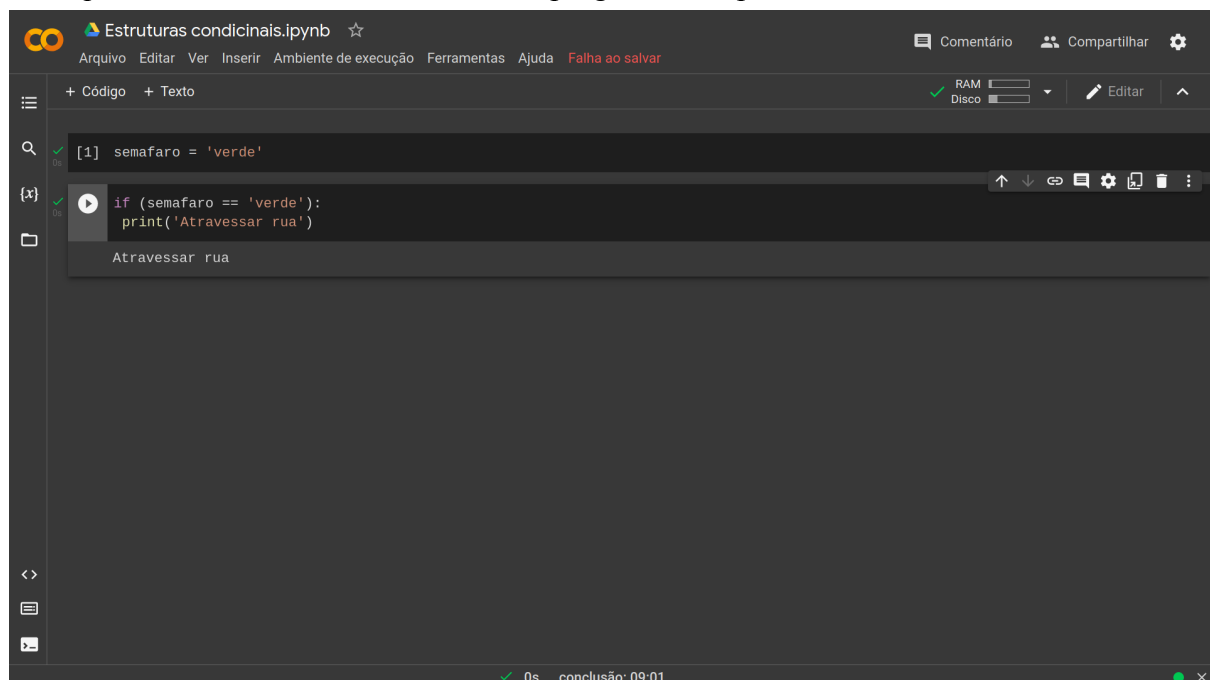
The screenshot shows a Jupyter Notebook interface with the title "Estruturas condicionais.ipynb". The code cell contains the following Python code:

```
[1] semaforo = 'verde'

if (semaforo == 'verde'):
  {print('Atravessar rua')}
```

The code is executed, but an error is shown: "IndentationError: expected an indented block". The error message points to the line containing the curly braces. The notebook interface includes a menu bar with options like "Arquivo", "Editar", "Ver", "Inserir", "Ambiente de execução", "Ferramentas", and "Ajuda". The status bar at the bottom shows "0s" and "conclusão: 09:00".

Note que se tentarmos utilizar as chaves o programa irá apresentar um erro.



The screenshot shows the same Jupyter Notebook interface, but the code is now correctly indented using spaces. The code cell contains the following Python code:

