

ГРАФІЧНЕ ВІДОБРАЖЕННЯ ДАНИХ ЗАСОБАМИ МОВИ ПРОГРАМУВАННЯ.



Відображення рисунків із
зовнішніх файлів.

Створення полотна для малювання у мові програмування Python :

Малювання у Python — це, мабуть, найцікавіша частина у всьому курсі програмування. **Полотно для малювання** — *частина вікна (або все вікно), у якій може бути здійснене малювання об'єктів*. Для створення полотна існує функція **Canvas()** і застосовується вона таким чином:

назва_полотна=Canvas(назва_вікна, атрибут1...)

Атрибути (властивості) полотна

- **width=число_у_пікселях** — ширина полотна;
- **height=число_у_пікселях** — висота полотна;
- **bg="колір"** — колір фону.

Увага! Потрібно обов'язково розмістити полотно у вікні за допомогою методу **place(x=число (відступ зліва), y=число (відступ зверху))** або додати метод **pack()** розміщення на все вікно поля для малювання **canvas.pack()**

Зображення основних графічних об'єктів у Python

У програмах часто використовують малюнки: ілюстрації, рухомі зображення, фони тощо. Деякі середовища програмування мають засоби, які забезпечують додавання готових зображень до програмного коду або створення і форматування малюнків у самій програмі.

Наприклад, у навчальному середовищі створення та виконання алгоритмів Скретч ви використовували такі засоби для роботи з графікою:

- змінювали образи об'єктів, завантажуючи їх із бібліотеки;
- малювали об'єкти у вбудованому графічному редакторі;
- виконували побудову зображень виконавцем, вказуючи у програмному коді команди групи Олівець

Полотно для побудови графічних об'єктів модуля tkinter

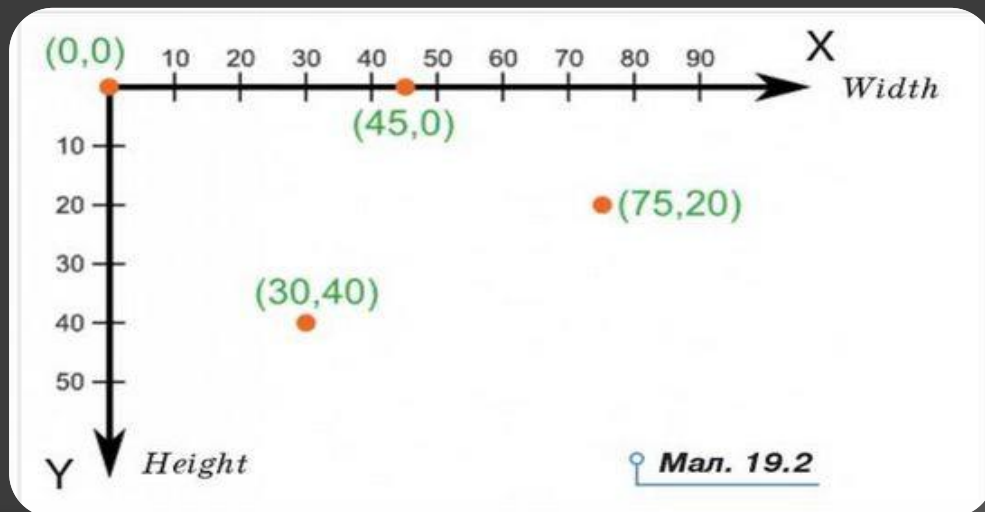
Для побудови графічних об'єктів, що складаються з геометричних примітивів і готових малюнків, викликають відповідний метод модуля **tkinter**, який імпортують до проекту

Створення
полотна

```
tkinter.Canvas(<назва форми>, width=<значення>,  
height=<значення>)
```

Полотно для побудови графічних об'єктів модуля tkinter

Розмір полотна визначається значеннями властивостей Height -кількість точок за вертикаллю, та Width — за горизонталлю. Полотно складається з окремих точок — пікселів, координати яких задаються значеннями x та y



Полотно для побудови графічних об'єктів модуля tkinter

Які геометричні примітиви можна створювати за допомогою модуля Canvas? Створення зображень на полотні викликається методом:

`<Ім'я_об'єкту_полотно>.create_<об'єкт>.`

Можна побудувати такі об'єкти:

- лінія — `line(x1, y1, x2, y2)`
- прямокутник — `rectangle(x1, y1, x2, y2)`
- багатокутник — `polygon(x1, y1, x2, y2, x3, y3, x4, y4)`
- ламана — `polyline(x1, y1, x2, y2, x3, y3)`
- еліпс — `oval(x1, y1, x2, y2)`
- дуга — `arc(x1, y1, x2, y2, x3, y3, x4, y4).`
- `arc(x1, y1, x2, y2, x3, y3, x4, y4).`

Об'єкти можуть мати параметри: `fill` — колір заливки, `dash` — тип заливки, `width` — ширина лінії та інші.

