Laboratorium 15 — Opracowanie programów realizujących obliczenia inżynierskie z wykorzystaniem biblioteki `NumPy` i `SciPy`.

Wczytywanie plików txt — biblioteka `NumPy`

Biblioteka `NumPy` zawiera dwie funkcje wczytujące pliki txt:

- `loadtxt()`, której parametry to m.in. `fname` (nazwa pliku), `dtype` (typ danych w tablicy wynikowej), `comments` (znaki używane do oznaczenia komentarza), `delimiter` (znak użyty do rozdzielenia wartości w wierszach), `usecols` (które kolumny mają zostać wczytane) i inne,
- `genfromtxt()`, która dodatkowo zapewnia obsługę plików z brakującymi danymi; niektóre z parametrów to: `fname` (nazwa pliku), `dtype` (typ danych), `delimiter` (znak użyty do rozdzielenia wartości w wierszach), `skip_header` (liczba wierszy z początku pliku, jaka ma zostać pominięta), `skip_footer` (liczba wierszy z końca pliku, jaka ma zostać pominięta), `missing_values` (oznaczenie brakujących wartości), `filling_values` (wskazanie wartości, która ma zastąpić braki w danych) itd.

Zadanie 1 Plik `irisP.txt` zawiera dane o 3 gatunkach irysów. Każda instancja opisana jest przez: `sepallength`, `sepalwidth`, `petallength`, `petalwidth` oraz jeden z trzech gatunków. Wyznacz średnią szerokość płatka w ramach każdego gatunku irysa.

Zadanie 2 Plik `irisP2.txt` zawiera dane o 3 gatunkach irysów, które są opisane 4 atrybutami i klasą. Niestety nie wszystkie dane w pliku są uzupełnione. Uzupełnij braki dla atrybutu `petallength` (długość płatka), w ramach każdej z klas, średnią z istniejących wartości.

Operacje na tablicach ze stringami

- `char.lower(A)` — zwraca tablicę z elementami, w których wielkie litery zostały zamienione na małe,
- `char.upper(A)` — zwraca tablicę z elementami, w których małe litery zostały zamienione na wielkie,
- `char.lstrip(A, chars=None)` — zwraca kopię tablicy z usuniętymi wskazanymi znakami `chars` znajdującymi się na początku każdego z elementów; gdy `char=None` (lub parametr jest pominięty) to usuwane są białe znaki,
- `char.rstrip(A, chars=None)` — zwraca kopię tablicy z usuniętymi wskazanymi znakami `chars` znajdującymi się na końcu każdego z elementów; gdy `char=None` (lub parametr jest pominięty) to usuwane są białe znaki,
- `char.strip(A, chars=None)` — zwraca kopię tablicy z usuniętymi wskazanymi znakami `chars`; gdy `char=None` (lub parametr jest pominięty) to usuwane są białe znaki,
- `char.split(A, sep=None, maxsplit=None)` — dla każdego elementu tablicy `A` zwraca listę słów powstałą poprzez podzielenie danego elementu względem wskazanego separatora `sep`; dodatkowo można określić maksymalną liczbę podziałów `maxsplit`,
- `char.replace(A, old, new, count=None)` — dla każdego elementu tablicy `A` zwraca jego kopię, w której wskazany ciąg znaków `old` został zastąpiony innym ciągiem znaków `new`. `count` wskazuje, ile pierwszych elementów ma zostać zamienionych.

Zadanie 3 Wczytaj zbiór `contact-lenses.txt` (5 kolumn reprezentujących kolejne atrybuty: age {young, pre-presbyopic, presbyopic}, spectacle-prescrip {myope, hypermetrope}, astigmatism {no, yes}, tear-prod-rate {reduced, normal}, contact-lenses {soft, hard, none}) i wykonaj poniższe polecenia bez stosowania pętli:

- zamień wszystkie małe litery na wielkie,
- dla atrybutu `astigmatism` zamień wszystkie `NO` na `FALSE` oraz `YES` na `TRUE`,
- zapisz wynikową macierz do pliku `contact-lenses-res.txt`.

Zadanie 4 Wczytaj zbiór `irisP.txt` i usuń z nazw kwiatów „Iris-" bez użycia pętli. Zapisz wynikowy zbiór do pliku `iris-res.txt`.

Zadanie 5 Jednym z popularnych algorytmów optymalizacji jest PSO (ang. particle swarm optimization), który można użyć np. w celu odnalezienia minimum funkcji dwóch zmiennych.

1. Wylosuj początkowe położenia n cząsteczek (P). Każdą cząsteczkę definiują dwie wartości: współrzędna x oraz y. Wartości współrzędnych powinny zostać wylosowane z zadanego przedziału $[A_x, A_y]$.
2. Przypisz każdej cząsteczce jej dotychczasową najlepszą pozycję tj. $Pbest = P$.
3. Zapisz dotychczasową najlepszą pozycję globalną $Gbest$ tj. pozycję cząsteczki, dla której $f(Pi)$ jest najmniejsze.
4. Wylosuj początkowe prędkości cząsteczek (V). Każdą cząsteczkę definiują dwie wartości: prędkość w kierunku x oraz y. Wartości prędkości powinny zostać wylosowane z zadanego przedziału $[-2 * A_x, 2 * A_y]$.
5. Powtarzaj przez określoną liczbę iteracji it.
   (a) Oblicz nową prędkość każdej i-tej cząsteczki zgodnie ze wzorem:
   $$V_i = 0.8 * V_i + 2r_p(Pbest_i - P_i) + 2r_g(Gbest - P_i), \tag{1}$$
   gdzie r_p oraz r_g są liczbami losowymi o rozkładzie jednostajnym.
   (b) Oblicz nowe położenie każdej i-tej cząsteczki zgodnie ze wzorem:
   $$P_i = P_i + V_i \tag{2}$$
   oraz zaktualizuj jej najlepsze dotychczasowe położenie $Pbest_i$ jeśli $f(P_i) < f(Pbest_i)$
   (c) Zaktualizuj najlepsze globalne położenie $Gbest$, jeśli nowe położenie którejś z cząstek zapewnia mniejszą wartość f niż $Gbest$.

Napisz funkcję *pso* realizującą powyższy algorytm. Funkcja powinna przyjmować 4 parametry: n — liczbę cząsteczek, it — liczbę iteracji, xy_min — początek przedziału poszukiwań dla x i y (załóż, że są one takie same) oraz xy_max — koniec przedziału poszukiwań dla x i y (załóż, że są one takie same). Funkcja ma zwracać minimum funkcji $f(x, y) = x^2 + 3y^2$, tj. punkt $Gbest$ oraz wartość funkcji dla tego punktu.
Przykładowe wywołanie funkcji: $pso(100, 20, -5, 5)$.