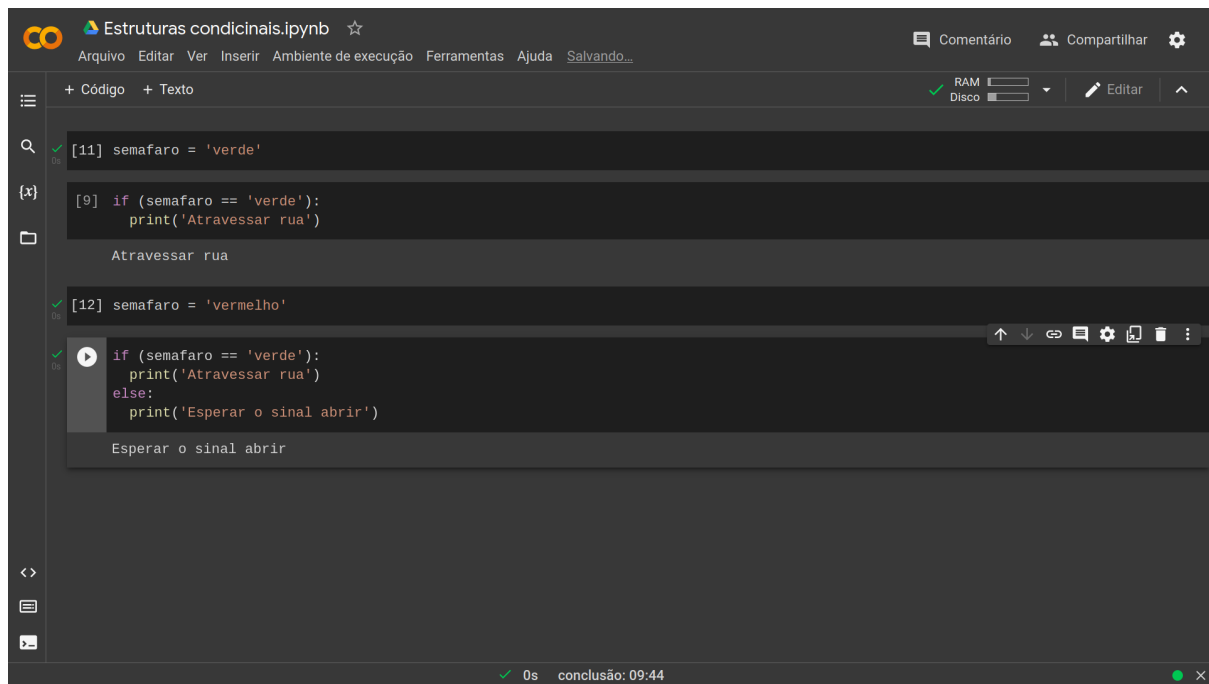
```
[1] semaforo = 'verde'

if (semaforo == 'verde'):
    print('Atravessar rua')
```

The code is executed successfully, and the output "Atravessar rua" is displayed. The notebook interface includes the same menu bar and status bar as the previous screenshot. The status bar at the bottom shows "0s" and "conclusão: 09:01".

Note também, que a gente pode utilizar um espaço apenas que o Python já irá entender que aquela linha de código pertence ao bloco if. Mas é mais recomendável que usemos o tab para fazer as indentações.

Continuando então, vamo fazer a mensagem que será mostrada caso o sinal não esteja verde:



```
[11] semaforo = 'verde'

[9] if (semaforo == 'verde'):
    print('Atravessar rua')

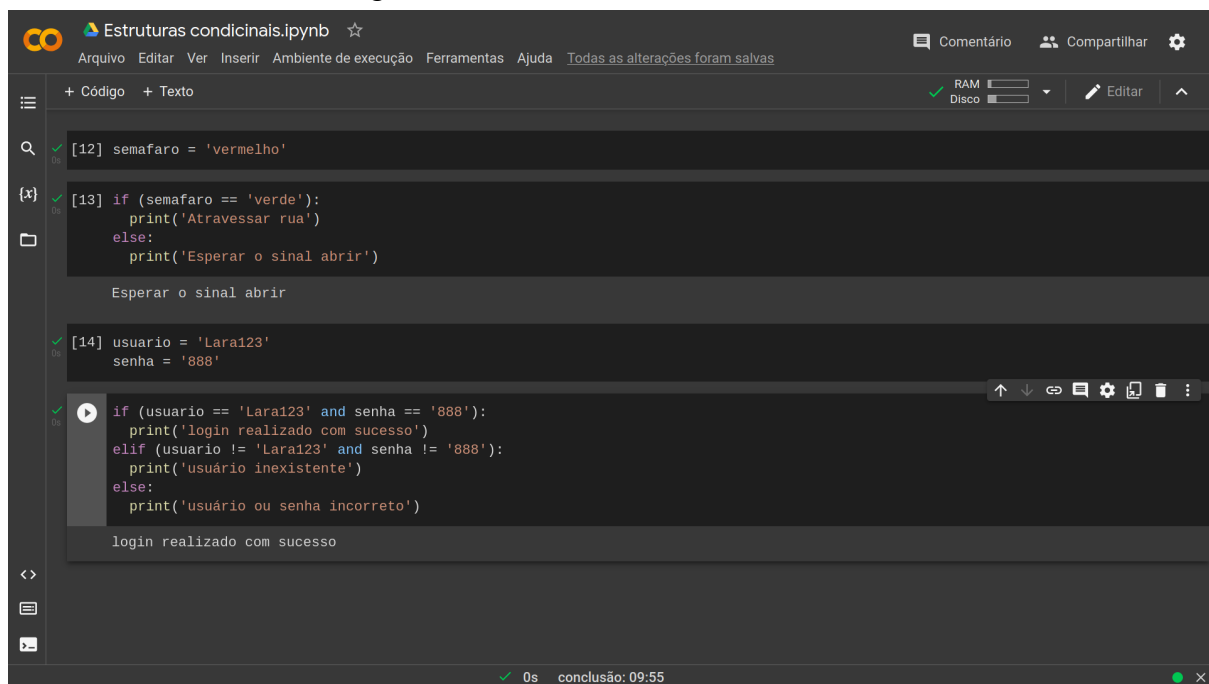
Atravessar rua

[12] semaforo = 'vermelho'

if (semaforo == 'verde'):
    print('Atravessar rua')
else:
    print('Esperar o sinal abrir')

Esperar o sinal abrir
```

