# Worksheet #2

***Classes***

***Objects***

***Methods***

1. A school intends to computerize its system, which stores information about employees and students. For all people it is intended to save: name (first name and last name), date of birth and telephone number. It should be possible to get people's ages. For employees, it is also intended to save the salary. There are two types of contributors: faculty and staff. For teachers, it is also necessary to keep the subject they teach. It should be possible to assign a percentage raise to any employee. For the remaining employees, it is still necessary to keep the workbook. For students, it is also necessary to save the student number (generated sequentially), the course they attend and the year they are in. It must be possible, at any time, to view the textual description of any of the persons stored in the register. It is intended to computerize this system using object-oriented programming in Python. The necessary classes must be created, with the necessary components to comply with the functionalities described above, as well as two objects for each class, to test the referred functionalities.
   1. Person Class:
      1. Create method \_ \_ init \_ \_ with attributes first name, last name, date of birth and phone
      2. Create the method age , which returns the person's age, according to their date of birth and the current date [[1]](#footnote-1), [[2]](#footnote-2). It should be possible to use this method as if it were a variable
      3. Create getter and setter for name. The setter takes a full name (separated by space) and fills in the first name and last name
      4. \_ \_ str \_ \_ method to return a Person's textual description in the following format (the name must use the getter ):

Name (age) [phone]

* 1. Collaborator Class:
     1. The Collaborator class inherits from the Person class.
     2. \_ \_ init \_ \_ method that calls the superclass constructor and also sets the salary attribute
     3. Create the method that returns a collaborator's textual description, using the superclass's textual description and dealing with the subclass's specific variables , in the following format

Employee <textual description of Person>: salary €

* + 1. Create the method assignIncrease ( pc ), which assigns a percentage increase to the employee.
  1. Class Teacher
     1. The class inherits from the Collaborator class
     2. \_ \_ init \_ \_ method that calls the superclass constructor and also sets the discipline attribute
     3. Create two teachers:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **name** | **Birth date** | **telephone** | **Wage** | **discipline** |
| **pr1** | Gustavo Gomes | date( 1977,1,1) | 987654321 | 800 | Schedule |
| **pr2** | Hugo Humberto | date( 1987,3,8) | 912584684 | 800 | Networks |

* + 1. Create the method that returns the textual description of a teacher, resorting to the textual description of the superclass and dealing with the subclass specific variables and test for the two existing teachers. Example:



* + 1. Assign a 5% raise to teacher p1 (salary should now be 840)
    2. Change teacher name pr2 to Hugo Horta
  1. Employee Class
     1. The class inherits from the Collaborator class
     2. \_ \_ init \_ \_ method that calls the superclass constructor and also sets the block attribute
     3. Create two employees:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **name** | **Birth date** | **telephone** | **wage** | **block** |
| **f1** | Igor Ílhavo | date( 1990,12,12) | 954785632 | 700 | A |
| **f2** | Joana Jacinto | date( 1992,2,12) | 974521365 | 700 | B |

* + 1. Create the method that returns the textual description of an employee, calling the superclass's textual description and handling the subclass-specific variables, and testing for the two existing employees. Example:



* + 1. Assign a 7.5% raise to employee f1 (salary should now be 752.5)
    2. Change the f2 employee name to Joana Jardim
  1. Student Class
     1. The class inherits from the Person class
     2. nrStudents class variable , which starts at 0
     3. \_ \_ init \_ \_ method that calls the superclass constructor and also defines the attributes nrStudent (sequentially generated), course and year
     4. Create two students:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **name** | **Birth date** | **telephone** | **course** | **year** |
| **to 1** | laura lis | date( 2000,1,1) | 957432658 | Programmer | 1 |
| **a2** | Miguel Moreira | date( 2000,5,21) | 916845239 | Adm. Networks | 3 |

* + 1. Create the method that returns a student's textual description, calling the superclass's textual description and dealing with subclass-specific variables, and testing for the two existing students. Example:



* + 1. Change the name of student a2 to “Mário Moreira”.

In the end, it should be possible to run the following code, obtaining the output shown below.

Code:



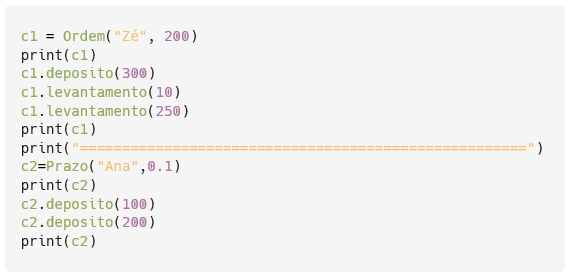
Output:

Text

Description automatically generated

1. A bank wants to computerize its accounts. In general, for each account, it is intended to save the account number (generated sequentially), the name of the holder, the balance (initial value: 0), the movements (list of tuples (type, value, balance)) and the number of movements (calculated from the movements). There are two types of account: current accounts and term accounts. For current accounts, it is also necessary to keep the withdrawal amount limit. These accounts allow you to make deposits (no limit) and withdrawals (with a defined maximum amount). All term accounts have an interest tax of 28%. For this type of account it is still necessary to save the interest rate. These accounts only allow deposits. Create the necessary Python code to run the following code, obtaining the output shown below.

Code:



Output:

Graphical user interface, application

Description automatically generated

1. You can use datetime (date) package : <https://docs.python.org/3/library/datetime.html> [↑](#footnote-ref-1)
2. The ages shown in the output examples were calculated on 12/20/2022 [↑](#footnote-ref-2)