

Formulário de Resposta de Recurso

ANULAÇÃO DE QUESTÃO

RECURSOS QUANTO A GABARITOS PRELIMINARES DA PROVA OBJETIVA



Protocolo: 0000000117

IFSC – INSTITUTO FEDERAL DE SANTA CATARINA – EDITAL 01/2026 CARGO DE
PROFESSOR (DOCENTE EBTT) - GERAL

RECURSOS QUANTO A GABARITOS PRELIMINARES DA PROVA OBJETIVA

RESPOSTA A RECURSO

29 - INFORMÁTICA - CAÇADOR

Nº DA QUESTÃO: 35

A Questão 35 analisa a sintaxe e a lógica de execução do tratamento de exceções em Python e considerou incorreta a Afirmativa III ("O bloco finally é destinado à liberação de recursos e sua execução é garantida, a menos que o fluxo de controle encontre uma instrução return ou break dentro do bloco try correspondente").

A redação da referida assertiva apresenta contradição e ambiguidade direta em relação à especificação formal da linguagem Python:

1 - Documentação Oficial da Linguagem

Python (Section 8.7 – Defining Clean-up Actions)

A documentação oficial do interpretador CPython

estabelece categoricamente: "If a break, continue or return statement executes inside the try clause, the finally clause will execute just prior to the break, continue or return statement's execution."

Ou seja, a presença de instruções return ou break dentro do bloco try não impede nem cancela a execução do bloco finally. Pelo contrário: o interpretador intercepta a instrução de desvio e executa o bloco finally antes de efetivar o retorno ou a quebra do laço.

2 - Ambiguidade e

Indução ao Erro

A redação da assertiva III afirma que a execução é garantida "a menos que..." encontre return ou break. Ao utilizar a preposição com valor de exceção ("a menos que"), o enunciado atribui a return e break uma propriedade de supressão do bloco finally que não existe em Python (fenômeno que só ocorreria com interrupções fatais de processo, como os._exit()).

3 - Inadequação do Item

A frase

gera um vício de formulação: a estrutura sintática confusa induz o candidato a tentar adivinhar se a banca considerou return/break como gatilhos normais de execução prévia ou como exceções à regra de garantia.

Diante do vício de redação que confronta a especificação técnica oficial da linguagem Python (Python Software Foundation), o item carece da clareza exigida para instrumentos avaliativos em concursos públicos.

Ante a dubiedade provocada pela redação da Assertiva III em relação à especificação oficial do Python, solicita-se à Banca Examinadora a ANULAÇÃO da Questão 35 ou a alteração do gabarito oficial para a Alternativa (E).

RESPOSTA DA BANCA: DEFERIDO

JUSTIFICATIVA: Em resposta à fundamentação do(a) candidato(a), após análise desta banca, conclui-se pelo acolhimento do pedido de anulação da questão, pelos fundamentos apresentados a seguir:

O recurso

sustenta vício de redação na assertiva III, porque, segundo a documentação, `return` e `break` no `try` não impedem a execução do `finally`, e a expressão “a menos que” atribuiria a essas instruções uma supressão inexistente, induzindo a erro. Pede a anulação ou a alteração do gabarito para a alternativa E.

A assertiva I afirma que o `else` é executado “exclusivamente quando” o código do `try` é “finalizado com sucesso”, e explica essa expressão por “ou seja, sem o levantamento de nenhuma exceção”. Segundo a documentação oficial, “The optional `else` clause is executed if the control flow leaves the `try` suite, no exception was raised, and no `return`, `continue`, or `break` statement was executed” (Python Language Reference, seção 8.4.3).

É correto que `return`, `break` ou `continue` no `try` impedem o `else`, mesmo sem exceção. A assertiva, porém, não afirma suficiência: “exclusivamente quando” enuncia condição necessária, e demonstrar que ela não é suficiente não a refuta. Esses casos, por si sós, não tornam I falsa.

A redação de I admite, contudo, duas leituras. Na leitura contextual, “finalizado com sucesso” corresponde à conclusão normal do `try`, sem propagação de exceção para fora dele; I é verdadeira, e a resposta é a alternativa C (I e II). Na leitura literal da explicação “nenhuma exceção”, a assertiva exclui até uma exceção levantada e tratada dentro do próprio `try`, o que não corresponde ao comportamento da linguagem. No exemplo abaixo, executado por esta banca, a exceção é levantada e tratada internamente e o `else` externo é executado; nessa leitura, I é falsa, e a resposta é a alternativa A (apenas II).

```
try:
try:
raise ValueError("exceção
interna")
except ValueError:
print("exceção interna tratada")
except ValueError:
print("EXCEPT
externo")
else:
print("ELSE externo executado")
```

Saída verificada:

```
exceção interna tratada
ELSE externo executado
```

As

assertivas II e III têm o mesmo julgamento nas duas leituras. II é correta: o tutorial oficial recomenda que as exceções definidas pelo programador derivem de Exception. III é incorreta: “When a return, break or continue statement is executed in the try suite of a try...finally statement, the finally clause is also executed ‘on the way out’” (seção 8.4.4). As alternativas A e C dependem, assim, apenas da leitura adotada para I, e não há uma única resposta que prescindia dessa escolha. Trocar o gabarito de C para A acolheria uma leitura e rejeitaria a outra sem corrigir o defeito de formulação que originou a controvérsia.

A constatação sobre o finally é correta, mas torna III falsa, como já ocorria no gabarito C; a falsidade de uma assertiva é própria do formato da questão, e não vício de redação. Não se acolhe a alteração para E, pois E inclui justamente III. O pedido de anulação é acolhido pelo defeito de I exposto acima, e não por considerar verdadeira a afirmação sobre a supressão do finally.

Assim, acolher a providência requerida não significa acolher todas as razões apresentadas. A anulação não significa que a banca considere a assertiva I tecnicamente falsa, nem que aceite todas as premissas apresentadas; decorre da coexistência de duas leituras da redação de I que conduzem a alternativas diferentes.

Anula-se a questão 35, com atribuição dos pontos a todas as pessoas candidatas que realizaram a prova, conforme o item 10.7 do edital.

Pelo exposto, recurso
DEFERIDO.

Referências

PYTHON SOFTWARE FOUNDATION. The Python language reference: compound statements. Versão 3.12. [S. l.], [s. d.]. Seções 8.4.1, 8.4.3 e 8.4.4. Disponível em: https://docs.python.org/3.12/reference/compound_stmts.html#the-try-statement. Acesso em: 9 out. 2026.

PYTHON SOFTWARE FOUNDATION. The Python tutorial: errors and exceptions. [S. l.], [s. d.]. Seções 8.3, 8.6 e 8.7. Disponível em: <https://docs.python.org/3/tutorial/errors.html>. Acesso em: 2 out. 2026.

INSTITUTO FEDERAL DE SANTA CATARINA. Edital nº 01/2026: concurso público para professor da carreira de magistério do ensino básico, técnico e tecnológico. [S. l.], 2026. Versão consolidada após retificação de 19 ago. 2026. Anexo III, item 1.1; Anexo X, Informática; itens 10.5 e 10.7. Disponível em: <https://portalfaurgs.com.br/LerArquivo/2f24ef5f-a655-4ddd-a3b2-ec3506edd4a9>. Acesso em: 9 out. 2026.