Practical 5

Functions

1. Create a function that takes 3 integers as arguments and returns their average value.

2. Create a function that takes a list of integers as an argument and returns the number of even values in the given list.

3. Create a function that takes **password** of type String as an argument and check if the given **password** corresponds to the password rules, returning True or False, accordingly:

   Password rules:
   The length of the password should be at least 10
   The password should contain at least 2 numbers (try to find how to check this on your own)

4. Create a function that takes a required argument **name** of type String and an optional argument **greeting** with a default value "Welcome to our company!". The function prints "**name**, **greeting**", using the values of the arguments **name** and **greeting**.

5. Create a function that takes one required argument **name** of type String and an undefined number of optional non-keyword arguments. The function should calculate the average value of the optional arguments (if at least one optional argument is given) and print the following: if at least one optional argument is given print "**X**, your average grade is: **Y**", where **X** is the value of the argument **name** and **Y** is the average value of the optional arguments. Otherwise, print "No grades available for **X**", where **X** is the value of the argument **name.**

6. Create a function that takes one required argument **user** of type String and an undefined number of optional non-keyword arguments. If the value of the argument **user** is "admin", the function should print all the optional arguments in the following format:

   "Argument1 name": "argument1 value"
   "Argument2 name": "argument2 value"
   "Argument3 name": "argument3 value"
   etc.
   Otherwise, if the value of the argument **user** is not "admin", the function should print "access denied to the user X", where X is the value of the argument **user.**

Modules

7. Create the python file **calc.py** that contains the following 2 functions:
   calculate_cube(x) - which takes an integer **x** as an argument and returns the cube of the given number x^3;
   calculate_square(x) - which takes an integer **x** as an argument and returns the square of the given number x^2;
   Create the python file **pretty_print.py** that contains the following 2 functions:
   simple_print(x) - which takes an integer **x** as an argument and prints "Result: **x**", using the value of the argument **x**;
   pro_print(x) - which takes an integer **x** as an argument and prints "The result of the operation is **x**", using the value of the argument **x**;
   You should create a function **main()** inside the **calc.py** file. Inside the **main** function, you should calculate the square of the number 2, using the appropriate function and print the result using the function **simple_print(2)**. You should then calculate the cube of the number 4, using the appropriate function and print the result using the function **pro_print(4)**.

Homework 5

1. Create the function **my_max** which gets an undefined number of non-keyword arguments and returns the maximum of those. In case the function is called without arguments, it should return the text "no numbers given". Don't use the in-built functions for calculating the maximum value.

2. Create a function that gets a list as an argument and returns a new list, which contains only the unique values from the original list.

3. Create a function **my_fib** which takes an integer **n** as an argument and calculates and returns the n-th fibonacci number. Don't use recursion.
   Note: fib(n) = fib(n-1) + fib(n-2)

4. Create a function **my_func** that takes one integer type argument **n** and returns **n+10** if the value is less than 10 and **n-10**, otherwise. Create a list **list1** containing integer values of your choice. Use the **map()** function on **list1** to apply the function **my_func** on each of the list values and get a new list.

5. Create the modules **<u>Productcheck.py</u>** and **<u>Customer.py</u>** following the instructions below:

<u>Productcheck.py</u>

At the beginning of the file, outside the functions, create the dictionary products = {“candy”: 10, “juice”: 5, “pen”: 50}.

check(product, num): Gets the name of the product and its quantity as arguments and checks if the product with the given quantity is present in the dictionary **products** (you should check if there is a key with the product name and if the value at that key is >= the given quantity). The function should return True or False accordingly.

<u>Customer.py</u>

buy(product, num, price): Gets the name of the purchased product, the quantity and the price as arguments and checks if the product with the given quantity is present using the function **check(product, num)** from the module **<u>Productcheck</u>**. If the product with the given quantity is present, the function prints “You bought ***product*** and spent ***num*price***”, otherwise it prints “Sorry! We are out of this product.”:

main(): The function calls the function **buy(product, num, price)** with some values of your choice.