Vamos fazer um programa diferente agora: vamos fazer duas variáveis usuário e senha. E caso o usuário e senha estiverem corretos vamos mostrar a mensagem: login realizado com sucesso, caso usuário ou senha estiverem incorretos, vamos mostrar a mensagem: usuário ou senha incorreto. E caso nem a senha e nem o usuário correspondam aos dados de login, então será mostrada a mensagem: Usuário inexistente:



```
[12] semaforo = 'vermelho'

[13] if (semaforo == 'verde'):
    print('Atravessar rua')
else:
    print('Esperar o sinal abrir')

Esperar o sinal abrir

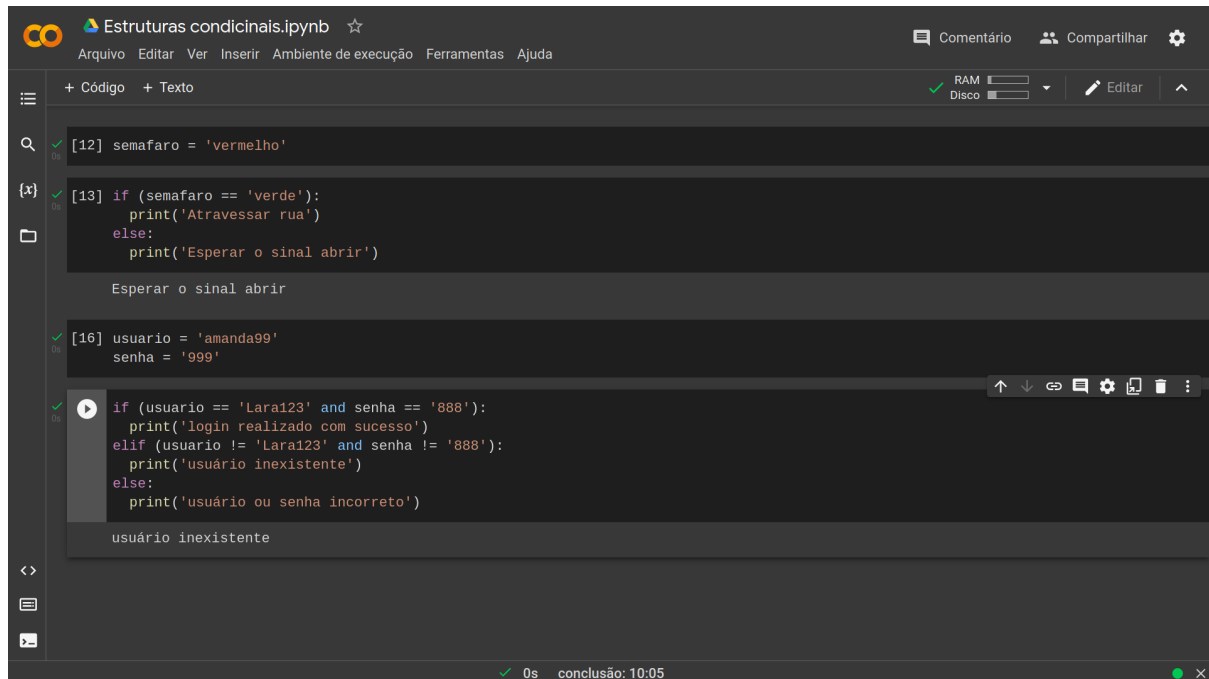
[14] usuario = 'Lara123'
    senha = '888'

if (usuario == 'Lara123' and senha == '888'):
    print('login realizado com sucesso')
elif (usuario != 'Lara123' and senha != '888'):
    print('usuário inexistente')
else:
    print('usuário ou senha incorreto')

login realizado com sucesso
```

