

Questão 9

a. Como podemos definir uma tupla simples ?

R - Uma tupla é um tipo de variável composta no python, onde podemos atribuir valores do tipo numérico, lógico e string. Para declarar uma tupla simples, podemos fazer de duas formas.

- A variável que será a nossa tupla poderá ser atribuída com o objeto `tuple()`.
 - `nome_variável_composta = tuple();`
- A variável que será a nossa tupla poderá ser atribuída simplesmente com um abrir e fechar de parênteses.
 - `nome_variável_composta = ()`

Figura 1: forma de declarar uma tupla simples

```
1  tupla1 = tuple()
2  tupla2 = ()
3
4  print(type(tupla1))
5  print(type(tupla2))
6
```

Figura 2: saída mostrando que tupla1 e tupla2 são objetos da classe 'tuple'.

```
PROBLEMAS  SAÍDA  CONSOLE DE DEPURACÃO  TERMINAL

/bin/python3 /home/luca/Documentos/UFPE/Python_UFPE/Lista1/teste.py
luca@lucas-santos ~/Documentos/UFPE/Python_UFPE/Lista1
$ /bin/python3 /home/luca/Documentos/UFPE/Python_UFPE/Lista1/teste.py
<class 'tuple'>
<class 'tuple'>
luca@lucas-santos ~/Documentos/UFPE/Python_UFPE/Lista1
$ |
```

b. Como devemos definir uma cópia de uma variável ?

R- Uma variável no python pode ser simples ou compostas a maneira como se copia uma variável simples é bem diferente da maneira que se copia para uma variável composta, vejamos nos exemplos abaixo abaixo:

- variável simples

Figura 3: atribuição de valores inteiros a variáveis simples e printagem dessas variáveis

```

1  a = 100
2  b = a
3  a = 10
4
5  print(a)
6  print(b)
7

```

- variável composta

Figura 4: Maneira corrente para copiar uma variável composta

```

1  x = [1, 2, 3]
2  y = x[:]
3  x.append(10)
4  print(x)
5  print(y)
6

```

Figura 5: Saída das listas x e y da cópia de variáveis compostas

```

luca@lucas-santos ~/Documentos/UFPE/Python_UFPE/Lista1
$ /bin/python3 /home/luca/Documentos/UFPE/Python_UFPE/Lista1/teste.py
[1, 2, 3, 10]
[1, 2, 3]
luca@lucas-santos ~/Documentos/UFPE/Python_UFPE/Lista1
$ |

```

c. O que são variáveis em python ? Como podemos definir uma variável ?

R - No python variável é o elemento mais básico da linguagem, e é usado para armazenar valores na memória. Para definir uma variável, basta usar o operador de atribuição, que na linguagem Python é o símbolo de igualdade.

Figura 6: Declaração de variáveis no Python

```

1
2  nome1 = "Lucas"
3  nome2 = 20
4  nome3 = 4.5
5  nome4 = True
6

```

Os nomes das variáveis podem ser qualquer um, mas é recomendado colocar os nomes das variáveis de acordo com a função dela. As variáveis também tem tipo, e o tipo pode ser:

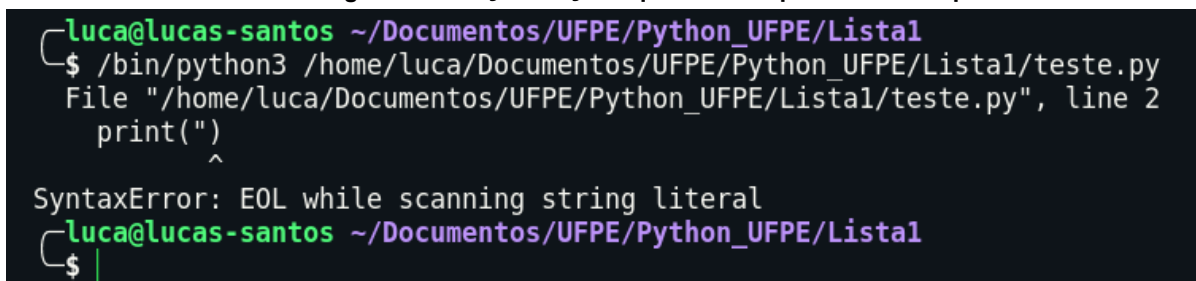
- String(cadeia de caracteres)
- numérico
 - inteiro
 - ponto flutuante
- booleano

d. Qual é a principal funcionalidade do `print()` ? O que representa ? O que acontece quando deixamos de inserir um parênteses, ou até mesmo ambos os parênteses do `print`?

