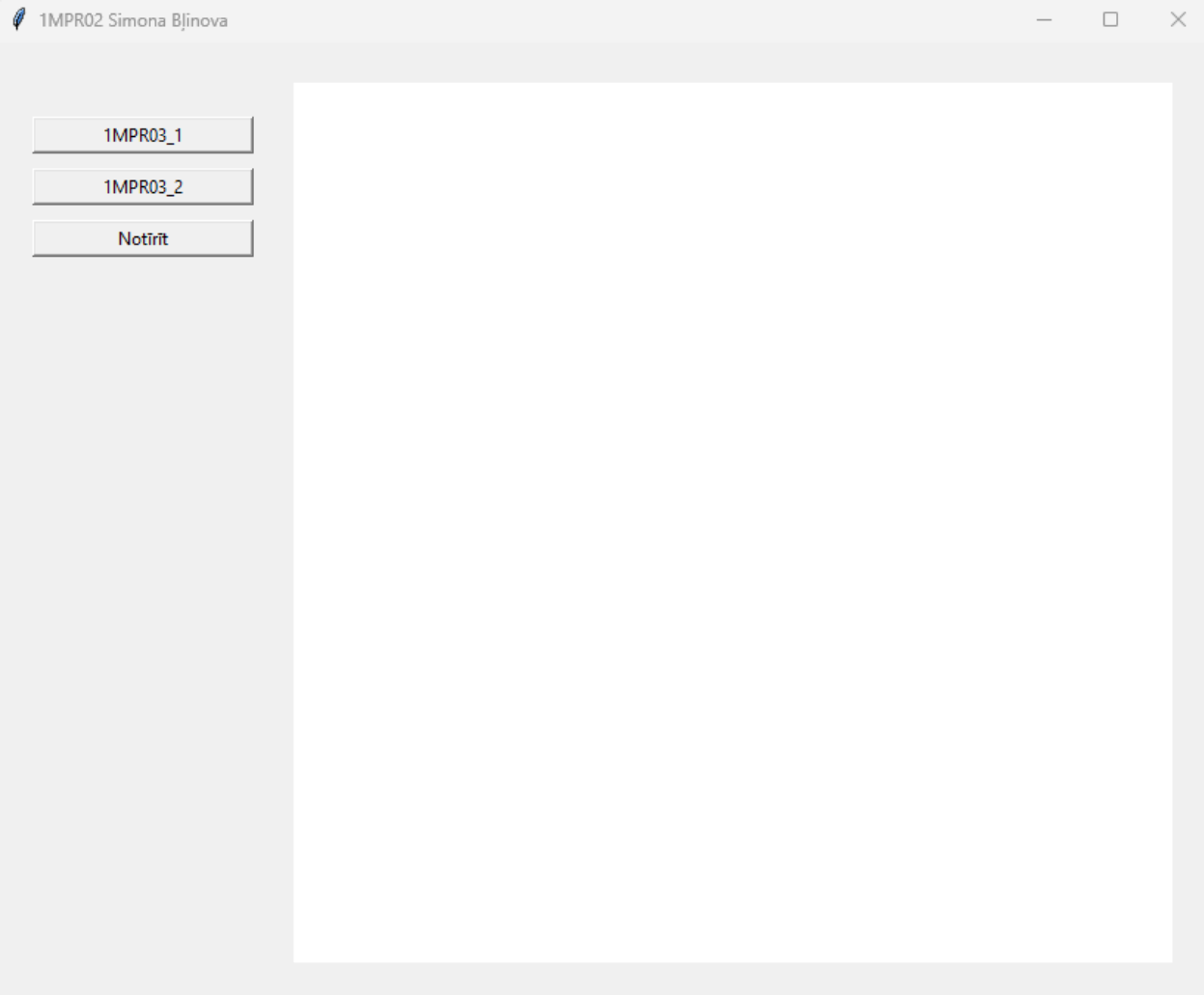
1MPR03_Simona_Bļinova sb24037

1.uzdevums un 2.uzdevums

Programma kas zīmē Serpinska paklāju un Koha zvaigzni.

