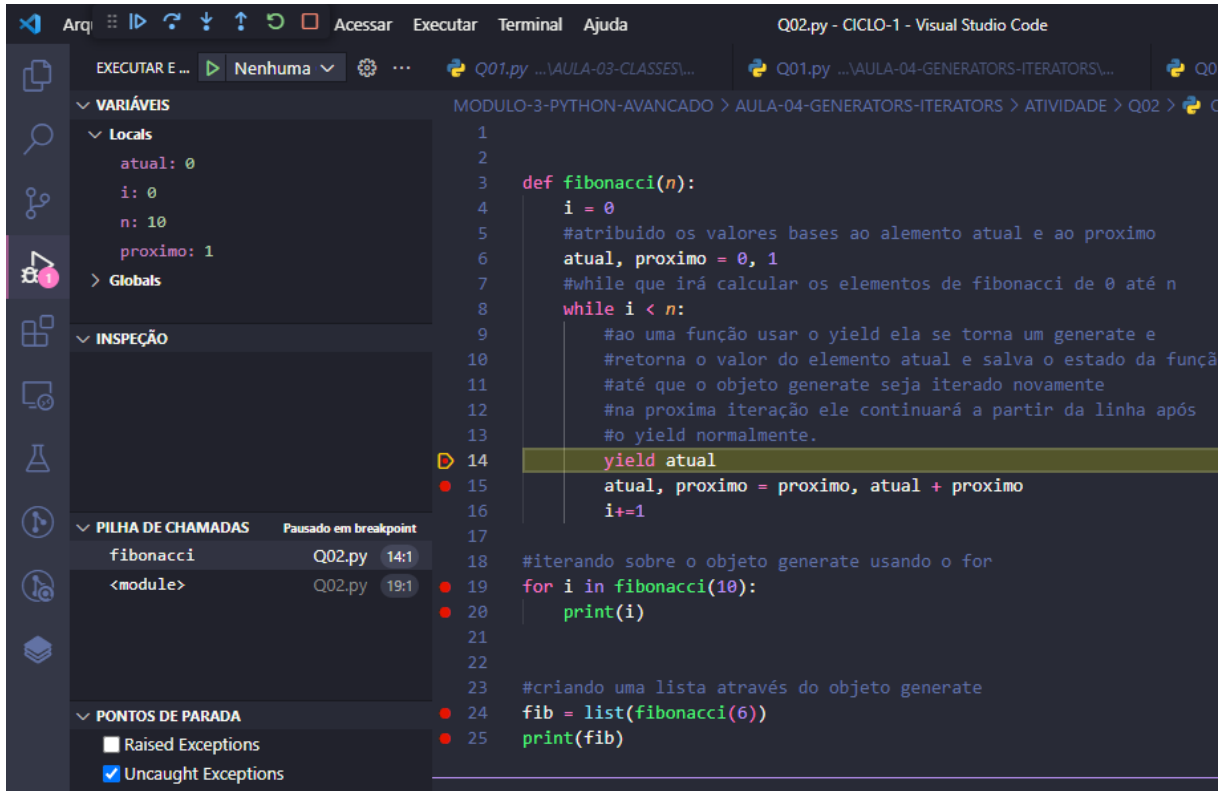


Python Avançado - Iterators and Generators - Aula 03 - Questão 2

Aluno:

Francisco Ruben Vasconcelos Freitas

Tela de execução do programa fibonacci usando generate na primeira iteração



```
1
2
3 def fibonacci(n):
4     i = 0
5     #atribuido os valores bases ao elemento atual e ao proximo
6     atual, proximo = 0, 1
7     #while que irá calcular os elementos de fibonacci de 0 até n
8     while i < n:
9         #ao uma função usar o yield ela se torna um generate e
10        #retorna o valor do elemento atual e salva o estado da função
11        #até que o objeto generate seja iterado novamente
12        #na proxima iteração ele continuará a partir da linha após
13        #o yield normalmente.
14        yield atual
15        atual, proximo = proximo, atual + proximo
16        i+=1
17
18 #iterando sobre o objeto generate usando o for
19 for i in fibonacci(10):
20     print(i)
21
22
23 #criando uma lista através do objeto generate
24 fib = list(fibonacci(6))
25 print(fib)
```