R - A função `print` é uma função built-in python, ela é usada para imprimir objetos. Como ela é uma função built-in do python, a função `print` já está disponível para ser usada sem precisar ser importada.

Quando é esquecido de incluir alguma aspa, o interpretador do python lança uma exceção do tipo `SyntaxError`, mostrado na figura abaixo:

Figura 7: Exceção lançado quando é esquecido uma aspa



```

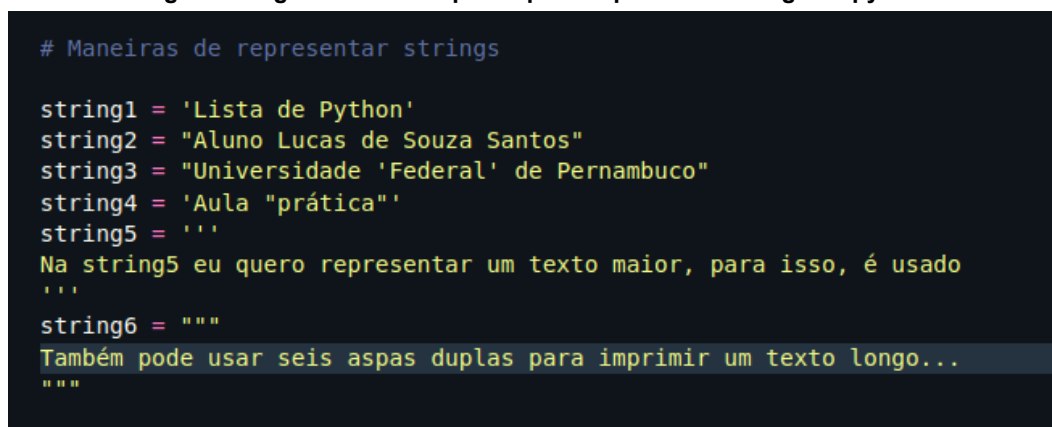
luca@lucas-santos ~/Documentos/UFPE/Python_UFPE/Lista1
$ /bin/python3 /home/luca/Documentos/UFPE/Python_UFPE/Lista1/teste.py
File "/home/luca/Documentos/UFPE/Python_UFPE/Lista1/teste.py", line 2
    print("
          ^
SyntaxError: EOL while scanning string literal
luca@lucas-santos ~/Documentos/UFPE/Python_UFPE/Lista1
$ |
  
```

Quando é esquecido as duas aspas o interpretador do python apenas pula uma linha e não lança nenhuma exceção.

e. O que é uma string ? Como podemos representá-las ? A string é mutável ? O que acontece quando deixamos de inserir as aspas ou uma das aspas ?

R - De acordo com a documentação oficial do python, string é uma sequência imutável de pontos de código unicode. Podemos representar uma string através de um texto ou uma palavra dentro de aspas simples ou aspas duplas.

Figura 8: Algumas formas que se pode representar strings no python



```

# Maneiras de representar strings

string1 = 'Lista de Python'
string2 = "Aluno Lucas de Souza Santos"
string3 = "Universidade 'Federal' de Pernambuco"
string4 = 'Aula "prática"'
string5 = '''
Na string5 eu quero representar um texto maior, para isso, é usado
'''
string6 = """
Também pode usar seis aspas duplas para imprimir um texto longo...
"""
  
```

Como já dito nesse texto, as strings são objetos imutáveis. Quando é esquecido de usar uma aspas ou as duas, o interpretador python lança uma exceção, como são visto abaixo:

