**1MPR04\_Simona\_Bļinova sb24037**

Visos uzdevumos lietoju savu moduli ar funkcijām. (programmu mapē *funkcijas.py*)

**1.uzdevums**

Programma ar funkcijām masīva izveidei, aizpildīšanai ar datiem, izvadei un mazāka un lielāka atrašanai.

**Kods:**

Funkcijas no *funkcijas.py*:

# Funkcija pārbaudei, lai skaitlis būtu reāls skaitlis

def reals\_skaitlis(a):

# meģinajumu skaitītājs (3 meģinājumi pareizi ievadīt)

meginajumi = 1

while meginajumi <= 3:

try:

a = float(a)

return float(a)

except:

if meginajumi < 3:

meginajumi += 1

a = input('Ievadiet reālo skaitli vēlreiz --> ')

else:

print('Programma beidz darbību!')

exit()

*1MPR04\_1\_Simona\_Blinova.py*:

import numpy

import funkcijas

# Masīva izveides funkcija

def masiva\_izveide(n):

return numpy.arange(n)

# Datu ievades masīvā funkcija

def datu\_ievade\_masiva(a):

garums = len(a)

for i in range(garums):

vertiba = input('Ievadiet vērtību --> ')

# pārbaude lai reāls, lai varētu atrast min un max

vertiba = funkcijas.reals\_skaitlis(vertiba)

a[i] = vertiba

return a

# Datu izvades funkcija

def datu\_izvade(a):

print(a)

# Masīva mazākas vērtības atrašanas funkcija

def mazaka\_vertiba(a):

garums = len(a)

for i in range(garums):

if i == 0:

mazakais = a[i]

else:

if mazakais > a[i]:

mazakais = a[i]

return mazakais

# Masīva lielākas vērtības atrašanas funkcija

def lielaka\_vertiba(a):

garums = len(a)

for i in range(garums):

if i == 0:

lielakais = a[i]

else:

if lielakais < a[i]:

lielakais = a[i]

return lielakais

n = input('Ievadiet elementu skaitu masīvā --> ')

n = funkcijas.reals\_skaitlis(n)

masivs = masiva\_izveide(n)

masivs = datu\_ievade\_masiva(masivs)

datu\_izvade(masivs)

print(f'Mazāka vērtība masīva: {mazaka\_vertiba(masivs)}')

print(f'Lielāka vērtība masīva: {lielaka\_vertiba(masivs)}')

Testa piemērs(1)

A white background with black text

AI-generated content may be incorrect.

Testa piemērs(2)

A screenshot of a computer

AI-generated content may be incorrect.

Testa piemērs(3)

A black text on a white background

AI-generated content may be incorrect.

**2.uzdevums**

Programma ar funkcijām masīva izveidei, aizpildīšanai ar datiem, izvadei un mazāka un lielāka atrašanai.

**Kods:**

Funkcijas no *funkcijas.py*:

# Funkcija pārbaudei, lai skaitlis būtu naturāls skaitlis

def naturals\_skaitlis(a):

# meģinājumu skaita skaitītājs (3 meģinājumi ierakstam)

meginajumi = 1

while meginajumi <= 3:

try:

a = int(a)

if a > 0:

return int(a)

else:

raise Exception

except:

if meginajumi < 3:

meginajumi += 1

a = input('Ievadiet naturālo skaitli vēlreiz --> ')

else:

print('Programma beidz darbību!')

exit()

*1MPR04\_2\_Simona\_Blinova.py*:

import funkcijas

import numpy

import math

def videjais\_aritmetiskais(a):

garums = len(a)

summa = 0

for i in range(garums):

summa += a[i]

return summa / garums

def videja\_kvadratiska\_vertiba(a):

garums = len(a)

summa = 0

for i in range(garums):

summa += a[i]\*a[i]

return math.sqrt(summa / garums)

def videja\_harmoniska\_vertiba(a):

garums = len(a)

summa = 0

for i in range(garums):

if a[i] != 0:

summa += 1 / a[i]

else:

garums -= 1

return garums / summa

def videja\_geometriska\_vertiba(a):

garums = len(a)

reizinajums = 1

for i in range(garums):

reizinajums \*= a[i]

return reizinajums \*\* (1/garums)

def videjas\_lineara\_novirze(a):

garums = len(a)

summa = 0

x = funkcijas.videjais\_aritmetiskais(a)

for i in range(garums):

summa += abs(a[i] - x)

return summa / garums

def standartnovirze(a):

garums = len(a)

summa = 0

x = funkcijas.videjais\_aritmetiskais(a)

for i in range(garums):

summa += (a[i] - x) \* (a[i] - x)

return math.sqrt(summa / garums)

n = input('Ievadiet skaitļu skaitu --> ')

n = funkcijas.naturals\_skaitlis(n)

saraksts = numpy.arange(n)

for i in range(n):

skaitlis = input('Ievadiet skaitli --> ')

skaitlis = funkcijas.naturals\_skaitlis(skaitlis)

saraksts[i] = skaitlis

# print(saraksts)

print(f'Vidējais aritmētiskais: {videjais\_aritmetiskais(saraksts)}')

print(f'Vidējais kvadrātiskais: {videja\_kvadratiska\_vertiba(saraksts)}')

print(f'Vidējais harmoniskais: {videja\_harmoniska\_vertiba(saraksts)}')

print(f'Vidējais ģeometriskais: {videja\_geometriska\_vertiba(saraksts)}')

print(f'Vidējā lineāra novirze: {videjas\_lineara\_novirze(saraksts)}')

print(f'Standartnovirze: {standartnovirze(saraksts)}')

Testa piemērs(1)

A screenshot of a computer code

AI-generated content may be incorrect.