Grafiska saskarne:



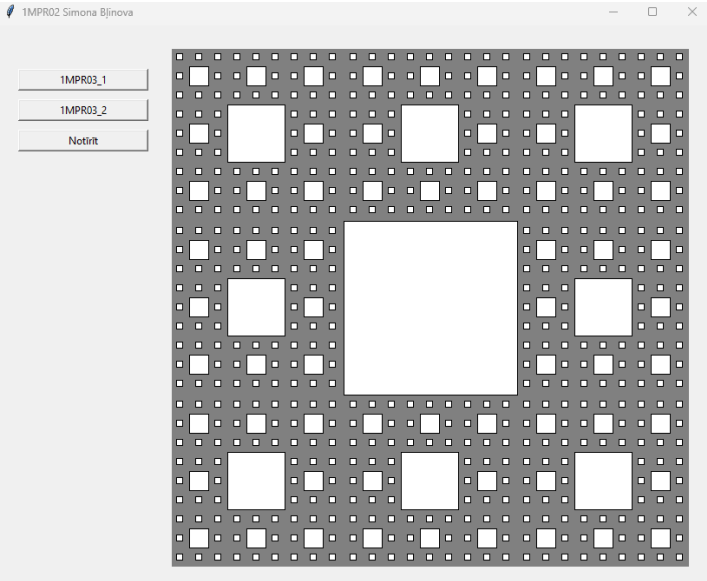
Kods:

```
import tkinter as tk
import math


def mpr1():
    garums_kanva = 600
    x = 0
    y = 0
    kanva.create_rectangle(0, 0, 600, 600, fill='grey')
```

