**MINISTERUL EDUCAŢIEI ȘI CERCETĂRII AL REPUBLICII MOLDOVA**

**Universitatea Tehnică a Moldovei**

**Facultatea Calculatoare, Informatică şi Microelectronică**

**Departamentul Ingineria Software și Automatică**

**Programul de studii: Tehnologia informației**

RAPORT

# LUCRARE DE LABORATOR NR. 4

# la Programarea Declarativă

**Tema: Introducere în Python***.*

A efectuat:

st. gr. TI-211 Popa Cătălin

A verificat: lect.dr. Mariana Rusu

UTM, Chișinău 2023

**Tema**

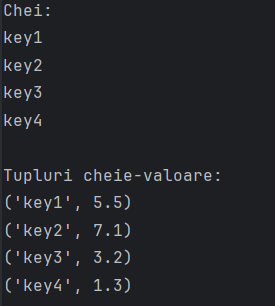
Introducere în Python. Definirea și apelarea funcțiilor de ordin înalt *map*(), *filter*(), *reduce*() și a funcțiilor anonime – *lambda*().

**Exercițiul 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.

dictionar = {  
 "key1": 5.5,  
 "key2": 7.1,  
 "key3": 3.2,  
 "key4": 1.3  
}  
  
print("Chei:")  
for key in dictionar.keys():  
 print(key)  
  
print("\nTupluri:")  
for key, value in dictionar.items():  
 tuplu = (key, value)  
 print(tuplu)

**Raspuns:**



**Exercițiul 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.

**Utilizând modulul datetime**

import random  
from datetime import timedelta  
  
num\_secunde = random.randint(0, 10000)  
  
s = timedelta(seconds=num\_secunde)  
  
ore = s.seconds // 3600  
minute = (s.seconds % 3600) // 60  
secunde = s.seconds % 60  
formatat = f"{ore:02d}:{minute:02d}:{secunde:02d}"  
  
print(num\_secunde)  
print(formatat)

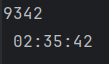
**Răspuns:**



**2. Utilizând modulul random și aritmetica standard.**

import random  
  
num\_secunde = random.randint(0, 10000)  
ore = num\_secunde // 3600  
minute = (num\_secunde % 3600) // 60  
secunde = num\_secunde % 60  
  
formatat = f"{ore:02d}:{minute:02d}:{secunde:02d}"  
  
print(num\_secunde)  
print(formatat)

**Răspuns:**

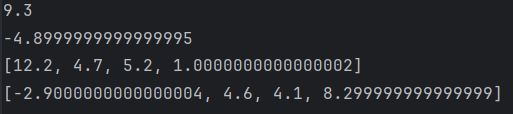


**Exercițiul 3**

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()*.

num1 = 2.2  
num2 = 7.1  
suma = lambda x, y: x + y  
diferenta = lambda x, y: x - y  
suma1 =suma(num1, num2)  
diferenta1 = diferenta(num1, num2)  
  
num3 = [10, 2.5, 3, -1.2]  
  
# Apelarea funcțiilor lambda utilizând map()  
suma2 = list(map(lambda x: suma(num1, x), num3))  
diferenta2 = list(map(lambda x: diferenta(num2, x), num3))  
   
print(suma1)  
print(diferenta1)  
print(suma2)  
print(diferenta2)

**Răspuns:**



**Exercițiul 4**

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

num = [4,4,1,7,43,3,3,1,1,6]  
distinct = list(filter(lambda x: num.count(x) == 1, num))  
print(distinct)

**Răspuns:**



**Exercițiul 5**

Definiți o funcție ce returnează numărul de apariții ale unui caracter într-un string (folosiți funcția *reduce()*). (aplica pe mai multe valori si afiseaza doar un raspus)

from functools import reduce  
def rep(șir, caracter):  
 count = reduce(lambda x, y: x + 1 if y == caracter else x, șir, 0)  
 return count  
sir = "catalin"  
variable = 'a'  
rezultat = rep(sir, variable)  
print(f"Apariții ale '{variable}' este: {rezultat}")

**Răspuns:**



**Exercițiul 6**

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

def con(\*text):  
 rez = ""   
 for s in text:  
 rez += s  
 return rez  
  
s1 = "Salut, "  
s2 = "ce "  
s3 = "faci?"  
  
rez = con(s1, s2, s3)  
print(rez)

**Răspuns:**

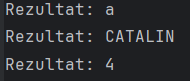


**Exercițiul 7**

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.

functii = [  
 lambda s: s[1],  
 lambda s: s.upper(),  
 lambda s, c: s.find(c)]  
  
text = "Catalin"  
  
for x in functii:  
 if x.\_\_code\_\_.co\_argcount == 1:  
 rez = x(text)  
 print(f"Rezultat: {rez}")  
 elif x.\_\_code\_\_.co\_argcount == 2:  
 var = 'l'  
 rez = x(text, var)  
 print(f"Rezultat: {rez}")

**Răspuns:**



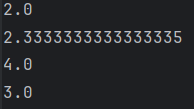
**Unesc cate valori nu oi da pentr ua afisa media**

**Exercițiul 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.

def media(nota1=4, nota2=4, nota3=4):  
 media = (nota1 + nota2 + nota3) / 3  
 return media  
  
m1 = media(1, 2, 3)  
m2 = media(1, 2)  
m3 = media()  
m4 = media(nota1=2, nota3=3)  
  
print(m1)  
print(m2)  
print(m3)  
print(m4)

**Răspuns:**



**Exercițiul 9**

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

def suma(n):  
 if n == 1:  
 return 1  
 else:  
 return n + suma(n - 1)  
  
n = 5  
rez = suma(n)  
print(rez)

**Răspuns:**



**Concluzie:**

În concluzie, raportul prezintă o serie de sarcini variate rezolvate în limbajul Python, evidențiind versatilitatea și puterea acestui limbaj de programare. Sarcinile acoperă aspecte precum lucrul cu dicționare, manipularea datelor și timpului, utilizarea funcțiilor lambda și a funcțiilor de înalt nivel, precum map, filter și reduce, precum și gestionarea valorilor implicite în funcții. Aceste exerciții demonstrează abilitățile fundamentale de programare în Python și capacitatea sa de a aborda diferite tipuri de probleme.