AC2 - Linguagem de Programação

Total de pontos 10/10

As questões contidas nessa atividade estão relacionadas aos conteúdos das Partes 01, 02, 03, 04, 05, 06, 07, 08 e 09.

O e-mail do participante (**fabiana.campanari@aluno.faculdadeimpacta.com.br**) foi registrado durante o envio deste formulário.

✓ Considerando nosso material didático, assinale a alternativa que contém uma afirmação FALSA sobre o tópico "variáveis locais e globais em Python": *1/1

- ○ A partir do instante em que uma variável global está criada, poderá ser acessada em qualquer instrução do mesmo código-fonte.
- ○ É possível evitar a criação de uma variável local no bloco de uma função usando o comando 'global' (sem apóstrofos), assim a função tratará qualquer referência a variável indicada como uma referência à uma variável global.
- ○ Uma característica da variável local é ser acessível apenas dentro do bloco de instruções da função em que foi criada.
- ◉ Se uma função tem uma variável local com o mesmo identificador de uma variável global, ao referenciar esse identificador em seu bloco de código fará acesso à variável global. ✓
- ○ Os parâmetros das funções são considerados variáveis locais dessas funções, assim como as variáveis criadas nos blocos de código das funções que não sejam explicitamente identificadas como globais.

Feedback

Caso uma função contenha uma variável local com o mesmo nome de uma global, a função optará por acessar a variável local em suas instruções, exceto se indicado explicitamente o contrário (com o uso do comando global, por exemplo). Ou seja, o escopo local tem precedência sobre o global.

Conteúdo abordado no Capítulo 6 da Unidade 2 da apostila.

✓ Considerando que o código-fonte a seguir foi construído com a linguagem de programação Python, indique qual das alternativas possui a sequência correta dos valores que as variáveis globais x e y possuirão em cada um dos momentos indicados no código. *1/1

```
def enigma1(x, y):
    x += 20
    y += 50

def enigma2():
    x = 30
    y = 60

def enigma3():
    global x, y
    x += 40
    y += 70

x = 100
y = 200

enigma1(x, y)
# momento 1

enigma2()
# momento 2

enigma3()
# momento 3
```

```
(momento 1)  x = 100  e  y = 200
(momento 2)  x = 100  e  y = 200
(momento 3)  x = 140  e  y = 270
```

Alternativa 1. ✓

```
(momento 1)  x = 100  e  y = 200
(momento 2)  x = 140  e  y = 270
(momento 3)  x = 140  e  y = 270
```

Alternativa 2.

```
(momento 1)  x = 120  e  y = 250
(momento 2)  x = 120  e  y = 250
(momento 3)  x = 160  e  y = 320
```

Alternativa 3.

```
(momento 1)  x = 100  e  y = 200
(momento 2)  x =  30  e  y =  60
(momento 3)  x =  70  e  y = 130
```

Alternativa 4.

```
(momento 1)  x = 120  e  y = 250
(momento 2)  x =  30  e  y =  60
(momento 3)  x =  70  e  y = 130
```

Alternativa 5.

Feedback

Basta simular a execução do programa e/ou executá-lo.

Conteúdo abordado no Capítulo 6 da Unidade 2 da apostila.

Linguagem de Programaç...

✓ O programa ilustrado na figura a seguir solicita ao usuário um número natural correspondente a um mês e outro número natural correspondente a um ano. O programa deveria exibir uma mensagem com a quantidade exata de dias que contém esse mês nesse ano, porém a função nome_mes não está adequada ao programa, o que impede o funcionamento desejado. Analise as afirmações a seguir e indique a alternativa que contém as afirmações que são verdadeiras a respeito deste programa, incluindo possíveis defeitos e melhorias que poderiam ser feitas nele: (I) Considerando a forma como foi usada, é correto afirmar que a função nome_mes foi implementada incorretamente, pois deveria retornar o nome do mês e não exibi-lo; (II) A função nome_mes possui uma tupla em seu bloco de código, o que não é permitido em funções; (III) Para ajustar a função nome_mes bastaria substituir a exibição por return. *1/1

```
def bissexto(ano):
    return ano%4==0 and ano%100!=0 or ano%400==0

def qtd_dias(mes, ano):
    if mes==2:
        if bissexto(ano):
            return 29
        else:
            return 28
    if mes==4 or mes==6 or mes==9 or mes==11:
        return 30
    return 31

def nome_mes(mes):
    meses = ('janeiro', 'fevereiro', 'março', 'abril',
             'maio', 'junho', 'julho', 'agosto',
             'setembro', 'outubro', 'novembro', 'dezembro')
    print(meses[mes-1])

mes = int(input('mês? '))
ano = int(input('ano? '))
print(f'{nome_mes(mes)} de {ano} terá {qtd_dias(mes, ano)} dias')
```

- ○ São verdadeiras apenas as afirmações I e II.
- ○ Todas as afirmações são verdadeiras.
- ○ Nenhuma das afirmações é verdadeira.
- ◉ São verdadeiras apenas as afirmações I e III. ✓
- ○ São verdadeiras apenas as afirmações II e III.