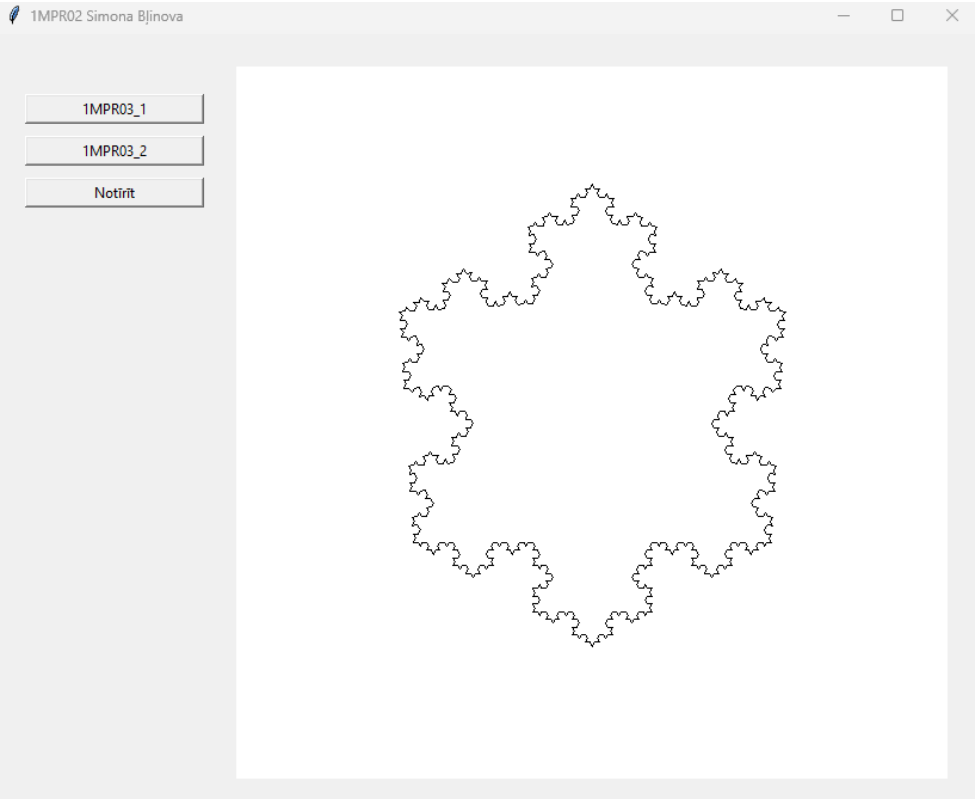
```
    kvadrati(x, y, garums_kanva)


def kvadrati(x, y, garums_kanva):
    garums = garums_kanva // 3
    x_balts = x + garums
    y_balts = y + garums
    kanva.create_rectangle(x_balts, y_balts, x_balts + garums, y_balts + garums, fill='white')
    if garums > 20:
        kvadrati(x, y, garums)
        kvadrati(x + garums, y, garums)
        kvadrati(x + garums*2, y, garums)
        kvadrati(x, y + garums, garums)
        kvadrati(x, y + garums*2, garums)
        kvadrati(x + garums, y + garums*2, garums)
        kvadrati(x + garums*2, y + garums, garums)
        kvadrati(x + garums*2, y + garums*2, garums)


def kohazvaigzne(kanva, virsotnex1, virsotney1, virsotnex2, virsotney2, pakape):
    if pakape == 0: # funkcijas pārtraukuma gadījums
        kanva.create_line(virsotnex1, virsotney1, virsotnex2, virsotney2, fill="black")
    else:
        x_garums = virsotnex2 - virsotnex1 # garums starp pēdejo un pirmo punktu
        y_garums = virsotney2 - virsotney1
        # aprēķina nākamos punktus
        x11 = virsotnex1 + x_garums / 3
        y11 = virsotney1 + y_garums / 3
        x22 = virsotnex1 + 2 * x_garums / 3
        y22 = virsotney1 + 2 * y_garums / 3
        x33 = (virsotnex1 + virsotnex2) / 2 - (virsotney2 - virsotney1)*(math.sqrt(3) / 6)
        y33 = (virsotney1 + virsotney2) / 2 + (virsotnex2 - virsotnex1)*(math.sqrt(3) / 6)
        # zīmē Koha līkni
        kohazvaigzne(kanva, virsotnex1, virsotney1, x11, y11, pakape - 1)
        kohazvaigzne(kanva, x11, y11, x33, y33, pakape - 1)
        kohazvaigzne(kanva, x33, y33, x22, y22, pakape - 1)
        kohazvaigzne(kanva, x22, y22, virsotnex2, virsotney2, pakape - 1)


def mpr2():
    virsotne_x1 = 300
    virsotne_y1 = 100
    virsotne_x2 = 150
    virsotne_y2 = 400
```

```
    virsotne_x3 = 450
    virsotne_y3 = 400
    kohazvaigzne(kanva, virsotne_x2, virsotne_y2, virsotne_x3, virsotne_y3, 4)
    kohazvaigzne(kanva, virsotne_x3, virsotne_y3, virsotne_x1, virsotne_y1, 4)
    kohazvaigzne(kanva, virsotne_x1, virsotne_y1, virsotne_x2, virsotne_y2, 4)


def notirit():
    kanva.delete('all')


logs = tk.Tk()
logs.geometry('825x650')
logs.title('1MPR02 Simona Bļinova')

kanva = tk.Canvas(logs, background='white')
kanva.place(x=200, y=25, height=600, width=600)

b1 = tk.Button(logs, text='1MPR03_1', command=mpr1)
b1.place(x=25, y=50, height=25, width=150)

b2 = tk.Button(logs, text='1MPR03_2', command=mpr2)
b2.place(x=25, y=85, height=25, width=150)

bnotirit = tk.Button(logs, text='Notīrīt', command=notirit)
bnotirit.place(x=25, y=120, height=25, width=150)

logs.mainloop()
```

Testa piemērs(1)



Testa piemērs(2)



3.uzdevums

Programma, kas saskaita dažādu krāsu pērlīšu skaitu n-tājā rindā.

Kods:

```
def perlisu_daudzums(rindas, zilas, sarkanas, oranzas):
    if rindas == 0:
        return zilas, sarkanas, oranzas

    zilas_rinda = zilas + 2 * sarkanas + 3 * oranzas
    sarkanas_rinda = 2 * zilas + 3 * sarkanas + oranzas
    oranzas_rinda = 3 * zilas + 2 * sarkanas + oranzas

    return perlisu_daudzums(rindas-1, zilas_rinda, sarkanas_rinda, oranzas_rinda)

rindas = int(input('Ievadiet rindu skaitu --> '))
krasa = input('Ievadiet sākuma krāsu (z, s, o) --> ')

zilas = 0
sarkanas = 0
oranzas = 0
```

```
match krasa:
  case 'z':
    zilas, sarkanas, oranzas = perlisu_daudzums(rindas, 1, sarkanas, oranzas)
  case 's':
    zilas, sarkanas, oranzas = perlisu_daudzums(rindas, zilas, 1, oranzas)
  case 'o':
    zilas, sarkanas, oranzas = perlisu_daudzums(rindas, zilas, sarkanas, 1)
  case _:
    print('Tādas krāsas nav!')

print(f'{rindas}. rindā zilo pērlīšu skaits ir {zilas}, sarkano - {sarkanas}, oranžo - {oranzas}.')
```

