1MPR08_Simona_Bļinova sb24037

1.uzdevums

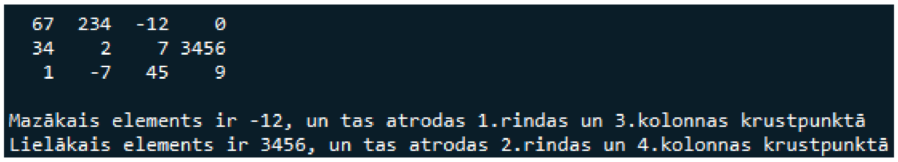
Programma, kas izvada matricu tabulas veidā un atrod tās mazāko un lielāko elementu ar to atrašanas vietām.

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)
```

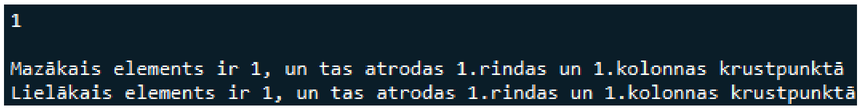
```
minimums, min_rinda, min_kolonna = matricas_min(matrica)
maksimums, max_rinda, max_kolonna = matricas_max(matrica)

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)



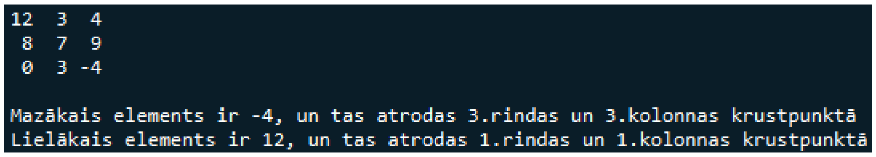
```
  67  234  -12    0
  34    2    7 3456
   1   -7   45    9

Mazākais elements ir -12, un tas atrodas 1.rindas un 3.kolonnas krustpunktā
Lielākais elements ir 3456, un tas atrodas 2.rindas un 4.kolonnas krustpunktā
```

Testa piemērs(2)



```
1

Mazākais elements ir 1, un tas atrodas 1.rindas un 1.kolonnas krustpunktā
Lielākais elements ir 1, un tas atrodas 1.rindas un 1.kolonnas krustpunktā
```

Testa piemērs(3)



```
12  3  4
 8  7  9
 0  3 -4

Mazākais elements ir -4, un tas atrodas 3.rindas un 3.kolonnas krustpunktā
Lielākais elements ir 12, un tas atrodas 1.rindas un 1.kolonnas krustpunktā
```

<u>**2.uzdevums**</u>

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

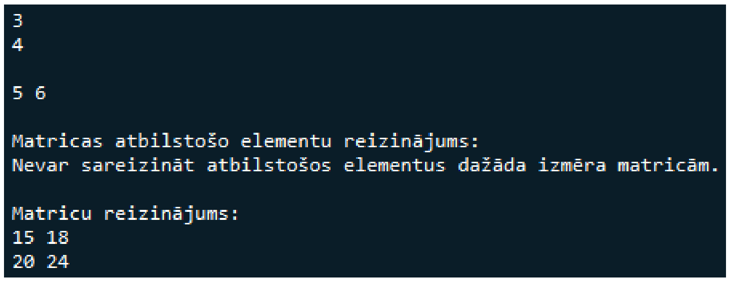
Kods:

```
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)
```

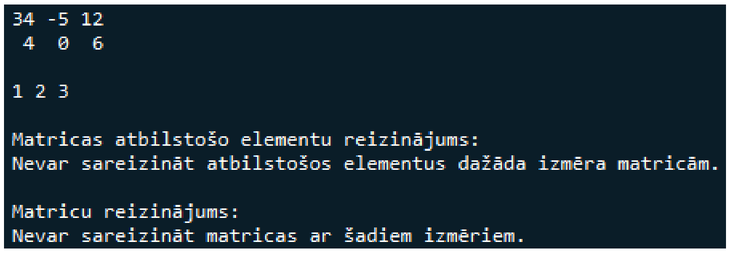
```
print(' ')

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)



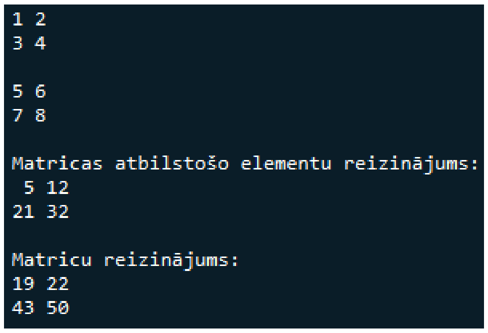
```
3
4

5 6

Matricas atbilstošo elementu reizinājums:
Nevar sareizināt atbilstošos elementus dažāda izmēra matricām.

Matricu reizinājums:
15 18
20 24
```

Testa piemērs(2)



```
34 -5 12
 4  0  6

1 2 3

Matricas atbilstošo elementu reizinājums:
Nevar sareizināt atbilstošos elementus dažāda izmēra matricām.

Matricu reizinājums:
Nevar sareizināt matricas ar šadiem izmēriem.
```

Testa piemērs(3)



```
1 2
3 4

5 6
7 8

Matricas atbilstošo elementu reizinājums:
 5 12
21 32

Matricu reizinājums:
19 22
43 50
```

3.uzdevums

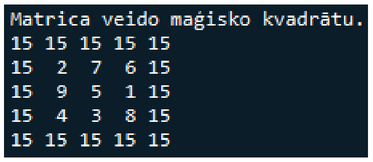
Programma, kas nosaka vai ievadīta matrica veido maģisko kvadrātu.

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
```

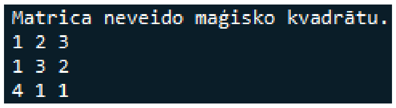
```
def vai_magiskais(a):
    n = a.shape[0]
    summas = []

    # 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)



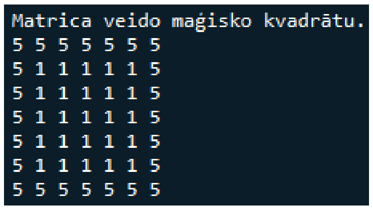
```
Matrica veido maģisko kvadrātu.
15 15 15 15 15
15  2  7  6 15
15  9  5  1 15
15  4  3  8 15
15 15 15 15 15
```

Testa piemērs(2)



```
Matrica neveido maģisko kvadrātu.
1 2 3
1 3 2
4 1 1
```

Testa piemērs(3)



```
Matrica veido maģisko kvadrātu.
5 5 5 5 5 5 5
5 1 1 1 1 1 5
5 1 1 1 1 1 5
5 1 1 1 1 1 5
5 1 1 1 1 1 5
5 1 1 1 1 1 5
5 5 5 5 5 5 5
```

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)
```

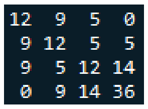
print(' ')

summas = summu_matrica(matrica)

summas = diagonalu_summas(matrica, summas)

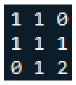
matricas_izvade(summas)

Testa piemērs(1)



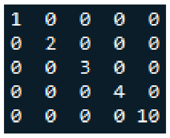
```
12  9  5  0
 9 12  5  5
 9  5 12 14
 0  9 14 36
```

Testa piemērs(2)



```
1 1 0
1 1 1
0 1 2
```

Testa piemērs(3)



```
1  0  0  0  0
0  2  0  0  0
0  0  3  0  0
0  0  0  4  0
0  0  0  0 10
```