Објекат правоугаоник

Објекат правоугаоник се одређује као две тачке: (x0, y0) је горње лево теме објекта а (x1, y1) је локација пиксела непосредно изван доњег десног темена објекта.

Правоугаоници се цртају из два дела: ивица је део правоугаоника дебљине један пиксел и унутрашњост правоугаоника ограничена ивицом; њен приказ по дифолту је провидан.

Синтакса објекта линија виџета платна: id_pravougaonika = platno.create_rectangle(x0, y0, x1, y1, opcije, …).

Неке од најчешће коришћених опција су:

назив опције	**опис опције**
dash	Даје испрекидану ивицу правоугаоника
fill	По дифолту, унутрашњост правоугаоника је празна, што је идентично са fill = ''. Овде се дефинише којом бојом се попуњава унутрашњост правоугаоника.
outline	Боја ивице правоугаоника (дифолт је "black").
state	По дифолту, правоугаоници су креирани у NORMAL state. Стање је ACTIVE када је миш изнад правоугаоника. Опција се може поставити у DISABLED чиме ивица посиви и више се не одазива на акције мишем.
width	Дебљина ивице. По дифолту ивица је дебљине једног пиксела. За width = 0 ивица је невидљива.

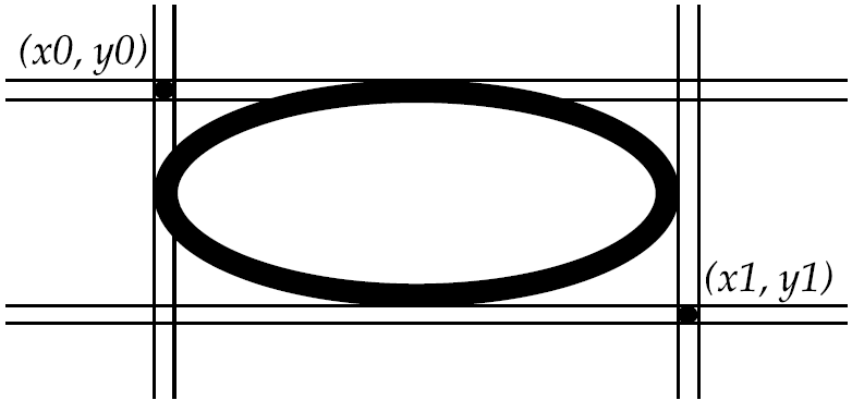
Објекат овал

Математички, овали су елипсе и код њих се кругови сматрају посебним случајевима.

Синтакса објекта правоугаоника виџета платна: id_ovala = platno.create_oval(x0, y0, x1, y1, opcije, …).

Овакав конструктор враћа објекат ID правоугаоника на платно.

Елипса се може поставити у правоугаоник дефинисан координатама:

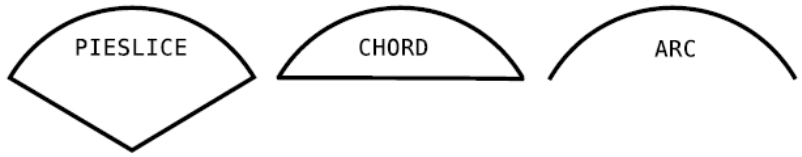


Објекат лук

Објекат лук је попут одсечка једне елипсе и укључује целу елипсу и кругове као посебан случај.

Синтакса објекта лук виџета платна: id_luk = platno.create_arc(x0, y0, x1, y1, opcije, …).

Опција style по дифолту црта цео лук (style = tk.PIESLICE); да би се цртао само циркуларни лук користи се style = tk.ARC; да би се цртао и лук и површина испод лука: style = tk.CHORD.



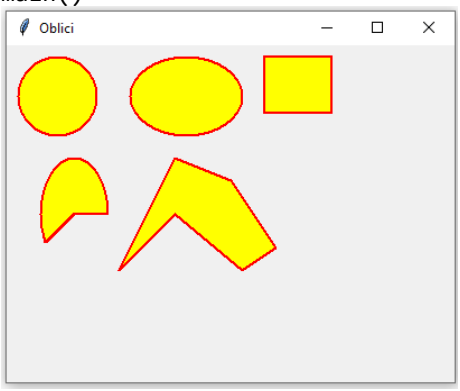
Израда лабораторијских вежби: време реализације 35 минута
Задатак 097: Обојити три правоугаоника различитим бојама.

```
from tkinter import Tk, Canvas, Frame, BOTH

class Primer(Frame):
    def __init__(self):
        super().__init__()
        self.ekran()

    def ekran(self):
        self.master.title("Boje")
        self.pack(fill = BOTH, expand = 1)
        platno = Canvas(self)
        platno.create_rectangle(30, 10, 120, 80, outline = "yellow", fill = "yellow")
        platno.create_rectangle(150, 10, 240, 80, outline = "red", fill = "red")
        platno.create_rectangle(270, 10, 370, 80, outline = "blue", fill = "blue")
        platno.pack(fill = BOTH, expand = 1)

def main():
    prozor = Tk()
    primer = Primer()
    prozor.geometry("400x100")
    prozor.mainloop()

main()
```

