Universitatea Tehnică a Moldovei

Facultatea Calculatoare Informatică şi Microelectronică

Departamentul Ingineria Software și Automatică

**RAPORT**

Lucrarea de laborator nr. 4

La disciplina „Programarea Declarativa”

Tema: " Funcții de ordin înalt în Python”

A efectuat: st. gr. SI-211 A. Chihai

A verificat: V. Rusu

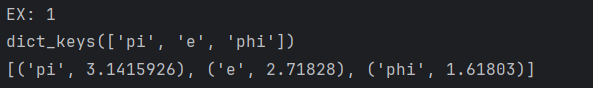
**Chișinău – 2023**

**Ex: 1**

Definiți un dicționar ce folosește șiruri de caractere pe post de chei și elemente float pe post de valori. Afișați doar cheile dicționarului și mai apoi tupluri formate din chei și valori.

print("EX: 1")  
mathNr = {  
 "pi": 3.1415926,  
 "e": 2.71828,  
 "phi": 1.61803  
}  
  
print(mathNr.keys())  
  
tuples = [(k, v) for k, v in mathNr.items()]  
print(tuples)

**Rezultat:**

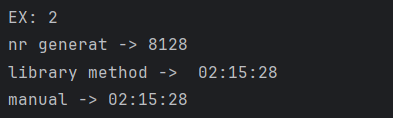


**Ex: 2**

Generați un număr aleator între 0 și 10000, ce reprezintă un număr de secunde. Calculați reprezentarea numărului de secunde în ore, minute și secunde și afișați rezultatul formatat sub forma hh:mm:ss. Alternativ, folosiți modulul datetime.

import math  
import random  
import time  
  
print("EX: 2")  
  
nr = random.randint(0, 10000)  
print(nr)  
timeConvertedLib = time.strftime("%H:%M:%S", time.gmtime(nr))  
print("library method -> ", timeConvertedLib)  
  
minutes = nr / 60  
if minutes >= 60:  
 hours = math.floor(minutes / 60)  
minutes = math.floor(minutes - hours \* 60)  
seconds = nr - (((hours \* 60) \* 60) + (minutes \* 60))  
  
time\_formatted = f"{hours:02d}:{minutes:02d}:{seconds:02d}"  
print(f"manual -> {time\_formatted}")

**Rezultat:**

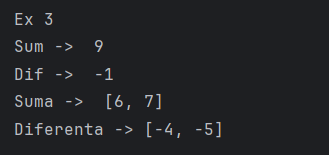
****

**Ex: 3**

Definiți două obiecte de tip float și calculați suma, diferența lor utilizand funcții lamda(). Apelați toate funcțiile din listă pe rând și cu alte tipuri de date, folosiți funcția map().

print("Ex 3")  
nr1 = 4  
nr2 = 5  
  
suma = lambda a, b: a + b  
diferenta = lambda a, b: a - b  
  
print("Sum -> ", suma(nr1, nr2))  
print("Dif -> ", diferenta(nr1, nr2))  
  
resSuma = list(map(suma, [2, 2], [nr1, nr2]))  
resDiferenta = list(map(diferenta, [0, 0], [nr1, nr2]))  
  
print(f"Suma -> {resSuma}")  
print(f"Diferenta -> {resDiferenta}")

**Rezultat:**

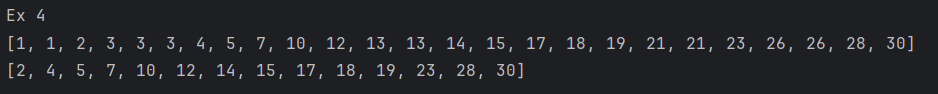
****

**Ex: 4**

Definiți o listă de valori întregi și afișați doar valorile distincte din aceasta (folosiți funcția filter()).

from random import randint  
print("Ex 4")  
arr = [randint(1, 30) for \_ in range(50)]  
print(sorted(arr))  
distinctElem = list(filter(lambda e: arr.count(e) == 1, arr))  
print(sorted(distinctElem))