VARIÁVEIS

- Locals
 - atual: 0
 - i: 0
 - n: 10
 - proximo: 1
- Globals

INSPEÇÃO

PILHA DE CHAMADAS Pausado em breakpoint

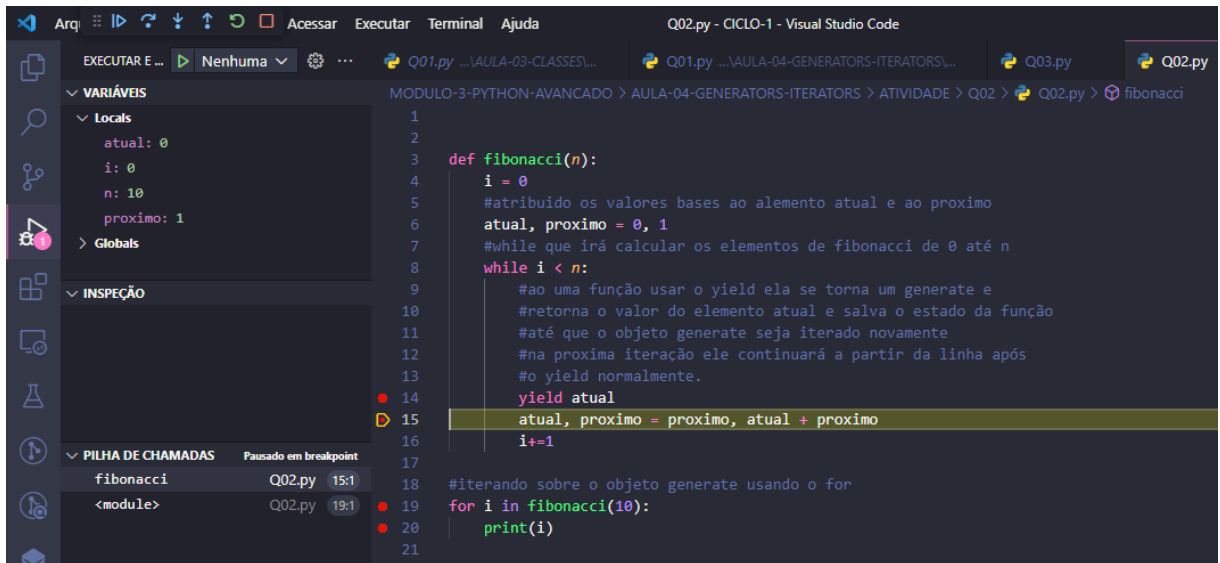
- fibonacci Q02.py 14:1
- <module> Q02.py 19:1

PONTOS DE PARADA

- ☐ Raised Exceptions
- ☒ Uncaught Exceptions

Podemos ver que na primeira iteração do generate é retornado o valor atual e o estado é salvo em memória.

Tela de execução do programa fibonacci usando generate na segunda iteração



```
1
2
3 def fibonacci(n):
4     i = 0
5     #atribuido os valores bases ao elemento atual e ao proximo
6     atual, proximo = 0, 1
7     #while que irá calcular os elementos de fibonacci de 0 até n
8     while i < n:
9         #ao uma função usar o yield ela se torna um generate e
10        #retorna o valor do elemento atual e salva o estado da função
11        #até que o objeto generate seja iterado novamente
12        #na próxima iteração ele continuará a partir da linha após
13        #o yield normalmente.
14        yield atual
15        atual, proximo = proximo, atual + proximo
16        i+=1
17
18 #iterando sobre o objeto generate usando o for
19 for i in fibonacci(10):
20     print(i)
21
```

VARIÁVEIS

- Locals
 - atual: 0
 - i: 0
 - n: 10
 - proximo: 1
- Globals

INSPEÇÃO

PILHA DE CHAMADAS Pausado em breakpoint

- fibonacci Q02.py 15:1
- <module> Q02.py 19:1

Podemos ver que na segunda iteração do generate ele não faz tudo novamente, na realidade ele continua de onde parou, ou seja, ele continua na linha 15 que é a linha após o yield. Isso se repete até que o while acabe que vai fazer com que o for que está iterando sobre o generate encerre.