Feedback

- Afirmação I: correta, pois a função foi utilizada como uma "função pergunta", por isso esperava-se o retorno de uma resposta que, neste caso, seria uma string com o nome do mês correspondente ao número natural informado como argumento;

- Afirmação II: incorreta, pois é permitido que o bloco de instruções de uma função contenha tuplas;

- Afirmação III: correta, pois para a função ser corrigida bastaria trocar a exibição com print por uma devolução de resposta com o comando return.

Conteúdo abordado no Capítulo 6 da Unidade 2 e Capítulo 9 da Unidade 3 da apostila.

✓ Veja o programa ilustrado na figura a seguir e assinale a alternativa que contém uma afirmação INCORRETA sobre ele. *1/1

```
def quociente(x, y):
    return x / y

x = int(input('dividendo? '))

while True:
    y = int(input('divisor? '))
    if y != 0: break

print(f'{x} / {y} = {quociente(x, y)}')
```

- ○ É possível que a função 'quociente' jamais seja executada neste programa.
- ○ Neste programa o laço de repetição só será encerrado caso a condição do 'if' que está no bloco de instruções do laço resulte em verdadeiro.
- ○ Caso não ocorra nenhum problema com a primeira instrução do programa, é possível garantir que o bloco de instruções do laço sempre será executado pelo menos uma vez.
- ◉ Neste programa o laço de repetição só será encerrado caso a condição do 'if' que está no bloco de instruções do laço resulte em falso. ✓
- ○ Neste programa a condição do laço sempre resultará em verdadeiro.

Feedback

É simples verificar que neste programa o laço tem como condição a constante booleana 'True', portanto sempre que a condição for avaliada resultará em verdadeiro e, consequentemente, o bloco de instruções do laço será executado.

Caso não ocorra nenhum problema, a única forma de encerrar o laço será por meio da execução do comando 'break' que está no bloco de instruções do 'if'. Evidentemente, como o comando 'break' está no bloco do 'if', só será executado caso a condição do 'if' resulte em verdadeiro.

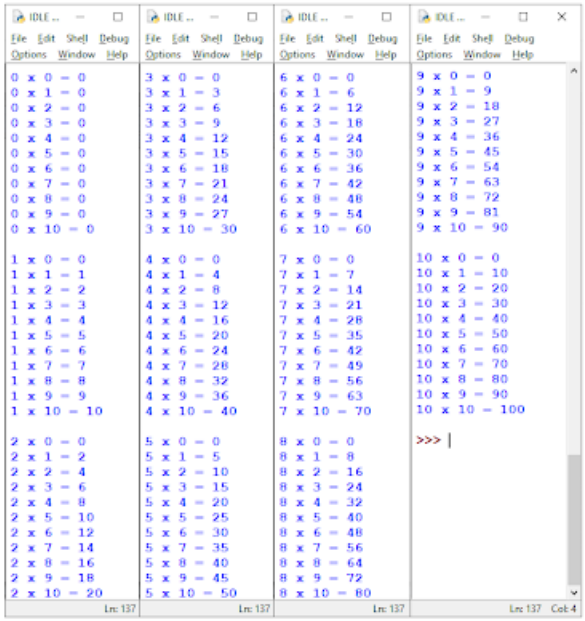
Essa é uma estrutura de repetição semelhante ao "repita... até que", bastante utilizada para validação de dados de entrada. Essa estrutura também garante que o bloco de instruções do laço será executado pelo menos uma vez.

Conteúdo abordado nos capítulos 7 e 8 da Unidade 3 da apostila.

✓ Considerando que a codificação ilustrada a seguir deveria exibir a tabuada de multiplicação de 0 até 10, selecione a alternativa que indica corretamente qual o problema que impede que o programa funcione adequadamente. Atenção: há uma imagem do enunciado com maior resolução na postagem desta atividade. *1/1

Deveria gerar a saída ilustrada à esquerda, mas há algum problema.

```
x = 0
while x <= 10:
    y = 0
    while y <= 10:
        print(f'{x} x {y} = {x * y}')
    y += 1
    print()
    x += 1
```



- ○ Há um erro de indentação na instrução que incrementa a variável 'x'.
- ◉ Há um erro de indentação na instrução que incrementa a variável 'y'. ✓
- ○ A variável 'y' não deveria receber zero no ponto em que recebeu neste código, pois isso fez com que o programa permanecesse em loop infinito mesmo com a variável 'y' sendo incrementada a cada execução do bloco de instruções do laço 'while' aninhado.
- ○ Há um erro de indentação na instrução 'print' que não possui argumentos.
- ○ O Python não permite a criação de códigos-fonte com uma instrução 'while' dentro do bloco de instruções de outro 'while'.

Feedback

Basta simplesmente analisar a codificação e verificar que a variável 'y' tem o papel de controlar quantas vezes o laço mais interno 'while' será executado. Como a variável 'y' inicia com o valor zero e a condição do laço 'while' é (y <= 10), para que o laço encerre a variável deveria ser incrementada de um em um, de modo que o programa repetisse a exibição das multiplicações corretamente. Como o incremento da variável está fora do bloco de instruções do 'while' mais interno, gera-se um loop infinito, pois 'y' não é atualizado.

Conteúdo abordado no Capítulo 8 da Unidade 3 da apostila.

✓ Analise o programa ilustrado na figura a seguir e assinale a alternativa que contém uma afirmação correta sobre ele. Observação: assuma que o valor dado inicialmente à variável 'vale_presente' sempre será um número real maior ou igual a zero. *1/1

```
vale_presente = float(input('Valor? '))

while vale_presente > 0:
    mercadoria = float(input('Mercadoria? '))
    if mercadoria > vale_presente:
        print('Muito caro! Escolha outra mercadoria')
        continue
    vale_presente -= mercadoria

print('Obrigado pela compra!')
```

- ○ Neste programa o laço 'while' será encerrado caso uma das duas situações ocorra: (I) se a variável 'vale_presente' estiver com um valor menor ou igual à zero quando a condição do laço for avaliada ou; (II) se o valor informado para a variável 'mercadoria' for superior ao valor da variável 'vale_presente'.
- ○ Quando o usuário inserir um valor para a variável 'mercadoria' que supere o valor que está na variável 'vale_presente', o laço será encerrado e a mensagem 'Obrigado pela compra!' será exibida.
- ○ Caso o usuário insira um valor para a variável 'mercadoria' superior ao que está na variável 'vale_presente', a mensagem 'Muito caro! Escolha outra mercadoria' será exibida, mas a variável 'vale_presente' também será decrementada, pois não foi utilizado o comando 'else' para complementar o comando 'if'.
- ◉ Neste programa a mensagem 'Obrigado pela compra!' só será exibida após a condição do laço ser avaliada e resultar em False. ✓
- ○ Neste programa é possível que a variável 'vale_presente' guarde um valor negativo, para isso basta que a variável 'mercadoria' receba um valor superior ao de 'vale_presente'.

Feedback

Basta simular a execução do programa e/ou executá-lo.

Conteúdo abordado no Capítulo 8 da Unidade 3 da apostila.

✓ O programa da figura a seguir recebe como entrada um número natural n e deve exibir a sequência de valores naturais de zero até (n-1) e de (n-1) até zero. Por exemplo, se n=4 o programa deverá exibir 0 1 2 3 e, na próxima linha, 3 2 1 0. Assinale a alternativa que contém os valores que substituem adequadamente os símbolos ? (interrogação), e na ordem correta, de modo que o programa funcione conforme a especificação. Observação: Desconsidere os ';' das alternativas, são apenas separadores da sequência de valores. *1/1

```
n = int(input('n? '))

comeco = ?
final  = ?
passo  = ?

for i in range(comeco, final):
    print(i, end=' ')

print()

for i in range(?, ?, passo):
    print(i, end=' ')
```

- ○ 0 ; n + 1 ; -1 ; final - 1 ; comeco + 1
- ◉ 0 ; n ; -1 ; final - 1 ; comeco - 1 ✓
- ○ 0 ; n ; 1 ; final + 1 ; comeco - 1
- ○ 0 ; n ; -1 ; final ; comeco + 1
- ○ 0 ; n ; -1 ; final + 1 ; comeco + 1

Feedback

Basta simplesmente analisar a codificação e/ou executá-la preenchendo as lacunas com os valores corretos para gerar a saída esperada. Em seguida, verificar qual das alternativas corresponde ao preenchimento feito.