Testa piemērs(1)

```
Ievadiet rindu skaitu --> 1
Ievadiet sākuma krāsu (z, s, o) --> z
1. rindā zilo pērlīšu skaits ir 1, sarkano - 2, oranžo - 3.
```

Testa piemērs(2)

```
Ievadiet rindu skaitu --> 12
Ievadiet sākuma krāsu (z, s, o) --> o
12. rindā zilo pērlīšu skaits ir 598612924, sarkano - 598615654, oranžo - 598617020.
```

Testa piemērs(3)

```
Ievadiet rindu skaitu --> 9
Ievadiet sākuma krāsu (z, s, o) --> s
9. rindā zilo pērlīšu skaits ir 4031078, sarkano - 4031079, oranžo - 4031078.
```

4.uzdevums

Programma, kas izveido sarkstu ar kortežiem, kas sastāv no skaitļu saraksta kārtas numuru un pati skaitli, kas lielāks par lietotāja ievādīto x.

Kods:

```
import random as r

n = int(input('n --> '))
x = int(input('x --> '))

skaitli = []
for num in range(n):
  skaitlis = r.randint(-100, 100)
  skaitli.append(skaitlis)

kortezi = []
for i in range(len(skaitli)):
  if skaitli[i] > x:
    kortezi.append((i, skaitli[i]))
```

print(kortezi)

Testa piemēri(1)

```
n --> 12
x --> 3
[(0, 47), (1, 71), (3, 61), (4, 93), (6, 4), (7, 59), (10, 60), (11, 86)]
```

Testa piemēri(2)

```
n --> 48
x --> 99
[]
```

Testa piemēri(3)

```
n --> 3
x --> -9
[(0, 5), (1, 76)]
```

5.uzdevums

Programma, kas ar grafisko saskarni veic darbības ar divam daļam un saīsina rezultātu.

Grafiska saskarne:

Kods:

```
import tkinter as tk
from tkinter import ttk

def izmaina():
    darbibas = ['+', '-', '*', '/']
    darbibas_vertiba = poga1.cget('text')
    # print(darbibas_vertiba)
    kartas_numurs = darbibas.index(darbibas_vertiba)
    # print(kartas_numurs)

    darbibu_skaits = len(darbibas)
    if kartas_numurs == darbibu_skaits - 1:
        kartas_numurs = 0
    else:
        kartas_numurs += 1

    poga1.config(text=f'{darbibas[kartas_numurs]}')
```