Notem agora que temos um código um pouco mais complexo, e duas palavras que não tínhamos utilizado antes, o and e o elif. O and é um operador lógico matemático. O and serve para que somente entre no if, caso as duas condições sejam verdadeiras, no caso, se o login e a senha forem verdadeiros, então a mensagem será mostrada: login realizado com sucesso.

O elif é um else que desejamos passar uma condição. O sinal != serve para perguntar ao algoritmo se dois valores são distintos: Então a terceira linha elif serve para verificar se caso o usuário e a senha forem diferentes, então será mostrada a mensagem: usuário inexistente. e o ultimo else serve para caso não caia em nenhuma das condições anteriores. Ou seja, se o usuário e a senha não são iguais, mas os dois também não são diferentes, significa que pelo menos um é igual, ou seja: mostra a mensagem: usuário ou senha incorreto.



```
[12] semafaro = 'vermelho'
```

```
[13] if (semafaro == 'verde'):
      print('Atravessar rua')
      else:
          print('Esperar o sinal abrir')

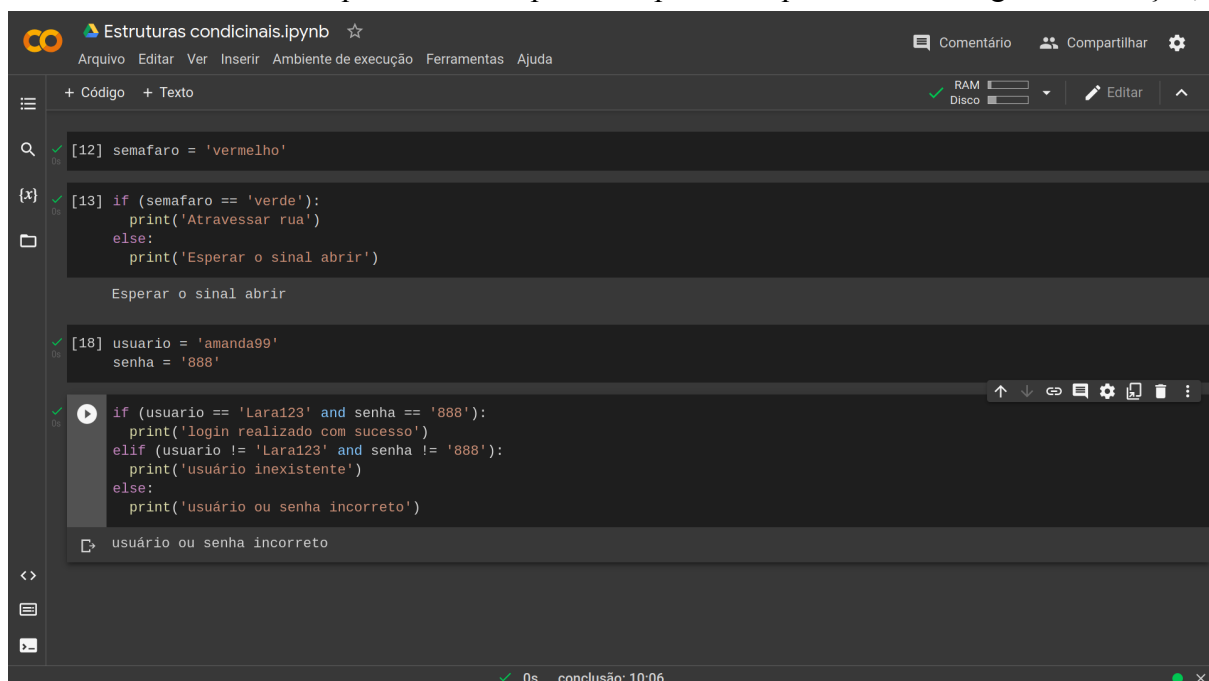
      Esperar o sinal abrir
```

```
[16] usuario = 'amanda99'
      senha = '999'
```

```
if (usuario == 'Lara123' and senha == '888'):
    print('login realizado com sucesso')
elif (usuario != 'Lara123' and senha != '888'):
    print('usuário inexistente')
else:
    print('usuário ou senha incorreto')

usuário inexistente
```

