Computação Distribuída
2019 / 2020

Licenciatura em Engenharia Informática

Lab. 01 – Tipos de dados, módulos e funções em Python

1. Números

Em Python, um dos tipos de dados mais simples de utilizar são os números. Usando o *REPL* do Python, é possível realizar operações matemáticas, como por exemplo:

```
>>> 2 + (2 * 4)
10
```

Recorrendo ao REPL do Python, resolva os seguintes exercícios:

1. Execute as seguintes operações matemáticas e diga o que faz cada uma delas: 3/2, 3//2, 3%2 e 3**2.

2. Calcule a média dos seguintes conjuntos de números (2, 4), (4, 8, 9), (12, 14/6, 15).

3. O volume de uma esfera é dada por 4/3*pi*r^3. Calcule o volume de uma esfera de raio 5.

4. Use a operação *modulo* para verificar quais dos seguintes números são pares ou ímpares: 1, 5, 29, 60/7. Sugestão: o resto da divisão de um número par por 2 é zero.

2. Strings

Uma *string* é uma sequência de caracteres geralmente utilizada para representar palavras, frases ou textos de um programa. Para inicializar uma *string* em Python, basta utilizar aspas ou plicas à volta do texto:

```
>>> a = “Hello World”
>>> print(a)
Hello World
```

Recorrendo à documentação do Python sobre *strings* (https://docs.python.org/3/library/string.html), resolva os seguintes problemas:

1. Inicialize a string “abc” numa variável de nome *s* e:

a) Utilize uma função que calcule o tamanho da string.
b) Escreva a sequência de operações necessária para transformar a string “abc” na string “aaabbbccc”. Sugestão: Utilize concatenação de strings e indexação de caracteres.

2. Inicialize a string “aaabbbccc” numa variável de nome *s:*

   a) Utilize uma função que lhe permita encontrar a primeira ocorrência de “b” e a primeira ocorrência de “ccc”.
   b) Utilize uma função que lhe permita alterar todas as ocorrências de “a” para “X”, e depois utilize a mesma função para alterar apenas a primeira ocorrência de “a” para “X”.

3. Partindo da string s com “aaa bbb ccc”, que sequência de operações pode escrever para chegar às seguintes strings. Sugestão: Pode utilizar a função *replace*.

   a) “AAA BBB CCC”
   b) “AAA bbb CCC”

3. Listas

Uma lista em Python é uma estrutura de dados que permite agrupar valores. As listas podem conter valores de vários tipos, mas normalmente os valores pertencem ao mesmo tipo.

```
>>> l = [1, 2, 3, 4, 5]
>>> l[0]      # Get 0th element of list (zero-based index)
1
>>> l[1:3]    # Get sublist using slicing
[2, 3]
```

Inicialize uma lista na variável “l” com os seguintes valores [1, 4, 9, 10, 23]. Recorrendo à documentação do Python sobre listas (https://docs.python.org/3.5/tutorial/introduction.html#lists), resolva os seguintes problemas:

1. Escreva a forma de obter as sublistas [4, 9] e [10, 23] usando *slicing*.

2. Acrescente o valor 90 ao fim da lista “l”. Utilize concatenação de listas e o método *append*.

3. Calcule a média dos valores da lista. Sugestão: utilize as função *sum* e *len*.

4. Remova a sublista [4, 9].

4. Módulos

Em Python, um módulo é um ficheiro que define e implementa um conjunto de funções que podem ser utilizadas noutros programas. Para importar um módulo em Python deve utilizar a palavra *import* seguido do nome do módulo e depois poderá aceder às funções.

```
import math
print(math.sin(0.0))
```

Recorrendo à documentação de matemática da linguagem Python (https://docs.python.org/3/library/math.html):

1. Calcule o máximo divisor comum dos seguintes pares: (15, 21), (152, 200), (1988, 9765).

2. Calcule o logaritmo base 2 dos seguintes números: 0, 1, 2, 6, 9, 15.

3. Faça um programa que peça um número ao utilizador e apresente os resultados das seguintes funções trigonométricas: *seno*, *coseno*, *tangente*.

5. Funções

Uma função consiste numa sequência de instruções que são executadas quando a própria função é invocada. Por exemplo, o seguinte programa permite-nos fazer *print* de duas palavras cada vez que se invoca a função *do_hello()*.

```
def do_hello():
    print("Hello")
    print("World")

do_hello()
```

As funções podem receber parâmetros e podem retornar valores (usando a palavra *return*):

```
def add_one(x):
    print("Got x=", x)
    return x + 1

value = add_one(1)
```

1. Implemente uma função *add2* que some dois números passados como parâmetros da função. Depois crie a função *add3* que some três números.

2. Faça uma função que retorne o maior de dois números.

3. Faça uma função *is_divisable* que receba uma variável *a* e uma variável *b*, e verifique se *a* é divisível por *b*, retornando verdadeiro se for divisível ou falso caso contrário.

4. Faça uma função *average* que calcule a média de uma lista de números passada como parâmetro.

6. Funções recursivas

Em programação, uma função recursiva é uma função que recorre a si própria de modo a resolver o problema. Tome como exemplo a seguinte função *factorial* que calcula o factorial de um número:

$$f(x)=\begin{cases} 1, se\ x=0 \\ x*f(x-1), cc \end{cases}$$

```
def factorial(x):
    if x == 0:
        return 1
    else:
        return x * factorial(x-1)
```

1. Implemente a função factorial e teste com vários valores.

2. Pretende-se implementar uma função recursiva para calcular a soma dos *n* primeiros números inteiros. Escreva a definição matemática da solução e depois implemente e teste.

3. A sequência de Fibonnaci é uma sequência de números inteiros em que cada termo da sequência corresponde à soma dos dois termos anteriores. Tendo em conta a definição matemática descrita, implemente a função *fib(n)*.

   $$fib(x)=\begin{cases} 0, se\ x=0 \\ 1, se\ x=1 \\ fib(n-1)+fib(n-2), cc \end{cases}$$

   Verifique os resultados para os primeiros *n* números da sequência: 0,1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89,..

(fim de enunciado)