**1MPR02\_Simona\_Bļinova sb24037**

Visiem uzdevumiem lietota viena grafiskā saskarne.

A screenshot of a computer

AI-generated content may be incorrect.

**Kods grafiskai saskarnei un funkcija zīmējumu notīrīšanai:**

import tkinter as tk

import math

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='1MPR02\_1', command=mpr1)

b1.place(x=25, y=50, height=25, width=150)

b2 = tk.Button(logs, text='1MPR02\_2', command=mpr2)

b2.place(x=25, y=85, height=25, width=150)

b3 = tk.Button(logs, text='1MPR02\_3', command=mpr3)

b3.place(x=25, y=120, height=25, width=150)

b4 = tk.Button(logs, text='1MPR02\_4', command=mpr4)

b4.place(x=25, y=155, height=25, width=150)

b5 = tk.Button(logs, text='1MPR02\_5', command=mpr5)

b5.place(x=25, y=190, height=25, width=150)

b6 = tk.Button(logs, text='PU1', command=pu1)

b6.place(x=25, y=225, height=25, width=150)

bnotirit = tk.Button(logs, text='Notīrīt', command=notirit)

bnotirit.place(x=25, y=260, height=25, width=150)

logs.mainloop()

**1.uzdevums**

Programma, kas zīme riņķu zīmējumu ar rekursiju.

**Kods:**

def mpr1():

x = 200

y = 300

r = 175

rinki1(x, y, r)

def rinkis(x, y, r):

kanva.create\_oval(x-r, y-r, x+r, y+r)

def rinki1(x, y, r):

rinkis(x, y, r)

if r > 3:

rinki1(x+r, y, r\*0.55)

Testa piemērs(1)

A group of circles in a row

AI-generated content may be incorrect.

**2.uzdevums**

Programma, kas zīme otro riņķu zīmējumu ar rekursiju.

**Kods:**

def mpr2():

x = 300

y = 300

r = 150

rinki2(x, y, r)

def rinkis(x, y, r):

kanva.create\_oval(x-r, y-r, x+r, y+r)

def rinki2(x, y, r):

rinkis(x, y, r)

if r > 5:

rinki2(x-r, y, r//2)

rinki2(x+r, y, r//2)

rinki2(x, y-r, r//2)

rinki2(x, y+r, r//2)

Testa piemērs(1)

A black and white diamond shape

AI-generated content may be incorrect.

**3.uzdevums**

Programma, kas zīme trešo riņķu zīmējumu ar rekursiju.

**Kods:**

def mpr3():

x = 300

y = 300

r = 75

rinki3(x, y, r)

def rinkis(x, y, r):

kanva.create\_oval(x-r, y-r, x+r, y+r)

def rinki3(x, y, r):

rinkis(x, y, r)

if r > 5:

mazais\_r = r // 3

for i in range(6):

lenkis = (math.pi / 3) \* i

x2 = x + (2\*r+mazais\_r) \* math.cos(lenkis)

y2 = y + (2\*r+mazais\_r) \* math.sin(lenkis)

rinki3(x2, y2, mazais\_r)

Testa piemērs(1)

A black and white drawing of a flower

AI-generated content may be incorrect.

**4.uzedvums**

Programma, kas zīme Pitagora koku.

**Kods:**

def mpr4():

x = 300

y = 600

garums = 200

lenkis = math.pi / 2

linijas(x, y, garums, lenkis)

def linijas(x, y, garums, lenkis):

x2, y2 = linija(x, y, garums, lenkis)

if garums > 2:

linijas(x2, y2, garums\*0.6, lenkis + math.pi / 4)

linijas(x2, y2, garums\*0.6, lenkis - math.pi / 4)

def linija(x, y, garums, lenkis):

x\_beigu = x-garums\*math.cos(lenkis)

y\_beigu = y-garums\*math.sin(lenkis)

kanva.create\_line(x, y, x\_beigu, y\_beigu)

return x\_beigu, y\_beigu

Testa piemērs(1)

A black and white image of a tree

AI-generated content may be incorrect.

**5.uzdevums**

Programma, kas zīme Serpinska trijstūri.

**Kods:**

def mpr5():

x = 300

y = 600

garums = 300

lenkis = math.pi / 3

trijsturi(x, y, garums, lenkis)

def linija(x, y, garums, lenkis):

x\_beigu = x-garums\*math.cos(lenkis)

y\_beigu = y-garums\*math.sin(lenkis)

kanva.create\_line(x, y, x\_beigu, y\_beigu)

return x\_beigu, y\_beigu

def trijsturi(x, y, garums, lenkis):

x1, y1 = linija(x, y, garums, lenkis)

x2, y2 = linija(x, y, garums, lenkis\*2)

kanva.create\_line(x1, y1, x2, y2)

if garums > 10:

x\_jaunais = (x1 + x2) // 2

y\_jaunais = (y1 + y2) // 2

jaunais\_garums = garums // 2

trijsturi(x-jaunais\_garums, y, jaunais\_garums, lenkis)

trijsturi(x+jaunais\_garums, y, jaunais\_garums, lenkis)

trijsturi(x\_jaunais, y\_jaunais, jaunais\_garums, lenkis)

Testa piemērs(1)

A black and white triangle pattern

AI-generated content may be incorrect.

**PU1**

Programma, kas zīme zīmējumu ar rekursiju.

**Kods:**

def pu1():

x=300

y=600

garums = 300

lenkis = math.pi / 2

linijas2(x, y, garums, lenkis)

def linija(x, y, garums, lenkis):

x\_beigu = x-garums\*math.cos(lenkis)

y\_beigu = y-garums\*math.sin(lenkis)

kanva.create\_line(x, y, x\_beigu, y\_beigu)

return x\_beigu, y\_beigu

def linijas2(x, y, garums, lenkis):

x2, y2 = linija(x, y, garums, lenkis)

if garums > 25:

linijas2(x2, y2, garums//2, lenkis - math.pi / 4)

linijas2(x2, y2, garums//2, lenkis + math.pi / 4)

linijas2(x2, y2, garums//2, lenkis - math.pi / 4 \* 3)

linijas2(x2, y2, garums//2, lenkis + math.pi / 4 \* 3)

Testa piemērs(1)

A black and white pattern

AI-generated content may be incorrect.