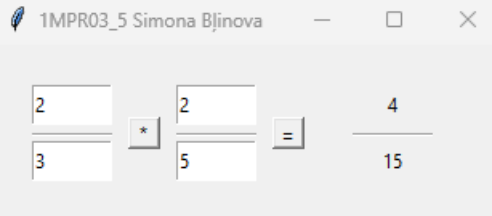
```
def komandas(e):
  parbaude(e)
  poga()

def parbaude(e):
  objekts = e.widget
  skaitlis = objekts.get()
  if naturals(skaitlis) == 'f':
    kr = 'red'
    poga2.config(state=tk.DISABLED)
  else:
    kr = 'white'
  krasa(objekts, kr)

def krasa(a, b):
  a.config(background=b)

def naturals(a):
  try:
    if a == '':
      raise Exception
    a = int(a)
    if a < 0:
      raise Exception
    else:
      return 't'
  except:
    return 'f'

def poga():
  if naturals(skaititajs1.get()) == 't' and naturals(saucejs1.get()) == 't' and naturals(skaititajs2.get())
== 't' and naturals(saucejs2.get()) == 't':
    poga2.config(state=tk.NORMAL)
  else:
    poga2.config(state=tk.DISABLED)

def LKD(starp_skaititajs, starp_saucejs):
  if starp_skaititajs > starp_saucejs:
    lielakais_skaitlis = starp_skaititajs
    mazakais_skaitlis = starp_saucejs
  else:
    lielakais_skaitlis = starp_saucejs
    mazakais_skaitlis = starp_skaititajs
```

```
    while mazakais_skaitlis != 0:
        atlikums = lielakais_skaitlis % mazakais_skaitlis
        lielakais_skaitlis = mazakais_skaitlis
        mazakais_skaitlis = atlikums

    return lielakais_skaitlis

def rezultats():
        dala1 = []
        dala2 = []

        dala1.append(int(skaititajs1.get()))
        dala1.append(int(saucejs1.get()))
        dala2.append(int(skaititajs2.get()))
        dala2.append(int(saucejs2.get()))

        darbibas_zime = poga1.cget('text')

        match darbibas_zime:
            case '+':
                if dala1[1] == dala2[1]:
                    starp_skaititajs = dala1[0] + dala2[0]
                    starp_saucejs = dala1[1]
                else:
                    starp_saucejs = dala1[1] * dala2[1]
                    starp_skaititajs1 = dala1[0] * dala2[1]
                    starp_skaititajs2 = dala2[0] * dala1[1]
                    starp_skaititajs = starp_skaititajs1 + starp_skaititajs2
            case '-':
                if dala1[1] == dala2[1]:
                    starp_skaititajs = dala1[0] - dala2[0]
                    starp_saucejs = dala1[1]
                else:
                    starp_saucejs = dala1[1] * dala2[1]
                    starp_skaititajs1 = dala1[0] * dala2[1]
                    starp_skaititajs2 = dala2[0] * dala1[1]
                    starp_skaititajs = starp_skaititajs1 - starp_skaititajs2
            case '*':
                starp_skaititajs = dala1[0] * dala2[0]
                starp_saucejs = dala1[1] * dala2[1]
            case '/':
                starp_skaititajs = dala1[0] * dala2[1]
                starp_saucejs = dala1[1] * dala2[0]
```

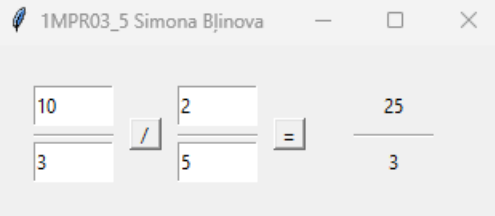
```
    if starp_skaititajs < 0:
        zime.config(text='-')
    else:
        zime.config(text=' ')

    starp_skaititajs = abs(starp_skaititajs)
    starp_saucejs = abs(starp_saucejs)

    dalitajs = LKD(starp_skaititajs, starp_saucejs)

    starp_skaititajs = starp_skaititajs / dalitajs
    starp_saucejs = starp_saucejs / dalitajs

    skaititajs3.config(text=f'{int(starp_skaititajs)}')
    saucejs3.config(text=f'{int(starp_saucejs)}')

logs = tk.Tk()
logs.geometry('300x110')
logs.title('1MPR03_5 Simona Bļinova')

skaititajs1 = tk.Entry(logs)
skaititajs1.place(x=25, y=25, height=25, width=50)
skaititajs1.bind('<KeyRelease>', komandas)

sadalisanas_linija1 = ttk.Separator(logs, orient='horizontal')
sadalisanas_linija1.place(x=25, y=55, height=2, width=50)

saucejs1 = tk.Entry(logs)
saucejs1.place(x=25, y=60, height=25, width=50)
saucejs1.bind('<KeyRelease>', komandas)

poga1 = tk.Button(logs, text='+', command=izmaina)
poga1.place(x=85, y=45, height=20, width=20)

skaititajs2 = tk.Entry(logs)
skaititajs2.place(x=115, y=25, height=25, width=50)
skaititajs2.bind('<KeyRelease>', komandas)

sadalisanas_linija2 = ttk.Separator(logs, orient='horizontal')
sadalisanas_linija2.place(x=115, y=55, height=2, width=50)

saucejs2 = tk.Entry(logs)
saucejs2.place(x=115, y=60, height=25, width=50)
saucejs2.bind('<KeyRelease>', komandas)
```

```
poga2 = tk.Button(logs, text='=', state=tk.DISABLED, command=rezultats)
poga2.place(x=175, y=45, height=20, width=20)

zime = tk.Label(logs, text=' ')
zime.place(x=205, y=50, height=10, width=10)

skaititajs3 = tk.Label(logs, text='...')
skaititajs3.place(x=225, y=25, height=25, width=50)

sadalisanas_linija3 = ttk.Separator(logs, orient='horizontal')
sadalisanas_linija3.place(x=225, y=55, height=2, width=50)

saucejs3 = tk.Label(logs, text='...')
saucejs3.place(x=225, y=60, height=25, width=50)

logs.mainloop()
```