Figura 9: Código python com erro de sintaxe.

```
string1 = 'Lista de Python
```

Figura 10: Exceção lançada quando é esquecido de usar uma aspas do código da figura 9

```
luca@lucas-santos ~/Documentos/UFPE/Python_UFPE/Listal
$ /bin/python3 /home/luca/Documentos/UFPE/Python_UFPE/Listal/teste.py
File "/home/luca/Documentos/UFPE/Python_UFPE/Listal/teste.py", line 4
    string1 = 'Lista de Python
                ^
SyntaxError: EOL while scanning string literal
luca@lucas-santos ~/Documentos/UFPE/Python_UFPE/Listal
$ |
```

Figura 11: código python com erro de sintaxe

```
string1 = Lista de Python'
```

Figura 12: Exceção lançada quando é esquecido de usar umas das aspas só que agora no lado oposto comparado com o erro da figura 9

```
luca@lucas-santos ~/Documentos/UFPE/Python_UFPE/Listal
$ /bin/python3 /home/luca/Documentos/UFPE/Python_UFPE/Listal/teste.py
File "/home/luca/Documentos/UFPE/Python_UFPE/Listal/teste.py", line 4
    string1 = Lista de Python'
                ^
SyntaxError: invalid syntax
luca@lucas-santos ~/Documentos/UFPE/Python_UFPE/Listal
$ |
```

Figura 13: código com erro de sintaxe

```
string1 = Lista de Python
```

Figura 13: Exceção lançada quando é esquecido de usar as duas aspas na declaração de uma string

```

luca@lucas-santos ~/Documentos/UFPE/Python_UFPE/Lista1
$ /bin/python3 /home/luca/Documentos/UFPE/Python_UFPE/Lista1/teste.py
File "/home/luca/Documentos/UFPE/Python_UFPE/Lista1/teste.py", line 4
    string1 = Lista de Python
                ^
SyntaxError: invalid syntax
luca@lucas-santos ~/Documentos/UFPE/Python_UFPE/Lista1
$ |

```

- f. Quais as operações e principais métodos utilizados com cada tipo de variável (numérica, tipo lógico e string) ? Para facilitar uma tabela, não esqueça de descrever cada operação e método.

○ Operações para numéricos

■ Operadores numéricos

Operadores	nome	Example
+	adição	$x + y$
-	subtração	$x - y$
*	multiplicação	$x * y$
/	divisão	x / y
%	resto da divisão	$x \% y$
**	potenciação	$x ** y$
//	divisão inteira	$x // y$

■ Operadores de atribuição

Operadores	Exemplo	Igual a
=	$x = 5$	$x = 5$
+=	$x += 3$	$x = x + 3$
-=	$x -= 3$	$x = x - 3$
*=	$x /= 3$	$x = x * 3$
/=	$x /= 3$	$x = x / 3$
%=	$x \% = 3$	$x = x \% 3$
**=	$x ** = 3$	$x = x ** 3$
//=	$x /= 3$	$x = x // 3$

■ Operadores de Comparação

Operadores	nome	Example
<code>==</code>	igual	<code>x == y</code>
<code>!=</code>	diferente	<code>x != y</code>
<code>></code>	maior que	<code>x > y</code>
<code><</code>	menor que	<code>x < y</code>
<code>>=</code>	maior e igual a	<code>x >= y</code>
<code><=</code>	menor e igual a	<code>x <= y</code>

■ Operadores de identidade

Operadores	descrição	Example
<code>is</code>	Retorna verdadeiro se ambos objetos forem iguais.	<code>x is y</code>
<code>is not</code>	Retorna verdadeiro se ambos os objetos forem diferentes.	<code>x is not y</code>

❖ Métodos usados para variáveis numéricas

Métodos	retorno	Example
<code>max()</code>	retorna o maior número	<code>max(6, 3) = 6</code>
<code>min()</code>	retorna o menor número	<code>min(4, 5) = 4</code>
<code>abs()</code>	retorna um número sem sinal	<code>abs(-5) = 5</code>
<code>pow()</code>	retorna um valor elevado a outro.	<code>pow(3, 2) = 4 ** 2 = 9</code>
<code>round()</code>	retorna o número arredondado	<code>round(3.123,2) = 3.12</code>
<code>math.sqrt()</code>	retorna a raiz quadrada de um número.	<code>math.sqrt(9) = 3</code>
<code>math.pi</code>	Retorna o valor da constante PI	<code>math.pi = 3.141592...</code>

- Variável lógica
 - Operadores lógicos

Operadores	descrição	Example
and	retorna verdadeiro se as duas condições forem verdadeiras.	$x = 3$ $x < 5$ and $x > 2$
or	retorna verdadeiro se uma das afirmações for verdadeira.	$x = 1$ $x > 5$ or $x < 2$
not	retorna verdadeira se a afirmação for falsa.	$x = 4$ not($x > 5$)

- Variável string
 - Operadores para strings

Operadores	descrição
+	retorna a união de duas ou mais string em uma.
*	Retorna a repetição de uma string em várias.

- métodos usado com strings

método	descrição
upper()	string em maiúscula
lower()	string em minúscula
strip()	elimina espaço no início e final da string
split()	retorna uma lista
count(item)	número de ocorrência do item
capitalize()	primeiro caractere maiúsculo
title()	primeiro caractere maiúsculo de todas as palavras
find(item)	retorna o índice da primeira ocorrência
center(largura)	centraliza string dentro da largura passada

- g. Você pode usar um sinal de menos para fazer um número negativo como -2. O que acontece se puser um sinal de mais antes de um número? E se inscrever assim: 2++2?

R - Quando é colocado o sinal de mais antes de um número o interpretador python retorna um número positivo ou sem sinal. Quando ocorre uma operação do tipo:

2++2,

o interpretador python vai entender com se fosse algo do tipo

2 + (+2)

(+2) == 2

2 + 2 == 4

logo a saída é igual a quatro.

Operação	Resultado
2 -+ 3	-1
2 ++ 2	4
2 -- 3	5
2 +- 3	-1

- h. Na notação matemática, zeros à esquerda são aceitáveis, como em 02. O que acontece se você tentar usar isso no Python?

R - Vai dar um erro de sintaxe, como é visto na imagem abaixo:

Figura 14: Erro de sintaxe do tipo zero à esquerda.

```

luca@lucas-santos ~/Documentos/UFPE/Python_UFPE/Lista1
$ /bin/python3 /home/luca/Documentos/UFPE/Python_UFPE/Lista1/teste.py
File "/home/luca/Documentos/UFPE/Python_UFPE/Lista1/teste.py", line 2
  print(02)
        ^
SyntaxError: leading zeros in decimal integer literals are not permitted; use an 0o prefix
for octal integers
luca@lucas-santos ~/Documentos/UFPE/Python_UFPE/Lista1
$

```

- i. Se você esquecer de inserir os operadores entre os valores, o que irá acontecer? Exemplo: x=2 3

R - Vai dar um erro de sintaxe, como na figura abaixo:

Figura 15: Erro de sintaxe

```

luca@lucas-santos ~/Documentos/UFPE/Python_UFPE/Lista1
$ /bin/python3 /home/luca/Documentos/UFPE/Python_UFPE/Lista1/teste.py
File "/home/luca/Documentos/UFPE/Python_UFPE/Lista1/teste.py", line 2
  x = 2 3
        ^
SyntaxError: invalid syntax
luca@lucas-santos ~/Documentos/UFPE/Python_UFPE/Lista1
$

```


- j. **Na operação de atribuição vimos que é aceitável inserir a expressão $n = 22$ (n recebe 22). Explique o que acontece se você inserir $42 = n$?**

R - Erro de sintaxe, já que no python as variáveis não podem ter um numeral como nome.

- k. **Python é uma linguagem bastante simples, me explique o que acontece se você inserir o ; (ponto e vírgula) no final de uma instrução em Python. E se colocássemos um ponto final, o que aconteceria?**

R - No python o uso de ponto e vírgula é opcional, quando usado o interpretador não retorna nenhum erro. Ele é usado, normalmente no python, para separar várias instruções que estão em uma mesma linha; contudo, o melhor caminho para separar vários comandos, é escrever cada um em uma linha diferente.

Já o uso do ponto ou operador dot é usado para poder acessar os métodos de um certo objeto. Se você pretende usar algum método de uma certa classe, no objeto que foi instanciado no seu programa, basta você colocar o nome da variável, ou do objeto criado, e no final do nome colocar um ponto. Assim todos os métodos da classe que o objeto pertence aparecerá.