

Wstęp do informatyki – WINF

Laboratorium nr 11 (Pythona ciąg dalszy - cz. 2)

G.Nieradka, M. Linczuk

Warszawa, Grudzień 2025

Uwagi do wykonania zadań.

- Programy z poszczególnych zadań należy umieścić w osobnych modułach, tworząc odrębne uruchamialne pliki z rozszerzeniem `.py`.
- Dobrym zwyczajem jest umieszczenie w nagłówku pliku, jako komentarz, informacji o autorze pliku, dacie jego powstania i celowi przyświecającemu jego stworzeniu (np. rozwiązanie zadania nr 2).
- Poszczególne operacje należy wykonywać na plikach graficznych dołączonych do instrukcji. Dopuszcza się wykorzystanie własnych plików graficznych (pod warunkiem przestrzegania ograniczeń wynikających z praw autorskich) oraz zamieszczenie ich jako części składowe rozwiązania.
- Można również poniższe zadania rozwiązywać z wykorzystaniem środowiska Jupyter. W takim przypadku jako rozwiązanie proszę utworzyć i przedstawić do ewaluacji rozwiązanie, cztery niezależne notatniki w formacie `.ipynb`.

$\forall \text{ errors } \exists \text{ "why u no work?!"} \rightarrow \text{Google}() \rightarrow \text{fix}() \Rightarrow \text{levelUp}++$

☕🔥 "Learning code all day 🧘, turning bugs into wins all night 🌙"

Zadanie 1.

Zapoznać się z metodą: `pyplot.subplots(matplotlib.pyplot.subplots)` biblioteki `matplotlib` (`matplotlib`).

Następnie napisać program, który utworzy figurę składającą się z czterech wykresów (na jednej figurze):

- W lewej, górnej części wykresu, wykorzystując metody modułu `pyplot.imread` (`pyplot.imread`) oraz `pyplot.imshow` (`pyplot.imshow`) umieścić i wyświetlić załączony rysunek kota,
- a w prawym, dolnym rysunek korytarza. (lub inne wybrane przez siebie obrazki, ale wtedy wykorzystane obrazki proszę umieścić w sprawozdaniu)
- W prawym, górnym kwadrancie figury proszę narysować krzywą daną równaniem parametrycznym:

$$\begin{aligned}x &= 1.5 \sin(t)^3 \\ y &= 3 \cos(t) - 1.2 \cos(2t) - 0.6 \cos(3t)\end{aligned}$$

dla $0 \leq t < 7$ i dowolnie przez siebie wybranym krokiem.

- W lewym, dolnym kwadrancie wykres funkcji sigmoidalnej wyrażony wzorem:

$$\frac{1}{1 + e^{-k(x-x_0)}}$$

o dowolnie dobranych parametrach k i x_0 , dziedzinie $x \in [-5; 5]$ i dowolnie dobranym kroku.

Zadanie 2.

Napisać program, który wykona wykres tłumionej sinusoidy. Wykres powinien:

- obejmować około pięciu okresów sinusoidy
 - tłumienie dobrać tak, by spadek amplitudy w zakresie rysunku był około dwukrotny
 - na rysunku należy umieścić obie obwiednie
 - każdy z wykresów (sinusoida i obwiednie) powinien być wykonany innym kolorem i innym rodzajem linii
 - rysunek powinien zawierać legendę, opisy osi, tytuł
 - obwiednie należy opisać wyrażeniami matematycznymi
-

Zadanie 3.

1. Zapoznać się z metodą `Image.paste` ([PIL.Image.paste](#)).
Za pomocą metody `paste` umieścić kota z załączonego obrazu `cat.png` w załączonym obrazie korytarza `corridor.png` i wyświetlić wynik tej operacji.
 2. Zapoznać się z metodą `ImageEnhance.Sharpness` ([PIL.ImageEnhance.Sharpness](#)), metodą `Image.effect_noise` ([PIL.Image.effect_noise](#)) i metodą `Image.blend` ([PIL.Image.blend](#)).
Następnie metodą `ImageEnhance.Sharpness` zmienić obraz `corridor.png` ze współczynnikiem równym 2, za pomocą metody `Image.effect_noise` wygenerować silnie zaszumiony obraz kota, wykorzystując obraz: `cat.png` a następnie obraz sumy ważonej ze współczynnikiem równym 0.5, za pomocą metody `Image.blend`.
Zmodyfikowane obrazy pośrednie i obraz końcowy przedstawić na jednej, wieloczęściowej figurze.
 3. Zapoznać się z metodą `Image.filter` ([PIL.Image.filter](#)) oraz modulem `PIL.ImageFilter` ([PIL.ImageFilter](#)).
Następnie dla obrazu `cat.png` zastosować filtr z predefiniowaną maską `DETAIL` a dla obrazu `corridor.png` zastosować filtr `ImageFilter.MedianFilter` z maską o rozmiarze 9x9 pikseli.
Obrazy oryginalne i obrazy po filtracji przedstawić na jednej wieloczęściowej figurze.
-

Zadanie 4.

Poniżej zapisany został algorytm sortowania gnomu (*ang. gnome sort*).

```
1: A tablica elementów do posortowania
2: n liczba elementów w tablicy A
3: function GNOMESORT(A)                                ▷ Sortowanie gnomu (ang. gnome sort)
4:   i := 1
5:   j := 2
6:   while i < n do
7:     if A[i-1] <= A[i] then
8:       i := j
9:       j := j+1
10:    else
11:      zamień miejscami A[i-1] i A[i]
12:      i := i - 1
13:      if i = 0 then
14:        i := 1
15:      end if
16:    end if
17:  end while
18: end function
```

Proszę utworzyć skrypt Pythona implementujący ten algorytm jako funkcję. Ponieważ, występuje

tutaj funkcja dodatkowym elementem skryptu powinna być deklaracja miejsca rozpoczęcia wykonywania skryptu, którą w Pythonie realizuje się poleceniem:

```
1     if __name__ == '__main__':  
2         # Execute when the module is not initialized from an import statement.  
3         ...
```

A po tej linii należy zdefiniować testową tablicę do posortowania, wywołać funkcję z tą tablicą jako jej argumentem i wyświetlić wynik po sortowaniu. Dodatkową zależnością jest, że interpreter Pythona musi znać funkcję przed jej wywołaniem, więc jej zawartość musi się znaleźć "wcześniej" w pliku niż deklaracja funkcji `__main__`.

Dobrze jest się uczyć Pythona 🚀💰