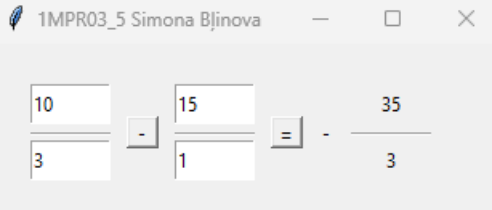
Testa piemērs(1)



Testa piemērs(2)



Testa piemērs(3)



PU2

5.uzdevuma programmas papildinājums, kurā no rezultātā iegūtas daļas tiek izteikta vesela daļa.

Grafiska saskarne:

Kods:

```
import tkinter as tk
from tkinter import ttk

def izmaina():
    darbibas = ['+', '-', '*', '/']
    darbibas_vertiba = poga1.cget('text')
    # print(darbibas_vertiba)
    kartas_numurs = darbibas.index(darbibas_vertiba)
    # print(kartas_numurs)

    darbibu_skaits = len(darbibas)
    if kartas_numurs == darbibu_skaits - 1:
        kartas_numurs = 0
    else:
        kartas_numurs += 1

    poga1.config(text=f'{darbibas[kartas_numurs]}')

def komandas(e):
    parbaude(e)
    poga()

def parbaude(e):
    objekts = e.widget
    skaitlis = objekts.get()
    if naturals(skaitlis) == 'f':
        kr = 'red'
        poga2.config(state=tk.DISABLED)
    else:
        kr = 'white'
    krasa(objekts, kr)

def krasa(a, b):
    a.config(background=b)
```

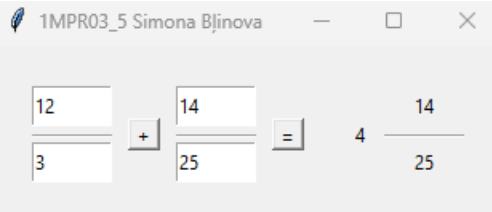
```
def naturals(a):
  try:
    if a == '':
      raise Exception
    a = int(a)
    if a < 0:
      raise Exception
    else:
      return 't'
  except:
    return 'f'

def poga():
  if naturals(skaititajs1.get()) == 't' and naturals(saucejs1.get()) == 't' and naturals(skaititajs2.get())
== 't' and naturals(saucejs2.get()) == 't':
    poga2.config(state=tk.NORMAL)
  else:
    poga2.config(state=tk.DISABLED)

def LKD(starp_skaititajs, starp_saucejs):
  if starp_skaititajs > starp_saucejs:
    lielakais_skaitlis = starp_skaititajs
    mazakais_skaitlis = starp_saucejs
  else:
    lielakais_skaitlis = starp_saucejs
    mazakais_skaitlis = starp_skaititajs

  while mazakais_skaitlis != 0:
    atlikums = lielakais_skaitlis % mazakais_skaitlis
    lielakais_skaitlis = mazakais_skaitlis
    mazakais_skaitlis = atlikums

  return lielakais_skaitlis

def rezultats():
    skaititajs1.configure(background='white')
    saucejs1.configure(background='white')
    skaititajs2.configure(background='white')
    saucejs2.configure(background='white')

    dala1 = []
    dala2 = []

    dala1.append(int(skaititajs1.get()))
```

```
dala1.append(int(saucejs1.get()))
dala2.append(int(skaititajs2.get()))
dala2.append(int(saucejs2.get()))

darbibas_zime = poga1.cget('text')

match darbibas_zime:
  case '+':
    if dala1[1] == dala2[1]:
      starp_skaititajs = dala1[0] + dala2[0]
      starp_saucejs = dala1[1]
    else:
      starp_saucejs = dala1[1] * dala2[1]
      starp_skaititajs1 = dala1[0] * dala2[1]
      starp_skaititajs2 = dala2[0] * dala1[1]
      starp_skaititajs = starp_skaititajs1 + starp_skaititajs2
  case '-':
    if dala1[1] == dala2[1]:
      starp_skaititajs = dala1[0] - dala2[0]
      starp_saucejs = dala1[1]
    else:
      starp_saucejs = dala1[1] * dala2[1]
      starp_skaititajs1 = dala1[0] * dala2[1]
      starp_skaititajs2 = dala2[0] * dala1[1]
      starp_skaititajs = starp_skaititajs1 - starp_skaititajs2
  case '*':
    starp_skaititajs = dala1[0] * dala2[0]
    starp_saucejs = dala1[1] * dala2[1]
  case '/':
    starp_skaititajs = dala1[0] * dala2[1]
    starp_saucejs = dala1[1] * dala2[0]

if starp_skaititajs < 0:
  zime.config(text='-')
else:
  zime.config(text=' ')

starp_skaititajs = abs(starp_skaititajs)
starp_saucejs = abs(starp_saucejs)

dalitajs = LKD(starp_skaititajs, starp_saucejs)

starp_skaititajs = starp_skaititajs / dalitajs
starp_saucejs = starp_saucejs / dalitajs
```