Приклади застосування:

Вправа 1. Стрілка.

Завдання. Створіть малюнок стрілки за зразком

1. Відкрийте середовище програмування. Створіть новий файл програми мовою Python з іменем Стрілка_Прізвище в папці Навчальні проекти своєї структури папок.

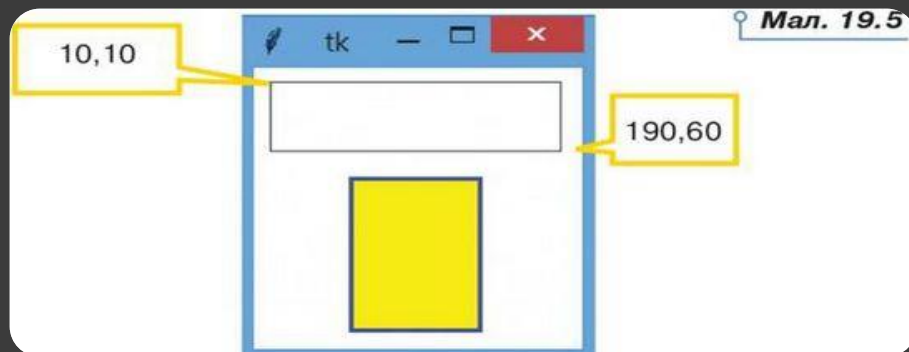


```
1 from tkinter import *
2 window = Tk()
3 c = Canvas(window, width=100, height=100, bg='white')
4 c.pack()
5 c.create_line( 10,10,  90,10)      #малювання горизонтальної лінії
6 c.create_line((50,20),(50,90),    #малювання вертикальної лінії
7               fill='green',        #колір лінії
8               width=3,              #товщина лінії
9               arrow=FIRST,          #напрямок стрілки: FIRST - вгору, LAST - донизу
10              dash=(10,2),          #тип штриховки
11              activefill='red',     #колір при наведенні
12              arrowshape="10 25 10")#вигляд стрілки
13 window.mainloop()
14
```

Приклади застосування:

Вправа 2. Прямокутники.

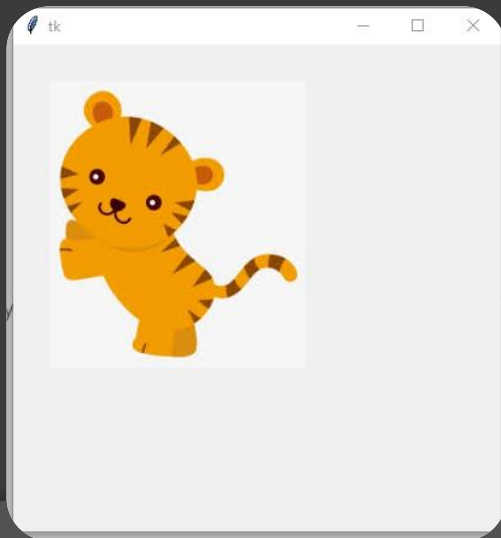
Завдання. Створіть малюнок двох прямокутників за зразком



```
1 from tkinter import *
2 window = Tk()
3 c = Canvas(window, width=200, height=200, bg="white")
4 c.pack()
5 c.create_rectangle(60,80,140,190, fill= "yellow", outline="blue", width=3)
6 c.create_rectangle(10, 10, 190, 60)
7 window.mainloop()
```


Відображення рисунків із зовнішніх файлів.

Модуль tkinter у Python надає можливість створювати графічні інтерфейси користувацьких програм. Один з важливих аспектів створення графічних інтерфейсів - це можливість відображати зображення у програмі.



Відображення рисунків із зовнішніх файлів.

Продовження

Для відображення рисунків у модулі tkinter можна використовувати зображення у форматі .gif, .png, .jpg та інших підтримуваних форматах. Основний підхід полягає в тому, щоб завантажити зображення з файлу, створити його примірник у форматі, зрозумілому для tkinter, і відобразити його на вікні за допомогою відповідного віджета.

```
from tkinter import *
tk = Tk()
canvas = Canvas(tk, width=400, height=400)
canvas.pack()
my_image = PhotoImage(file='c:\\test.gif')
canvas.create_image(0, 0, anchor=NW, image=my_image)
```

Відображення рисунків із зовнішніх файлів.