**Rezultat:**

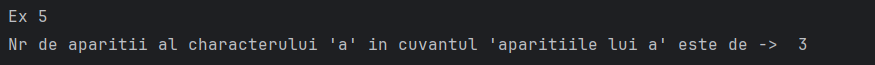
****

**Ex: 5**

Definiți o funcție ce returnează numărul de apariții ale unui caracter într-un string (folosiți funcția reduce()).

from functools import reduce  
print("Ex 5")  
def wordChar(word, charachter):  
 return reduce(lambda app, c: app + 1 if c == charachter else app, word, 0)  
  
print("Nr de aparitii al characterului 'a' in cuvantul 'aparitiile lui a' este de -> ",  
 wordChar("aparitiile lui a", "a"))

**Rezultat:**

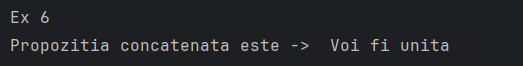


**Ex: 6**

Definiți o funcție ce concatenează oricâte stringuri sunt date la intrarea sa.

print("Ex 6")  
def stickyStrings(\*args):  
 return ' '.join(args)  
  
connected = stickyStrings("Voi", "fi", "unita")  
print("Propozitia concatenata este -> ", connected)

**Rezultat:**

****

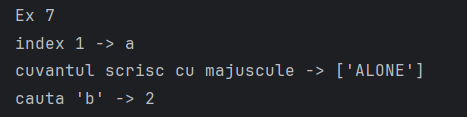
**Ex: 7**

Definiți o listă de funcții lambda ce returnează: al doilea caracter dintr-un string; stringul cu litere majuscule; poziția pe care se găsește un anumit caracter dat la intrare. Apelați toate funcțiile din listă pe rând

print("Ex 7")  
  
  
funList = [  
 lambda c: c[1] if len(c) > 1 else None,  
 lambda words: [word for word in words if word.isupper()],  
 lambda s, c: s.find(c) if isinstance(s, str) else None  
]  
  
wordIndex = "Laborator"  
findUppercase = ["HELlo", "ASFss", "UpPer", "ALONE"]  
findC = "b"  
  
for fun in funList:  
 if funList.index(fun) == 0:  
 print(f"index 1 -> {fun(wordIndex)}")  
 elif funList.index(fun) == 1:  
 print(f"cuvantul scrisc cu majuscule -> {fun(findUppercase)}")  
 else:  
 print(f"cauta 'b' -> {fun(wordIndex, findC)}")

print(fun(wordIndex, findC))

**Rezultat:**

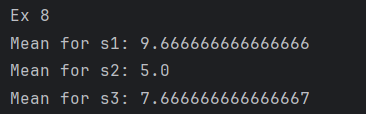
****

**Ex: 8**

Definiți o funcție ce calculează media a trei note sprecificate la intrare. Dacă la apel nu se trimit toate notele, se vor folosi valori implicite egale cu 4. Apelați funcția cu diferite combinații de argumente poziționale și keyword.

print("Ex 8")  
def mean3Grades(gradesList):  
 while len(gradesList) < 3:  
 gradesList.append(4)  
  
 return sum(gradesList) / 3  
  
grades = {  
"s1": [10, 10, 9],  
"s2": [5, 6],  
"s3": [8, 6, 9]  
}  
  
for student, sGrades in grades.items():  
 print(f"Mean for {student}: {mean3Grades(sGrades)}")

**Rezultat:**

****

**Ex: 9**

Definiți o funcție recursivă ce afișează suma primelor N numere naturale.

print("Ex9")  
  
def printSumNr(limit):  
 if limit == 1:  
 return 1  
 return limit + printSumNr(limit - 1)  
  
limit = 10  
print(f"Suma primelor {limit} numere este {printSumNr(limit)}")

**Rezultat:**

****

**Concluzie:**

Lucrarea de laborator nr. 4 la disciplina "Programarea Declarativa" a acoperit diverse concepte legate de programarea în Python, cu accent pe funcțiile de ordin înalt și utilizarea funcțiilor lambda, map, filter, reduce, precum și argumentele implicite în funcții.