**1MPR03\_Simona\_Bļinova sb24037**

**1.uzdevums un 2.uzdevums**

Programma kas zīmē Serpinska paklāju un Koha zvaigzni.

**Grafiska saskarne:**

A screenshot of a computer

AI-generated content may be incorrect.

**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)**

A screenshot of a computer screen

AI-generated content may be incorrect.

**Testa piemērs(2)**

A screenshot of a computer

AI-generated content may be incorrect.

**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)**

A number and letters on a white background

AI-generated content may be incorrect.

**Testa piemērs(2)**

A black text on a white background

AI-generated content may be incorrect.

**Testa piemērs(3)**

A black numbers and letters on a white background

AI-generated content may be incorrect.

**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)**

A group of black text

AI-generated content may be incorrect.

**Testa piemēri(2)**

A number and arrows with black lines

AI-generated content may be incorrect.

**Testa piemēri(3)**

A number and a number

AI-generated content may be incorrect.

**5.uzdevums**

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

**Grafiska saskarne:**

A screenshot of a computer

AI-generated content may be incorrect.

**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)**

A screenshot of a computer

AI-generated content may be incorrect.

**Testa piemērs(2)**

A screenshot of a computer

AI-generated content may be incorrect.

**Testa piemērs(3)**

A screenshot of a computer

AI-generated content may be incorrect.

**PU2**

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

**Grafiska saskarne:**

A screenshot of a computer

AI-generated content may be incorrect.

**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)**

A screenshot of a computer

AI-generated content may be incorrect.

**Testa piemērs(2)**

A screenshot of a computer

AI-generated content may be incorrect.

**Testa piemērs(3)**

A screenshot of a computer

AI-generated content may be incorrect.