Introdução à programação em Python

Renato Hidaka Torres

Assunto: Funções

Questão 01: Na linguagem Python, sobre os parâmetros de uma função, marque V ou F:

(F) Na definição dos parâmetros de uma função, é possível definir parâmetros com argumento default. Para isso, basta realizar uma atribuição simples ou múltipla.

(V) Na definição dos parâmetros de uma função, se nenhum parâmetro tiver valor default definido, então a passagem de argumentos para esses parâmetros é obrigatória.

(V) Os parâmetros de uma função são utilizados para armazenar os argumentos que são passados para a função. Os parâmetros formais, também conhecido como simples, podem receber argumentos na forma posicional ou nomeada por palavra-chave.

(V) O parâmetro de uma função que possui o * precedendo o seu nome, é um parâmetro especial que pode receber um número indeterminado de argumentos. Ao passar argumentos para esse parâmetro, só é possível utilizar a forma posicional.

(F) Na definição dos parâmetros de uma função, só se pode definir parâmetros com argumentos default após a definição dos parâmetros formais ou do parâmetro especial que possui o * precedendo o seu nome.

(F) Os parâmetros de uma função são utilizados para armazenar os argumentos que são passados para a função. Ao utilizar uma função, o número de argumentos passados para a função deve ser igual ao número de parâmetros da função.

(V) Na definição dos parâmetros de uma função, se um parâmetro tiver valor default definido, então a passagem de argumento para esse parâmetro é opcional, podendo ser feita de forma posicional ou nomeada por palavra-chave.

(V) Na definição dos parâmetros de uma função, é sempre necessário que os parâmetros tenham nomes distintos, considerando a diferença entre caixa alta e caixa baixa. Ou seja, valor é diferente de VALOR, por exemplo.

Questão 02: Na linguagem Python, sobre os argumentos de uma função, marque V ou F:

(V) Ao chamar uma função, se necessário, devemos passar os seus argumentos. Na passagem de argumentos, podemos passar qualquer informação, independentemente do tipo.

(V) Ao chamar uma função, podemos passar argumentos de forma posicional ou nomeada por palavra-chave. Se as duas estratégias forem utilizadas, então todas as passagens por palavra-chave devem vir depois das passagens posicionais.

(F) Na passagem de argumentos nomeado por palavra-chave, a ordem dos argumentos não é importante. Logo, na mesma chamada da função, podemos passar mais de um argumento para o mesmo parâmetro formal, também conhecido como simples.

(V) Na passagem de argumentos posicionais, o número de argumentos deve coincidir com o número de parâmetros da função.

(V) Na passagem de argumentos nomeado por palavra-chave, se um argumento for passado para um parâmetro que possui valor default, o argumento passado sobrescreve o valor default desse parâmetro.

(F) Na passagem de argumentos, uma função que possui parâmetros pode não receber argumentos. Nesse caso, é necessário que os parâmetros sejam especiais com o * precedendo os seus nomes.

(F) Na passagem de argumentos, os argumentos passados da forma posicional para o parâmetro especial com o * precedendo o seu nome são armazenados em uma tupla. Na linguagem Python, uma tupla é uma coleção imutável onde os elementos devem ficar entre chaves.

Questão 03: Na linguagem Python, sobre o retorno de uma função, marque V ou F:

(V) Ao definir uma função, podemos declarar mais de um retorno. Porém, toda vez que a função for chamada para execução, somente um retorno é executado, uma vez que o retorno sempre interrompe o fluxo de execução da função.

(V) No retorno de uma função, é possível retornar valores armazenados em diferentes variáveis. Para isso, basta separar cada variável por vírgula. Quando este for o caso, na chamada da função, deve-se utilizar a atribuição múltipla para receber os valores retornados.

(V) Se um função não possuir retorno definido, então ela é uma função que retorna None e, por isso, é denominada como void.

(F) Ao definir uma função, podemos declarar um retorno sem especificar o valor a ser retornado. Nesse caso, a palavra reservada return é utilizada exclusivamente para interromper o fluxo de execução da função.

(V) Ao definir uma função, se você precisar retornar valores armazenados em diferentes variáveis, você pode construir e retornar uma lista com os valores armazenados nas variáveis.

(V) Ao definir uma função, o retorno representa a saída da função. Ou seja, quando a função for chamada, a variável que recebe a atribuição da função sempre armazena o valor retornado.

Questão 04: Construa a docstring das seguintes funções e renomeie cada função para um nome mais apropriado.

a)

```
def f(*valores):

  M = valores[0]
  for item in valores:
    if item > M:
      M = item
  return M
```

b)

```
def f(x, y=10, z=5):

  if y > x > z or z > x > y:
    return x
  elif x > y > z or z > y > x:
    return y
  else:
    return z
```

c)

```
def f(arquivo):

  with open(arquivo, 'r', encoding='utf-8') as f:
    return len(f.readlines())
```

d)

```
def f(arquivo, arg='teste'):

  with open(arquivo, 'r', encoding='utf-8') as f:
    n = 0
    for item in f:
      if item[:len(arg)] == arg:
        n += 1

    return n
```

e)

```
def f(n=2):

  if n < 2:
    return False

  for q in range(2, n):
    if n % q == 0:
      return False

  return True
```