import math

from datetime import datetime

from datetime import date

import pytz

#Ex 1

def get\_sum(a, b):

print('Suma numerelor introduse este', a + b)

return a + b

get\_sum(4, 6)

get\_sum(110, 458)

#Ex 2

def get\_evenorodd\_num(num):

if num % 2 == 0:

print(f'Numarul {num} este par.')

return True

else:

print(f'Numarul {num} este impar.')

return False

get\_evenorodd\_num(19)

get\_evenorodd\_num(30)

#Ex 3

def length\_of\_myname(nume, prenume, nume\_mijlociu):

total = len(nume) + len(prenume) + len(nume\_mijlociu)

print(f'Numele complet are {total} caractere.')

return total

length\_of\_myname('Caprar', 'Silviu', 'Raul')

length\_of\_myname('Muntean', 'Aurelian', 'Paraschiv')

#Ex 4

def area\_of\_rectangle(length, width):

area = length \* width

print(f'Aria dreptunghiului este {area}.')

return area

area\_of\_rectangle(24, 12)

area\_of\_rectangle(9, 6)

#Ex 5

def area\_of\_circle(radius):

area = math.pi \* radius \* radius

print(f'Aria cercului este {area}.')

return area

area\_of\_circle(10)

area\_of\_circle(22)

#Ex 6

def get\_char(c):

var = 'Life is a great adventure.'

if c in var:

print(f'Caracterul {c} se gaseste in string-ul dat.')

return True

else:

print(f'Caracterul {c} nu se gaseste in string-ul dat.')

return False

get\_char('L')

get\_char('z')

#Ex 7

def lower\_upper\_chars(string1):

lower\_chars = 0

upper\_chars = 0

for i in string1:

if i.islower():

lower\_chars += 1

elif i.isupper():

upper\_chars += 1

print('Nr de caractere lower case este', lower\_chars)

print('Nr de caractere upper case este', upper\_chars)

lower\_upper\_chars('Ala bala PortocaLA')

lower\_upper\_chars('AnA Are MerE mUlTe si MarunTe')

#Ex 8

def pos\_nums\_list(positivenums):

for num in reversed(positivenums):

if num < 0:

positivenums.remove(num)

return positivenums

mynums = [1, 3, -1, -10, 7, -5, 17]

print(pos\_nums\_list(mynums))

#Ex 9

def get\_func(a, b):

if a > b:

print(f'Primul numar {a} este mai mare decat al doilea nr {b}.')

elif b > a:

print(f'Al doilea numar {b} este mai mare decat primul nr {a}.')

else:

print('Numerele sunt egale.')

get\_func(40, 10)

get\_func(14, 29)

get\_func(15, 15)

#Ex 10

def num\_in\_set(num, set1):

if num not in set1:

set1.add(num)

print('Am adaugat numarul nou in set.')

return True

else:

print('Nu am adaugat numarul in set. Acesta exista deja.')

return False

myset = {7, -15, 'soare', 6.44, True, 'victorie'}

num\_in\_set('soare', myset)

num\_in\_set('drujba', myset)

#Ex 1 optional

def days\_in\_month(month):

list1 = ['january', 'march', 'may', 'july', 'august', 'october', 'december']

list2 = ['april', 'june', 'september', 'november']

if month in list1:

return 31

if month in list2:

return 30

if month == 'february':

return 28

print(f"Luna are {days\_in\_month('july')} zile.")

print(f"Luna are {days\_in\_month('february')} zile.")

#Ex 2 optional

def calculator(x, y):

a = x + y

print('Suma:', a)

b = x - y

print('Diferenta:', b)

c = x \* y

print('Inmultirea:', c)

d = x / y

print('Impartirea:', d)

return a, b, c, d

calculator(10, 2)

calculator(-10, 25)

#Ex 3 optional

def return\_dict(nums):

return {i: nums.count(i) for i in nums }

mylist = [1, 2, 4, 4, 4, 5, 1]

anotherlist = [4, 0, 9, 3, 8, 9, 3, 8, 2, 9, 3, 8, 8]

print(return\_dict(mylist))

print(return\_dict(anotherlist))

#Ex 4 optional

def get\_largest(a, b, c):

return max(a, b, c)

print(get\_largest(15, 20, 43))

print(get\_largest(16, 16, 19))

#Ex 5 optional

def get\_sum\_to\_num(num):

sum = 0

for num in range(num+1):

sum += num

return sum

print(get\_sum\_to\_num(6)) # 0+1+2+3+4+5+6 = 21

print(get\_sum\_to\_num(10)) # 0+1+2+3+4+5+6+7+8+9+10 = 55

#Ex 1 optional bonus

def get\_common\_nums(list1, list2):

set(list1), set(list2)

return set(list1).intersection(set(list2))

mylist1 = [1, 1, 5, 2, 1, 5, 4, 2, 3, 9]

mylist2 = [0, 1, 2, 5, 6, 9, 3, 4, 2, 5]

print(get\_common\_nums(mylist1, mylist2))

#Ex 2 optional bonus

def offer\_discount(price, disc):

newprice = price - (disc \* price / 100)

if disc == 0 or disc > 99:

print('Reducerea introdusa este invalida.')

return ValueError

else:

print(f'Ai primit o reducere de {disc}%.')

return newprice

print(offer\_discount(100, 90))

print(offer\_discount(50, 100))

#Ex 3 optional bonus

def get\_datetime():

return datetime.now()

print(get\_datetime())

def get\_datetime\_china():

return datetime.now(pytz.timezone('Asia/Shanghai'))

print(get\_datetime\_china())

#Ex 4 optional bonus

def days\_to\_mybday(bday):

return bday - date.today()

print(days\_to\_mybday(date(2023, 3, 31)))

#Ex cu insert

def my\_strange\_list(n):

strange\_list = []

for i in range(0, n):

strange\_list.insert(2, i)

return strange\_list

print(my\_strange\_list(5))

# functia introduce intr-o lista numere de la 0 la n(argument-1 in cazul nostru 5-1)

# se face un for de la 0-4 in care i(elem curent in fiecare iteratie intra pe pozitia 2.

# i = 0 si i = 1 se afiseaza normal in fiecare iteratie.

# Insertul intervine pe pozitia 2 pt fiecare element de i >= 2 pana la 4.

# i = 2 si ramane pe pozitia 2.

# i = 3 si intra pe pozitia 2, dupa 1 iar 2 este mutat in dreapta pe pozitia 3.

# i = 4 si intra pe pozitia 2, dupa 1 iar 3 este mutat in dreapta pe pozitia 3 si 2 pe pozitia 4.

# daca insert ar intra pe poz 0 fiecare i anterior va fi mutat la dreapta dupa fiecare iteratie

# astfel ca intotdeauna valoarea curenta i va fi pe poz 0 deci devine o lista

# descrescatoare de la n la 0.