Щоб відображати рисунки з зовнішніх файлів у модулі tkinter, вам знадобиться використати бібліотеку PIL (Python Imaging Library). Ось приклад коду, який показує, як відобразити зображення з файлу у вікні tkinter:

```
1 from PIL import ImageTk, Image
2 import tkinter as tk
3 # Створення вікна
4 window = tk.Tk()
5 # Завантаження зображення з файлу
6 image = Image.open("шлях_до_файлу_зображення.jpg")
7 # Створення примірника ImageTk для зображення
8 image_tk = ImageTk.PhotoImage(image)
9 # Створення етикетки для відображення зображення
10 label = tk.Label(window, image=image_tk)
11 label.pack()
12 # Запуск головного циклу вікна
13 window.mainloop()
```



Відображення рисунків із зовнішніх файлів.

Для роботи із зображеннями в [Tkinter](#) є два класи **BitmapImage** та **PhotoImage** .

- **BitmapImage** є простим двоколірним зображенням,
- **PhotoImage** — повнокольоровим зображенням.

BitmapImage

Конструктор класу приймає такі аргументи:

background та **foreground** — кольори фону та переднього плану для зображення.

Оскільки зображення двокольорове, ці параметри визначають відповідно чорний і білий колір;

file і **maskfile** - шляхи до файлу із зображенням і до маски (зображення, що вказує які пікселі будуть прозорими);

data та **maskdata** – замість шляху до файлу можна вказати вже завантажені в пам'ять дані зображення. Ця можливість зручна для вбудовування зображення у програму.

Відображення рисунків із зовнішніх файлів.

Photolmage

Photolmage дозволяє використовувати повнокольорове зображення. Крім того, у цього класу є кілька (досить примітивних) методів для роботи із зображеннями. Photolmage гарантовано розуміє формат GIF. Аргументи конструктора:

file - шлях до файлу із зображенням;

data — замість шляху до файлу можна вказати вже завантажені на згадку дані зображення. Зображення у форматі GIF можуть бути закодовані за допомогою base64. Ця можливість зручна для вбудовування зображення у програму;

format - явна вказівка формату зображення;

width , **height** - ширина та висота зображення;

gamma - корекція гами;

palette - зображення палітри.

Приклади BitmapImage та PhotoImage

```
1 import tkinter as tk
2 # Створення вікна
3 window = tk.Tk()
4 # Завантаження зображення у форматі BMP
5 image = tk.BitmapImage(file="шлях_до_файлу_зображення.bmp")
6 # Створення елементу Label для відображення зображення
7 label = tk.Label(window, image=image)
8 label.pack()
9 # Запуск головного циклу вікна
10 window.mainloop()
```

BitmapImage

PhotoImage

```
1 from tkinter import *
2 root = Tk()
3 root.title("Відображення зображення")
4 canvas = Canvas(root, width=300, height=300)
5 canvas.pack()
6 def show_image():
7     image = PhotoImage(file="image.gif")
8     canvas.create_image(0, 0, anchor=NW, image=image)
9 button = Button(root, text="Показати зображення", command=show_image)
10 button.pack()
11 root.mainloop()
```

Методи tkinter для малювання на полотні:

Метод `create_image` є частиною модуля `canvas` в бібліотеці `tkinter` і використовується для створення об'єкта зображення на полотні (`canvas`). Він дозволяє відображати зображення у вікні програми. Синтаксис методу `create_image` виглядає так:

```
python
```

[Copy code](#)

```
canvas.create_image(x, y, anchor=anchor, image=image)
```

Загалом, метод `create_image` дозволяє вставляти зображення на полотно та контролювати його розташування та відображення у вікні програми.

Параметри методу create_image

- ❖ `x` і `y`: вказують координати верхнього лівого кута зображення на полотні.
- ❖ `anchor` (необов'язковий): вказує якорну позицію зображення на полотні. Можливі значення для `anchor` - N, NE, E, SE, S, SW, W, NW або CENTER. За замовчуванням використовується CENTER.
- ❖ `image`: об'єкт `PhotoImage`, який представляє зображення, яке потрібно відобразити.

Домашнє завдання.

Вправа 1. Побудова графічних об'єктів модуля tkinter

Створити вікно програми з розміром **500x500** пікселів. На вікні створити полотно (canvas) розміром **500x500** пікселів з блакитним фоном. На полотні намалювати лінію з координатами **[100,100]** до **[200,200]**. Лінія повинна мати ширину **3** пікселі і зелений колір. Запустити головний цикл вікна програми для його відображення і обробки подій. Таким чином, програма створює вікно з полотном і малює на ньому зелену лінію.

Домашнє завдання.

Вправа 2. Завантаження картинки до модуля tkinter

Створити вікно tkinter з полотном (canvas) розміром 400x400 пікселів і відобразити зображення з файлу на цьому полотні. Координати розміщення зображення на полотні слідуючі: (0, 0, anchor=NW)