Conteúdo abordado no Capítulo 9 da Unidade 3 da apostila.

✓ Considerando que a codificação ilustrada a seguir deveria simular um relógio digital (hh:mm:ss) que mostraria um horário arbitrário durante exatamente uma semana, e atualizado a cada segundo, selecione a alternativa que indica corretamente qual o problema que impede que o programa funcione adequadamente. *1/1

```
from time import sleep

dias = 7

while dias > 0:
    for h in range(24):
        for m in range(60):
            for s in range(60):
                print(f'{h:02}:{m:02}:{s:02}')
                sleep(1)
    dias += 1
```

- ○ O problema é que a variável 'dias' está com a indentação incorreta, ajustando esta indentação o programa funcionaria adequadamente.
- ○ O problema é que a função 'sleep' não pode ser usada, pois não existe em Python.
- ○ O problema é que as variáveis 'h', 'm' e 's' não foram criadas antes de serem usadas nos laços 'for'.
- ○ O problema é que não é possível criar uma codificação em Python em que um laço 'for' esteja aninhado em um laço 'while', portanto deve-se adaptar o programa para ter apenas 'while' aninhado em 'while' e 'for' aninhado em 'for'.
- ◉ O problema é que a variável 'dias' está sendo incrementada, sendo que deveria ser decrementada. ✓

Feedback

Basta simplesmente analisar a codificação e verificar que a variável 'dias' tem o papel de controlar quantas vezes o laço mais externo 'while' será executado. Como a variável 'dias' inicia com o valor sete e a condição do laço 'while' é (dias > 0), para que o laço encerre a variável deveria ser decrementada de um em um, de modo que o programa repetisse a exibição dos horários durante exatamente 7 dias.

Conteúdo abordado nos capítulos 8 e 9 da Unidade 3 da apostila.

✓ Veja o programa ilustrado na figura a seguir, analise as afirmações sobre ele e assinale a alternativa correta: (I) A execução da terceira instrução não irá gerar um erro no programa, pois a sequência referenciada pela variável 'idades' é uma lista e, consequentemente, é uma sequência mutável, por isso poderá ter seus itens modificados; (II) O programa irá gerar um erro na terceira instrução, pois não há nessa lista o índice indicado entre colchetes; (III) A quarta instrução não será executada, pois ocorrerá algum erro antes do fluxo de execução atingi-la. *1/1

```
idades = [45,67,12,30,51,24] # 1ª instrução

tamanho = len(idades)        # 2ª instrução

idades[tamanho] += 1         # 3ª instrução

print(idades)                # 4ª instrução
```

- ○ Todas as afirmações estão corretas.
- ○ Nenhuma das afirmações está correta.
- ○ Apenas as afirmações I e II estão corretas.
- ○ Apenas as afirmações I e III estão corretas.
- ◉ Apenas as afirmações II e III estão corretas. ✓

Feedback

Basta simular a execução do programa e/ou executá-lo.

Conteúdo abordado no Capítulo 9 da Unidade 3 da apostila.

✓ Veja o programa ilustrado na figura a seguir, analise as afirmações sobre ele e assinale a alternativa correta: (I) O programa irá gerar um erro, pois a sequência referenciada pela variável 'frase' é uma string e, consequentemente, é uma sequência mutável, por isso não poderá ser modificada pela terceira instrução; (II) O programa irá gerar um erro, pois em Python as strings devem ser delimitadas somente por aspas/aspas duplas, como em "exemplo de string", e não por apóstrofos/aspas simples, como em 'exemplo de string'; (III) O programa irá gerar um erro, pois a indexação da string na terceira instrução está errada, afinal não existe nesta string a posição (tamanho-1). *1/1

```
frase = 'A linda casa amarela!' # 1ª instrução

tamanho = len(frase)            # 2ª instrução

frase[tamanho-1] = '?'          # 3ª instrução

print(frase)                    # 4ª instrução
```

- ○ Apenas as afirmações II e III estão corretas.
- ◉ Nenhuma das afirmações está correta. ✓
- ○ Apenas as afirmações I e III estão corretas.
- ○ Apenas as afirmações I e II estão corretas.
- ○ Todas as afirmações estão corretas.

Feedback

Basta simular a execução do programa e/ou executá-lo.

Conteúdo abordado no Capítulo 9 da Unidade 3 da apostila.

Este formulário foi criado em FACULDADE IMPACTA DE TECNOLOGIA - FIT.

Google Formulários