Задатак 098: Нацртати селекцију објеката на платну.

```
from tkinter import Tk, Canvas, Frame, BOTH

class Primer(Frame):
    def __init__(self):
        super().__init__()
        self.ekran()

    def ekran(self):
        self.master.title("Oblici")
        self.pack(fill = BOTH, expand = 1)
        platno = Canvas(self)
        b1 = "red"
        b2 = "yellow"
        platno.create_oval(10, 10, 80, 80, outline = b1, fill = b2, width = 2)
        platno.create_oval(110, 10, 210, 80, outline = b1, fill = b2, width = 2)
        platno.create_rectangle(230, 10, 290, 60, outline = b1, fill = b2, width = 2)
        platno.create_arc(30, 200, 90, 100, start = 0, extent = 210, outline = b1, fill = b2,
width = 2)

        tacke = [150, 100, 200, 120, 240, 180, 210, 200, 150, 150, 100, 200]
        platno.create_polygon(tacke, outline = b1, fill = b2, width = 2)
```

```
        platno.pack(fill = BOTH, expand = 1)

def main():
    prozor = Tk()
    primer = Primer()
    prozor.geometry("400x300")
    prozor.mainloop()

main()
```



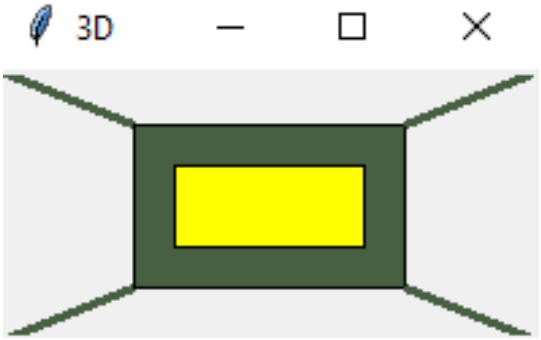
Задатак 099: Креирати једноставну апликацију за цртање.

```
from tkinter import *
from tkinter import ttk

class SlobodnoCrtanje(Canvas):
    def __init__(self, roditelj, **kwargs):
        super().__init__(roditelj, **kwargs)
        self.bind("<Button-1>", self.pamcenje_pozicije)
        self.bind("<B1-Motion>", self.dodavanje_linije)

    def pamcenje_pozicije(self, dogadjaj1):
        self.zadnji_x, self.zadnji_y = dogadjaj1.x, dogadjaj1.y

    def dodavanje_linije(self, dogadjaj2):
        self.create_line((self.zadnji_x, self.zadnji_y, dogadjaj2.x, dogadjaj2.y))
        self.pamcenje_pozicije(dogadjaj2)

ekran = Tk()
ekran.columnconfigure(0, weight = 1)
ekran.rowconfigure(0, weight = 1)

crtez = SlobodnoCrtanje(ekran)
crtez.grid(column = 0, row = 0, sticky = (N, W, E, S))

ekran.mainloop()
```

**kwarg (keyword arguments) је специјалан симбол који се користи за придруживање листе аргумената функцији.
Сви аргументи могу бити додељени различитим промењивима и није ограничен број аргумената у тој листи.
Двострука звезда омогућава да се користе промењиве са вредностима, као аргументи.
Користе се глобалне промењиве за памћење позиције старта цртања.
Креира се подкласа SlobodnoCrtanje од Canvas, која се третира као засебан виџет.
Када се притисне по први пут лево дугме миша, памти се та позиција као стартне координате следеће линије.
Када се миш помера док је притиснуто лево дугме миша, креира се линија од стартне позиције до тренутне позиције миша.
Тренутна позиција миша постаје стартна за следећу линију.

Задаци за самосталан рад: време реализације 70 минута

82. Написати скрипт којим се црта следећа слика:



83. Написати скрипт којим се црта следећа слика: