DEPARTAMENTO CIÊNCIA E TECNOLOGIA

Worksheet #3

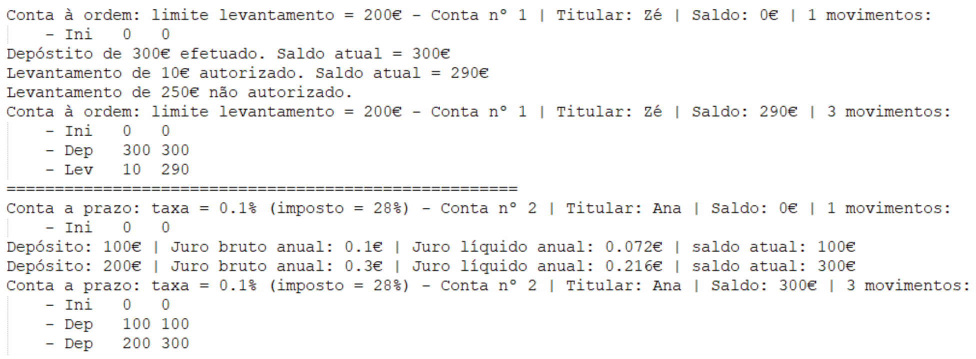
Modules
recursion

1. create a *package* `classesEscola` and inside adapt exercise 1 of the worksheet on "Classes, objects and methods" to use the organization by modules. In the end, the *package* must have the following files:

- **person.py** : contains all the code needed to define the Person class
- **collaborator.py** : contains all the necessary code for defining the Collaborator class , which inherits from Pessoa (*import* from the person module)
- **professor.py** : contains all the code necessary for defining the Professor class , which inherits from Collaborator (*import* from collaborator)
- **employee.py** : contains all the code necessary for defining the Employee class , which inherits from Collaborator (*import from the* Collaborator module)
- **student.py** : contains all the code needed to define the Student class , which inherits from Pessoa (*import* from module person)
- **main.py** : contains the *imports* and code needed to show the same output as at the end of the exercise in the previous sheet:

```
Professor Colaborador Gustavo Gomes (45) [987654321]: 800€ => Programação
Professor Colaborador Hugo Humberto (35) [912584684]: 800€ => Redes
Professor Colaborador Gustavo Gomes (45) [987654321]: 840.0€ => Programação
Professor Colaborador Hugo Horta (35) [912584684]: 800€ => Redes
Funcionário Colaborador Igor Ilhavo (32) [954785632]: 700€ => Bloco: A
Funcionário Colaborador Joana Jacinto (30) [974521365]: 700€ => Bloco: B
Funcionário Colaborador Igor Ilhavo (32) [954785632]: 752.5€ => Bloco: A
Aluno n° 1 - Laura Lis (22) [957432658] => 1° ano do curso de Programador
Aluno n° 2 - Miguel Moreira (22) [916845239] => 3° ano do curso de Adm. Redes
Aluno n° 2 - Mário Moreira (22) [916845239] => 3° ano do curso de Adm. Redes
```

2. create a *package* `classesBanco` and inside adapt exercise 2 of the same form to use the organization by modules. In the end, when running the `main.py file` , you should get the same output:



```
Conta à ordem: limite levantamento = 200€ - Conta n° 1 | Titular: Zé | Saldo: 0€ | 1 movimentos:
    - Ini   0   0
Depóstito de 300€ efetuado. Saldo atual = 300€
Levantamento de 10€ autorizado. Saldo atual = 290€
Levantamento de 250€ não autorizado.
Conta à ordem: limite levantamento = 200€ - Conta n° 1 | Titular: Zé | Saldo: 290€ | 3 movimentos:
    - Ini   0   0
    - Dep   300 300
    - Lev   10  290
=====================================================
Conta a prazo: taxa = 0.1% (imposto = 28%) - Conta n° 2 | Titular: Ana | Saldo: 0€ | 1 movimentos:
    - Ini   0   0
Depósito: 100€ | Juro bruto anual: 0.1€ | Juro líquido anual: 0.072€ | saldo atual: 100€
Depósito: 200€ | Juro bruto anual: 0.3€ | Juro líquido anual: 0.216€ | saldo atual: 300€
Conta a prazo: taxa = 0.1% (imposto = 28%) - Conta n° 2 | Titular: Ana | Saldo: 300€ | 3 movimentos:
    - Ini   0   0
    - Dep   100 100
    - Dep   200 300
```

3. Create recursive functions that implement the following features:

3.1. Print values from n to 0. To call the function, the code must ask the user for the value n and ensure that it is not less than 0.

Example:

Input	Output
3	3 2 1 0

3.2. Consecutively sums values from 1 to n. To call the function, the code must ask the user for the value n and ensure that it is not less than 0.

Example:

Input	Output
3	`6 # (3+2+1)`

3.3. Performs a multiplication by making successive sums. To call the function, the values must be asked from the user and it must be guaranteed that none of them is less than 1.

Input	Output
3 4	`12 # (4+4+4)`

1.1. Place the developed recursive functions in a file (module) `recursivas.py` . Create a file `main.py` , which contains code to show the user a menu that allows him to choose which recursive function he wants to use from that module.

1.2. Create a *package* called `funcoes` and place the `recursivas.py module` inside that *package* . The `main.py` file must stay outside the package, but keep the functionality.