**1MPR08\_Simona\_Bļinova sb24037**

**1.uzdevums**

Programma, kas izvada matricu tabulas veidā un atrod tās mazāko un lielāko elementu ar to atrašanas vietām.

**Kods:**

import numpy

def parbaude(a):

skaititajs = 1

while skaititajs <= 3:

try:

a = int(a)

if a > 9999 or a < -999:

raise Exception

else:

return int(a)

except:

skaititajs += 1

a = input('Ievadiet elementu vēlreiz --> ')

else:

print('Programma beidz darbību!')

exit()

def elementu\_ievade(a):

for i in range(len(a)):

for j in range(len(a[i])):

b = input('Ievadiet matricas elementu a('+str(i+1)+','+str(j+1)+') --> ')

b = parbaude(b)

a[i][j] = b

return a

def matricas\_min(a):

min\_elem = a[0][0]

rinda = 0

kolonna = 0

for i in range(len(a)):

for j in range(len(a[i])):

if min\_elem > a[i][j]:

min\_elem = a[i][j]

rinda = i

kolonna = j

return min\_elem, rinda, kolonna

def matricas\_max(a):

max\_elem = a[0][0]

rinda = 0

kolonna = 0

for i in range(len(a)):

for j in range(len(a[i])):

if max\_elem < a[i][j]:

max\_elem = a[i][j]

rinda = i

kolonna = j

return max\_elem, rinda, kolonna

def matricas\_izvade(a, g):

for i in range(len(a)):

virkne = ''

for j in range(len(a[i])):

skaits = g - len(str(a[i][j]))

virkne = virkne + ' '\*skaits

virkne = virkne + '{:.0f}'.format(a[i][j])

if j == len(a[i]) - 1:

print(virkne)

else:

virkne = virkne + ' '

rindas = int(input('Ievadiet matricas rindu skaitu --> '))

kolonnas = int(input('Ievadiet matricas kolonnu skaitu --> '))

matrica = numpy.empty((rindas, kolonnas))

matrica = elementu\_ievade(matrica)

#print(masivs)

minimums, min\_rinda, min\_kolonna = matricas\_min(matrica)

maksimums, max\_rinda, max\_kolonna = matricas\_max(matrica)

if len(str(minimums)) > len(str(maksimums)):

garums = len(str(minimums))

else:

garums = len(str(maksimums))

print(' ')

matricas\_izvade(matrica, garums)

print(' ')

print(f'Mazākais elements ir {int(minimums)}, un tas atrodas {min\_rinda+1}.rindas un {min\_kolonna+1}.kolonnas krustpunktā')

print(f'Lielākais elements ir {int(maksimums)}, un tas atrodas {max\_rinda+1}.rindas un {max\_kolonna+1}.kolonnas krustpunktā')

Testa piemērs(1)

A screen shot of a computer

AI-generated content may be incorrect.

Testa piemērs(2)

A screen shot of a computer

AI-generated content may be incorrect.

Testa piemērs(3)

A screen shot of a computer

AI-generated content may be incorrect.

**2.uzdevums**

Programma, kas veic divu matricu reizināšanu un atbilstošo divi matricu elementu reizināšanu, ja tas ir iespējams.

**Kods:**

import numpy

def parbaude(a):

skaititajs = 1

while skaititajs < 3:

try:

a = int(a)

return int(a)

except:

skaititajs += 1

a = input('Ievadiet elementu vēlreiz --> ')

else:

print('Programma beidz darbību!')

exit()

def elementu\_ievade(a):

for i in range(len(a)):

for j in range(len(a[i])):

b = input('Ievadiet matricas elementu a('+str(i+1)+','+str(j+1)+') --> ')

b = parbaude(b)

a[i][j] = b

return a

def elementu\_reizinajums(a, b):

n1 = a.shape[0]

m1 = a.shape[1]

n2 = b.shape[0]

m2 = b.shape[1]

if n1 == n2 and m1 == m2:

c = numpy.empty((n1, m1))

for i in range(n1):

for j in range(m1):

c[i][j] = a[i][j] \* b[i][j]

else:

c = numpy.zeros((1, 1))

c[0][0] = 0.1

return c

def matricu\_reizinajums(a, b):

n1 = a.shape[0]

m1 = a.shape[1]

n2 = b.shape[0]

m2 = b.shape[1]

if m1 == n2:

c = numpy.zeros((n1, m2))

for i in range(n1):

for j in range(m2):

for k in range(m1):

c[i][j] = c[i][j] + a[i][k] \* b[k][j]

else:

c = numpy.empty((1, 1))

c[0][0] = 0.1

return c

def matricas\_min(a):

min\_elem = a[0][0]