Mudando tanto o usuário quanto a senha podemos perceber que caímos na segunda condição;



```
[12] semafaro = 'vermelho'
```

```
[13] if (semafaro == 'verde'):
      print('Atravessar rua')
      else:
          print('Esperar o sinal abrir')

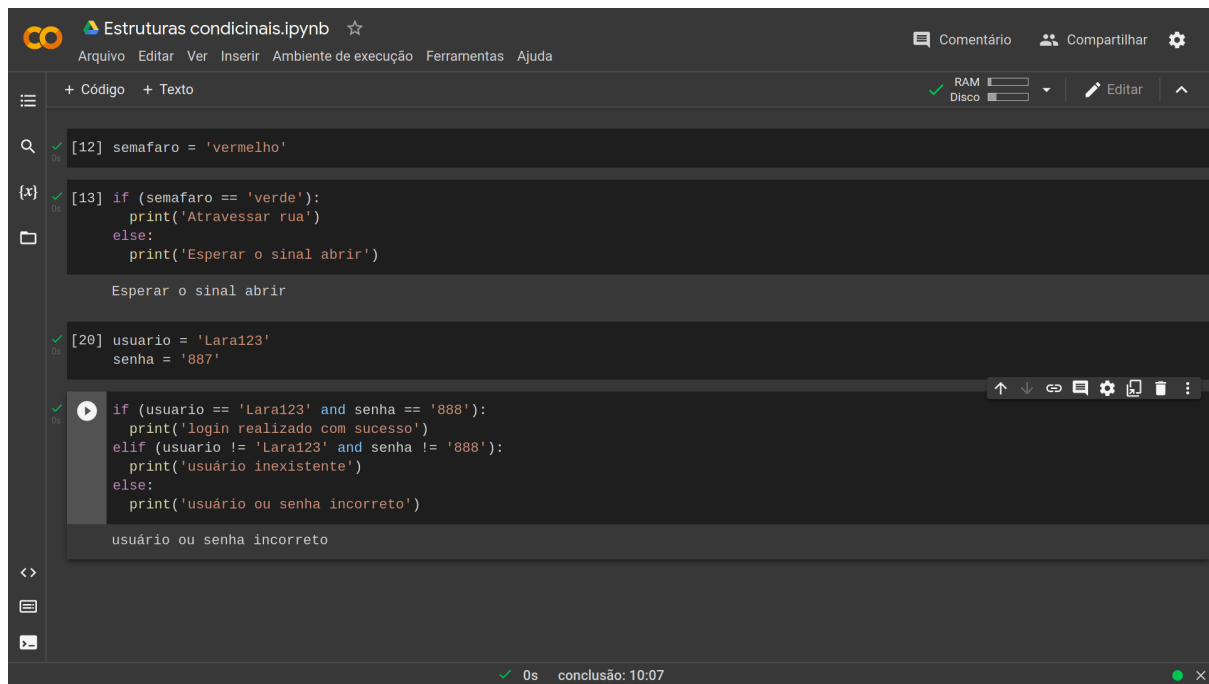
      Esperar o sinal abrir
```

```
[18] usuario = 'amanda99'
      senha = '888'
```

```
if (usuario == 'Lara123' and senha == '888'):
    print('login realizado com sucesso')
elif (usuario != 'Lara123' and senha != '888'):
    print('usuário inexistente')
else:
    print('usuário ou senha incorreto')

usuário ou senha incorreto
```

caso a senha ou o usuário estiver no nosso login caímos na terceira condição:



```
[12] semafaro = 'vermelho'

[13] if (semaforo == 'verde'):
    print('Atravessar rua')
    else:
        print('Esperar o sinal abrir')

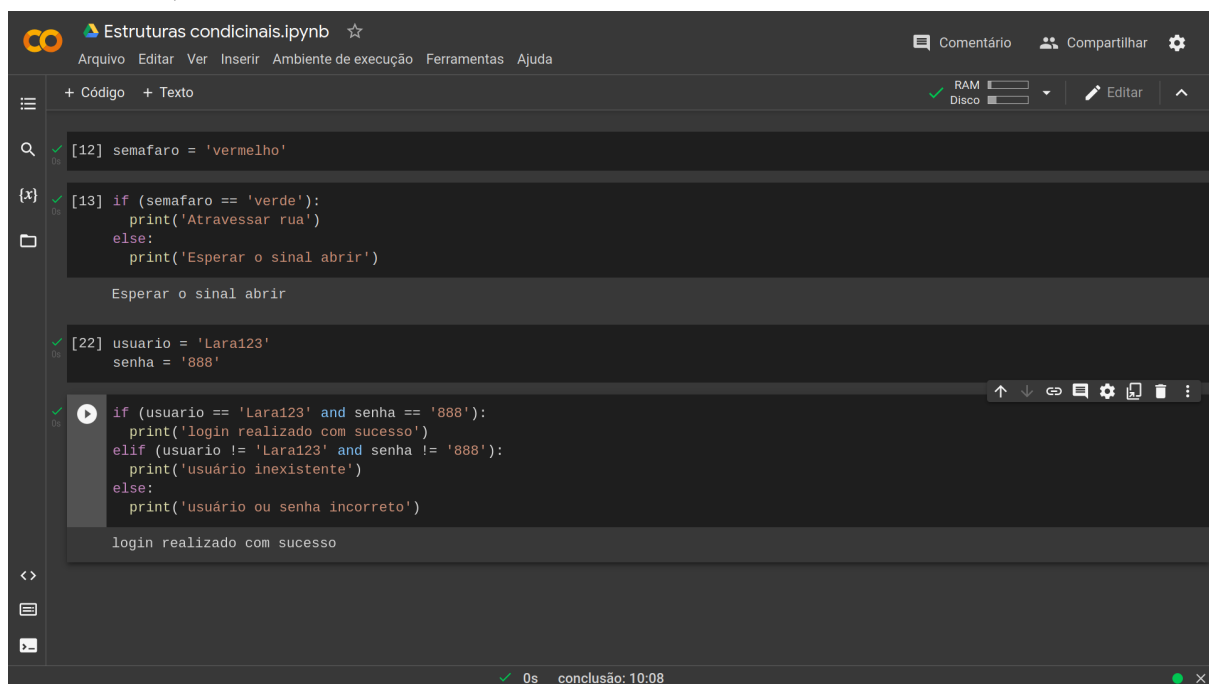
Esperar o sinal abrir

[20] usuario = 'Lara123'
    senha = '887'

if (usuario == 'Lara123' and senha == '888'):
    print('login realizado com sucesso')
elif (usuario != 'Lara123' and senha != '888'):
    print('usuário inexistente')
else:
    print('usuário ou senha incorreto')

usuário ou senha incorreto
```

E como visto, se o usuário e senha estiverem corretos então:



```
[12] semafaro = 'vermelho'

[13] if (semaforo == 'verde'):
    print('Atravessar rua')
    else:
        print('Esperar o sinal abrir')

Esperar o sinal abrir

[22] usuario = 'Lara123'
    senha = '888'

if (usuario == 'Lara123' and senha == '888'):
    print('login realizado com sucesso')
elif (usuario != 'Lara123' and senha != '888'):
    print('usuário inexistente')
else:
    print('usuário ou senha incorreto')

login realizado com sucesso
```

Mas por esta aula é isto, vimos um pouco como funcionam estruturas condicionais. E como podemos utilizá-las para tomada de decisão dentro do código. Na próxima aula, vamos entender como funcionam as estruturas de repetição. Muito obrigado por assistir até aqui e até a próxima aula.