Systemy AI 3 – (Python)

1. Dla poniższych:

$$\mathbf{A} = \begin{bmatrix} 2 & -3 & 1 \\ 4 & 5 & 0 \\ 2 & -1 & 3 \end{bmatrix} \quad \mathbf{B} = [3, -4, -2] \quad \mathbf{C} = \begin{bmatrix} 2 & 4 \\ -2 & 1 \\ 5 & 0 \end{bmatrix} \quad \mathbf{D} = \begin{bmatrix} 3 \\ 6 \\ 8 \end{bmatrix}$$

   wykonaj polecenia:
   - Zdefiniuj **tensory** NumPy **A**, **B**, **C** i **D** odpowiadające powyższym macierzom.
   - Zmień **shape** tensora **B** na (4,1) i zapisz (jako kopię) w tablicy **B1**.
   - Wykonaj wszystkie możliwe działania ***** dla par tensorów **A**, **B**, **B1**, **C** i **D**. Czy operacja * jest mnożeniem macierzy?
   - Wykonaj wszystkie możliwe działania **np.matmul** dla par tensorów **A**, **B**, **B1**, **C** i **D**. Czy operacja **np.matmul(x,y)** jest mnożeniem macierzy?
   - Sprawdź czy operacja **np.dot(x,y)** jest tożsama z operacją **np.dot np.matmul**.
   - W przypadku której macierzy możliwe jest znalezienie **macierzy odwrotnej**? Znajdź tę wartość wykorzystując odpowiednią operację z **numpy.linalg**.
   - Zastosuj operację **np.sum** dla powyższych tablic, a także do ich osi. Jaka jest interpretacja tej operacji? Definicję operacji **np.sum** odszukaj w dokumentacji biblioteki **NumPy**.

2. Zdefiniuj kilka tensorów **rzędu 3** o różnym kształcie zawierających **12** liczb całkowitych. Przetestuj działanie operacji **np.matmul**. Wykorzystaj metodę **np.arange(n)** oraz **reshape**.

3. Za pomocą instrukcji:

```
import pandas as pd
data = pd.read_csv('folder/nazwa_pliku.csv')
print(data)
```

   wczytaj dane z pliku **`simple_dataset.csv`** i następnie zdefiniuj tablice (nie będące widokami, ale kopiami) zawierające zaznaczone fragmenty obiektu **DataFrame**.

Tablica **S1**:

	X	B	C	D	E
0	1	12	6	5	-4
1	2	11	-4	7	-2
2	3	21	8	-2	9
3	4	4	12	1	10

Tablica **S3**:

	X	B	C	D	E
0	1	12	6	5	-4
1	2	11	-4	7	-2
2	3	21	8	-2	9
3	4	4	12	1	10

Tablica **S2**:

	X	B	C	D	E
0	1	12	6	5	-4
1	2	11	-4	7	-2
2	3	21	8	-2	9
3	4	4	12	1	10

Tablica **S4**:

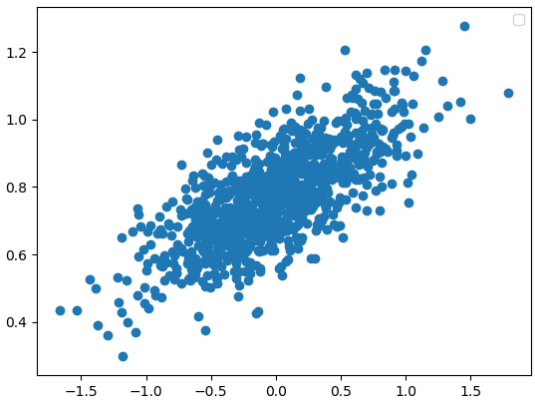
	X	B	C	D	E
0	1	12	6	5	-4
1	2	11	-4	7	-2
2	3	21	8	-2	9
3	4	4	12	1	10

4. Dla danych z pliku **`president_heights.csv`** znajdź:

   - wartość średnią **`.mean()`**
   - odchylenie standardowe **`.std()`**
   - minimum **`.min()`** i maximum **`.max()`**
   - medianę **`np.median()`**
   - narysuj **histogram** dla 10 przedziałów. Wykorzystaj:

```
import matplotlib.pyplot as plt
plt.hist(heights,10,color='red')
plt.title('opis wykresu')
plt.xlabel('opis osi X')
plt.ylabel('opis osi Y')
plt.show()
```

5. Wykonaj polecenia:
   - Wykorzystując **`np.random.normal`** wygeneruj dane testowe (1000 punktów na płaszczyźnie) do których będzie można zastosować metodę najmniejszych kwadratów. Przykład:



   - Wykorzystując:

```
import matplotlib.pyplot as plt
plt.scatter(x,y)
plt.show()
```

     przedstaw na wykresie wygenerowane punkty.

6. Do danych wygenerowanych w **Zadaniu 5** zastosuj metodę najmniejszych kwadratów (omówioną na wykładzie) i znajdź prostą aproksymującą.

   Uzyskaną prostą dodaj do wykresu metodą `plt.plot(x,y)`.

7. Wczytaj dane z pliku **`california_cities.csv:`**
   - Znajdź współczynniki **korelacji Pearsona** dla atrybutów: *population_total, area_total_sq_mi, area_land_sq_mi.* Wykorzystaj: **np.corrcoef**.
   - Przedstaw miasta na wykresie rozrzutu: *population_total - area_total_sq_mi*
   - Narysuj histogramy w oparciu o powyższe dwa atrybuty. Przetestuj różną liczbę przedziałów.