rinda = 0

kolonna = 0

for i in range(len(a)):

for j in range(len(a[i])):

if min\_elem > a[i][j]:

min\_elem = a[i][j]

rinda = i

kolonna = j

return min\_elem, rinda, kolonna

def matricas\_max(a):

max\_elem = a[0][0]

rinda = 0

kolonna = 0

for i in range(len(a)):

for j in range(len(a[i])):

if max\_elem < a[i][j]:

max\_elem = a[i][j]

rinda = i

kolonna = j

return max\_elem, rinda, kolonna

def matricas\_izvade(a):

minimums, min\_rinda, min\_kolonna = matricas\_min(a)

maksimums, max\_rinda, max\_kolonna = matricas\_max(a)

if len(str(minimums)) > len(str(maksimums)):

garums = len(str(minimums))

else:

garums = len(str(maksimums))

for i in range(len(a)):

virkne = ''

for j in range(len(a[i])):

skaits = garums - len(str(a[i][j]))

virkne = virkne + ' '\*skaits

virkne = virkne + '{:.0f}'.format(a[i][j])

if j == len(a[i]) - 1:

print(virkne)

else:

virkne = virkne + ' '

rindas1 = int(input('Ievadiet 1.matricas rindu skaitu --> '))

kolonnas1 = int(input('Ievadiet 1.matricas kolonnu skaitu --> '))

matrica1 = numpy.empty((rindas1, kolonnas1))

matrica1 = elementu\_ievade(matrica1)

rindas2 = int(input('Ievadiet 2.matricas rindu skaitu --> '))

kolonnas2 = int(input('Ievadiet 2.matricas kolonnu skaitu --> '))

matrica2 = numpy.empty((rindas2, kolonnas2))

matrica2 = elementu\_ievade(matrica2)

print(' ')

matricas\_izvade(matrica1)

print(' ')

matricas\_izvade(matrica2)

print(' ')

print('Matricas atbilstošo elementu reizinājums: ')

rez1\_matrica = elementu\_reizinajums(matrica1, matrica2)

if rez1\_matrica[0][0] == 0.1:

print('Nevar sareizināt atbilstošos elementus dažāda izmēra matricām.')

else:

matricas\_izvade(rez1\_matrica)

print(' ')

print('Matricu reizinājums: ')

rez2\_matrica = matricu\_reizinajums(matrica1, matrica2)

if rez2\_matrica[0][0] == 0.1:

print('Nevar sareizināt matricas ar šadiem izmēriem.')

else:

matricas\_izvade(rez2\_matrica)

Testa piemērs(1)

A computer screen shot of white text

AI-generated content may be incorrect.

Testa piemērs(2)

A screen shot of a computer

AI-generated content may be incorrect.

Testa piemērs(3)

A blue screen with white text

AI-generated content may be incorrect.

**3.uzdevums**

Programma, kas nosaka vai ievadīta matrica veido maģisko kvadrātu.

**Kods:**

import numpy

def parbaude(a):

skaititajs = 1

while skaititajs <= 3:

try:

a = int(a)

if a < 1:

raise Exception

else:

return int(a)

except:

skaititajs += 1

a = input('Ievadiet elementu vēlreiz --> ')

else:

print('Programma beidz darbību!')

exit()

def elementu\_ievade(a):

for i in range(len(a)):

for j in range(len(a[i])):

b = input('Ievadiet matricas elementu a('+str(i+1)+','+str(j+1)+') --> ')

b = parbaude(b)

a[i][j] = b

return a

def vai\_magiskais(a):

n = a.shape[0]

summas = []

# Horizontāles

for i in range(n):

s = 0

for j in range(n):

s += a[i][j]

summas.append(s)

# Vertikāles

for i in range(n):

s = 0

for j in range(n):

s += a[j][i]

summas.append(s)

# Galvenā diagonāle

s = 0

for i in range(n):

s += a[i][i]

summas.append(s)

# Otrā diagonāle

s = 0

for i in range(n-1, -1, -1):

s += a[i][i]

summas.append(s)

p = summas[0]

for i in range(1, len(summas)):

if p != summas[i]:

break

else:

return True, p

return False, p

def matricas\_min(a):

min\_elem = a[0][0]

rinda = 0

kolonna = 0

for i in range(len(a)):

for j in range(len(a[i])):

if min\_elem > a[i][j]:

min\_elem = a[i][j]

rinda = i

kolonna = j

return min\_elem, rinda, kolonna

def matricas\_max(a):

max\_elem = a[0][0]

rinda = 0

kolonna = 0

for i in range(len(a)):

for j in range(len(a[i])):

if max\_elem < a[i][j]:

max\_elem = a[i][j]

rinda = i

kolonna = j

return max\_elem, rinda, kolonna

def matricas\_izvade(a):

minimums, min\_rinda, min\_kolonna = matricas\_min(a)

maksimums, max\_rinda, max\_kolonna = matricas\_max(a)

if len(str(minimums)) > len(str(maksimums)):

garums = len(str(minimums))

else:

garums = len(str(maksimums))

for i in range(len(a)):

virkne = ''

for j in range(len(a[i])):

skaits = garums - len(str(a[i][j]))

virkne = virkne + ' '\*skaits

virkne = virkne + '{:.0f}'.format(a[i][j])

if j == len(a[i]) - 1:

print(virkne)

else:

virkne = virkne + ' '

n = int(input('Ievadiet matricas rindu un kolonnu skaitu --> '))

matrica = numpy.empty((n, n))

matrica = elementu\_ievade(matrica)

paz, summa = vai\_magiskais(matrica)

#print(paz, summa)

print(' ')

if paz == False:

print('Matrica neveido maģisko kvadrātu.')

matricas\_izvade(matrica)

else:

print('Matrica veido maģisko kvadrātu.')

jauna\_matrica = numpy.empty((n+2, n+2))

for i in range(n+2):

for j in range(n+2):

if i == 0 or i == n+1 or j == 0 or j == n+1:

jauna\_matrica[i][j] = summa

else:

jauna\_matrica[i][j] = matrica[i-1][j-1]

matricas\_izvade(jauna\_matrica)

Testa piemērs(1)

A screenshot of a computer

AI-generated content may be incorrect.

Testa piemērs(2)

A blue screen with white text

AI-generated content may be incorrect.

Testa piemērs(3)

A screenshot of a computer program

AI-generated content may be incorrect.

**4.uzdevums**

Programma, kas saskaita diagonāļu elementu summas.

**Kods:**

import numpy

def parbaude(a):

skaititajs = 1

while skaititajs < 3:

try:

a = int(a)

return int(a)

except:

skaititajs += 1

a = input('Ievadiet elementu vēlreiz --> ')

else:

print('Programma beidz darbību!')

exit()

def elementu\_ievade(a):

for i in range(len(a)):

for j in range(len(a[i])):

b = input('Ievadiet matricas elementu a('+str(i+1)+','+str(j+1)+') --> ')

b = parbaude(b)

a[i][j] = b

return a

def summu\_matrica(a):

n = a.shape[0]

b = numpy.zeros((n+1, n+1))

for i in range(n):

for j in range(n):

b[i][j] = a[i][j]

return b

def diagonalu\_summas(a, b):

n = a.shape[0]

m = b.shape[0]

k = 0

while k < n:

if k == 0:

for i in range(n):

b[m-1][m-1] += a[i][i]

else:

for i in range(n):

if i+k < n:

b[m-1-k][m-1] += a[i][i+k]

b[m-1][m-1-k] += a[i+k][i]

else:

break

k += 1

return b

def matricas\_min(a):

min\_elem = a[0][0]

rinda = 0

kolonna = 0

for i in range(len(a)):

for j in range(len(a[i])):

if min\_elem > a[i][j]:

min\_elem = a[i][j]

rinda = i

kolonna = j

return min\_elem, rinda, kolonna

def matricas\_max(a):

max\_elem = a[0][0]

rinda = 0

kolonna = 0

for i in range(len(a)):

for j in range(len(a[i])):

if max\_elem < a[i][j]:

max\_elem = a[i][j]

rinda = i

kolonna = j

return max\_elem, rinda, kolonna

def matricas\_izvade(a):

minimums, min\_rinda, min\_kolonna = matricas\_min(a)

maksimums, max\_rinda, max\_kolonna = matricas\_max(a)

if len(str(minimums)) > len(str(maksimums)):

garums = len(str(minimums))

else:

garums = len(str(maksimums))

for i in range(len(a)):

virkne = ''

for j in range(len(a[i])):

skaits = garums - len(str(a[i][j]))

virkne = virkne + ' '\*skaits

virkne = virkne + '{:.0f}'.format(a[i][j])

if j == len(a[i]) - 1:

print(virkne)

else:

virkne = virkne + ' '

n = int(input('Ievadiet matricas rindu un kolonnu skaitu --> '))

matrica = numpy.empty((n, n))

matrica = elementu\_ievade(matrica)

print(' ')

summas = summu\_matrica(matrica)

summas = diagonalu\_summas(matrica, summas)

matricas\_izvade(summas)

Testa piemērs(1)

A number on a black background

AI-generated content may be incorrect.

Testa piemērs(2)



Testa piemērs(3)

A number in a row

AI-generated content may be incorrect.