Testa piemērs(2)

A screenshot of a computer

AI-generated content may be incorrect.

Testa piemērs(3)

A screen shot of a computer

AI-generated content may be incorrect.

**3.uzdevums**

Programma ar funkcijām masīva izveidei, aizpildīšanai ar datiem, izvadei un mazāka un lielāka atrašanai.

**Kods:**

Funkcijas no *funkcijas.py*:

# Funkcija pārbaudei, lai skaitlis būtu naturāls skaitlis

def naturals\_skaitlis(a):

# meģinājumu skaita skaitītājs (3 meģinājumi ierakstam)

meginajumi = 1

while meginajumi <= 3:

try:

a = int(a)

if a > 0:

return int(a)

else:

raise Exception

except:

if meginajumi < 3:

meginajumi += 1

a = input('Ievadiet naturālo skaitli vēlreiz --> ')

else:

print('Programma beidz darbību!')

exit()

# Funkcija pārbaudei, lai skaitlis būtu reāls skaitlis

def reals\_skaitlis(a):

# meģinajumu skaitītājs (3 meģinājumi pareizi ievadīt)

meginajumi = 1

while meginajumi <= 3:

try:

a = float(a)

return float(a)

except:

if meginajumi < 3:

meginajumi += 1

a = input('Ievadiet reālo skaitli vēlreiz --> ')

else:

print('Programma beidz darbību!')

exit()

*1MPR04\_3\_Simona\_Blinova.py*:

import numpy

import funkcijas

def videja\_sverta(x, y):

summa1 = 0

summa2 = 0

for i in range(len(x)):

summa1 += x[i]\*y[i]

summa2 += x[i]

return summa1 / summa2

n = input('Ievadiet masīvu izmēru --> ')

n = funkcijas.naturals\_skaitlis(n)

vertibas = numpy.arange(n)

skaits = numpy.arange(n)

for i in range(n):

vertiba = input('Ieadiet vērtību --> ')

vertiba = funkcijas.reals\_skaitlis(vertiba)

vertibas[i] = vertiba

vertibas\_skaits = input('Ieadiet vērtības skaitu --> ')

vertibas\_skaits = funkcijas.naturals\_skaitlis(vertibas\_skaits)

skaits[i] = vertibas\_skaits

print(f'Vidēja svērta vērtība: {videja\_sverta(vertibas, skaits)}')

Testa piemērs(1)

A computer screen shot of text

AI-generated content may be incorrect.

Testa piemērs(2)

A black and white text

AI-generated content may be incorrect.

Testa piemērs(3)

A screenshot of a computer code

AI-generated content may be incorrect.

**4.uzdevums**

Programma ar funkcijām masīva izveidei, aizpildīšanai ar datiem, izvadei un mazāka un lielāka atrašanai.

**Kods:**

Funkcijas no *funkcijas.py*:

# Funkcija pārbaudei, lai skaitlis būtu naturāls skaitlis

def naturals\_skaitlis(a):

# meģinājumu skaita skaitītājs (3 meģinājumi ierakstam)

meginajumi = 1

while meginajumi <= 3:

try:

a = int(a)

if a > 0:

return int(a)

else:

raise Exception

except:

if meginajumi < 3:

meginajumi += 1

a = input('Ievadiet naturālo skaitli vēlreiz --> ')

else:

print('Programma beidz darbību!')

exit()

# Funkcija pārbaudei, lai skaitlis būtu reāls skaitlis

def reals\_skaitlis(a):

# meģinajumu skaitītājs (3 meģinājumi pareizi ievadīt)

meginajumi = 1

while meginajumi <= 3:

try:

a = float(a)

return float(a)

except:

if meginajumi < 3:

meginajumi += 1

a = input('Ievadiet reālo skaitli vēlreiz --> ')

else:

print('Programma beidz darbību!')

exit()

# Vidēja aritmētiskā funkcija

def videjais\_aritmetiskais(a):

garums = len(a)

summa = 0

for i in range(garums):

summa += a[i]

return summa / garums

*1MPR04\_4\_Simona\_Blinova.py*:

import funkcijas

import numpy

import math

def linearas\_korelacijas\_koeficients(a, b, n):

x = funkcijas.videjais\_aritmetiskais(a)

y = funkcijas.videjais\_aritmetiskais(b)

summa1 = 0

summa2 = 0

summa3 = 0

for i in range(n):

summa1 += (a[i] - x) \* (b[i] - y)

summa2 += (a[i] - x) \* (a[i] - x)

summa3 += (b[i] - y) \* (b[i] - y)

if summa2 == 0 or summa3 == 0:

return 'Kļūda, dalīšana ar nulli'

return summa1 / math.sqrt(summa2 \* summa3)

n = input('Ievadiet masīvu izmēru --> ')

n = funkcijas.naturals\_skaitlis(n)

noverojumi1 = numpy.arange(n)

noverojumi2 = numpy.arange(n)

for i in range(n):

skaitlis = input('Ievadiet skaitli 1. masīvā --> ')

skaitlis = funkcijas.reals\_skaitlis(skaitlis)

noverojumi1[i] = skaitlis

skaitlis = input('Ievadiet skaitli 2. masīvā --> ')

skaitlis = funkcijas.reals\_skaitlis(skaitlis)

noverojumi2[i] = skaitlis

print(f'Lineāras korelācijas koeficients: {linearas\_korelacijas\_koeficients(noverojumi1, noverojumi2, n)}')

Testa piemērs(1)

A screen shot of a computer

AI-generated content may be incorrect.

Testa piemērs(2)

A computer screen shot of a computer code

AI-generated content may be incorrect.

Testa piemērs(3)

A close up of a text

AI-generated content may be incorrect.