Programowanie I R

Zadania – seria 12.

Przekazywanie argumentów i parę przydanych funkcji z Numpy

Zadanie 1. `argskwargs` – **Przekazywanie dowolnych argumentów**

W wielu implementacjach klas i metod napotkamy sygnatury `def func(*args, **kwargs)`. W zadaniu przetestuj sposoby pakowania i rozpakowywania argumentów przy pomocy operatorów `*` oraz `**`.

1. Napisz funkcję `method_1(*args, **kwargs)`, która wypisuje otrzymane argumenty. Przetestuj różnorodne kombinacje argumentów i podając same argumenty pozycyjne, same argumenty nazwane klucz-wartość, a następnie mieszając je ze sobą. Przekaż zmienne na standardowe wyjście i zobacz w jakiej postaci funkcja widzi je.

2. Napisz funkcję `method_2(a, b, *args, c=None, **kwargs)`, która posiada złożoną sygnaturę. Przetestuj ją z różnymi kombinacjami argumentów, przekaż na standardowe wyjście i zobacz w jakiej postaci funkcja widzi te zmienne.

3. Napisz funkcję `plot_with_params(x, y, **params)`, która przyjmuje wektory danych (np x, $\sin x$), a także dowolne parametry graficzne przy pomocy konstrukcji `**params`. Przekaż `params` wprost do rozpakowania w poleceniu `plt.plot(x, y, **params)`. Przetestuj jej użycie tworząc wykres ze zdefiniowanym kolorem, stylem linii i etykietą legendy, przekazywanymi w wywołaniu `plot_with_params`.

Opracowanie: Bartosz Kasza.

Zadanie 2. `numpy_various` – **Selekcja wybranych przydatnych funkcji w Numpy.**

Napisz program `numpy_various.py`, który wykorzysta funkcje `np.fromfunction`, `np.where` oraz `np.roll` z biblioteki `Numpy`. Program w argumentach wywołania powinien przyjmować:

- jedną z flag wybierających kształt oraz wymiary. Przykład w argparse z wykorzystaniem wykluczającej się grupy: `group = parser.add_mutually_exclusive_group(required=True) group.add_argument(''-s'', ''-square'', type=float, metavar=''A'', help=''Kwadrat o boku A'')`

- wymiar macierzy `-m, -matrix-size`, szerokość filtra `-n, -filter-size`, tuplę odpowiadającej kodowaniu koloru w formacie RGB `-c, -color`,

Zaimplementuj następujące funkcjonalności:

Wykorzystaj abstrakcyjną klasę bazową `Shape` z abstrakcyjną metodą `draw`. Zaimplementuj klasy dziedziczące: `Square`, `Rectangle` oraz `Ellipse`. Przy inicjalizacji powinny być przyjmowane parametry opisujące wymiary figury. Wykorzystaj funkcję `numpy.fromfunction`, aby wypełnić macierz o powierzonych wymiarach $m \times m$ wartościami 1 w miejscach przestrzeni należących do zadanego kształtu (wycentrowanego w środku macierzy), a wartościami 0 w pozostałych. Metoda ta przyjmuje funkcję, która realizuje warunek logiczny na geometrię danego obiektu.

Utwórz funkcję `color_shape(shape_matrix, color, n)`, która wykorzystując `numpy.where` znajdzie indeksy, gdzie wartości są 1, a następnie nałoży zadaną barwę (w postaci krotek RGB) na wygenerowany wcześniej obszar, a tło pozostawi czarne. Następnie zaimplementuj przestrzenne wygładzanie filtrem w kształcie kwadratu. Uśrednij każdy piksel z sąsiedztwem okna o rozmiarze $n \times n$ (zakładamy nieparzyste n), wykorzystując podejście oparte o przesuwanie elementów macierzy w określonych kierunkach z pomocą metody `numpy.roll`.

Wyświetl obrazek z pomocą `plt.imshow`. Dla macierzy krotek w formacie `np.uint8` funkcja `plt.imshow` automatycznie przetłumaczy wartości RGB na kolory. Sprawdź na krawędziach, czy wygładzanie zadziałało.

Opracowanie: Bartosz Kasza.

Dziękuję za udział w tegorocznych zajęciach!