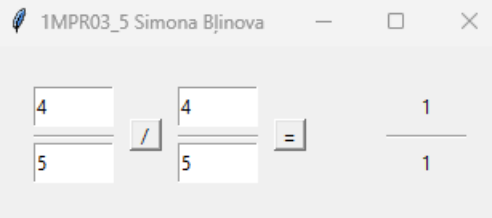
```
    if starp_saucejs < starp_skaititajs:
        vesela_dala = int(starp_skaititajs // starp_saucejs)
        starp_skaititajs -= vesela_dala * starp_saucejs
    else:
        vesela_dala = ' '

    if starp_skaititajs == 0:
        starp_skaititajs = 1

    veseli.config(text=f'{vesela_dala}')
    skaititajs3.config(text=f'{int(starp_skaititajs)}')
    saucejs3.config(text=f'{int(starp_saucejs)}')

logs = tk.Tk()
logs.geometry('320x110')
logs.title('1MPR03_5 Simona Bļinova')

skaititajs1 = tk.Entry(logs)
skaititajs1.place(x=25, y=25, height=25, width=50)
skaititajs1.bind('<KeyRelease>', komandas)

sadalisanas_linija1 = ttk.Separator(logs, orient='horizontal')
sadalisanas_linija1.place(x=25, y=55, height=2, width=50)

saucejs1 = tk.Entry(logs)
saucejs1.place(x=25, y=60, height=25, width=50)
saucejs1.bind('<KeyRelease>', komandas)

poga1 = tk.Button(logs, text='+', command=izmaina)
poga1.place(x=85, y=45, height=20, width=20)

skaititajs2 = tk.Entry(logs)
skaititajs2.place(x=115, y=25, height=25, width=50)
skaititajs2.bind('<KeyRelease>', komandas)

sadalisanas_linija2 = ttk.Separator(logs, orient='horizontal')
sadalisanas_linija2.place(x=115, y=55, height=2, width=50)

saucejs2 = tk.Entry(logs)
saucejs2.place(x=115, y=60, height=25, width=50)
saucejs2.bind('<KeyRelease>', komandas)

poga2 = tk.Button(logs, text='=', state=tk.DISABLED, command=rezultats)
```

```
poga2.place(x=175, y=45, height=20, width=20)

zime = tk.Label(logs, text=' ')
zime.place(x=205, y=50, height=10, width=10)

veseli = tk.Label(logs, text=' ')
veseli.place(x=225, y=50, height=10, width=10)

skaititajs3 = tk.Label(logs, text='...')
skaititajs3.place(x=245, y=25, height=25, width=50)

sadalisanas_linija3 = ttk.Separator(logs, orient='horizontal')
sadalisanas_linija3.place(x=245, y=55, height=2, width=50)

saucejs3 = tk.Label(logs, text='...')
saucejs3.place(x=245, y=60, height=25, width=50)

logs.mainloop()
```

Testa piemērs(1)



Testa piemērs(2)



Testa piemērs(3)