|  |  |  |  |
| --- | --- | --- | --- |
| Obraz zawierający symbol, design  Opis wygenerowany automatycznie | Politechnika Bydgoska im. J. J. Śniadeckich  **Wydział Telekomunikacji,**  **Informatyki i Elektrotechniki** | | Obraz zawierający symbol, logo, Grafika, Czcionka  Opis wygenerowany automatycznie |
| **Przedmiot** | Skryptowe języki programowania | | |
| **Prowadzący** | mgr inż. Martyna Tarczewska | | |
| **Temat** | *Numpy* | | |
| **Student** | Marcin Ogórkiewicz | | |
| **Nr ćw.** | 9 | **Data wykonania** | 28.11.2023 |
| **Ocena** |  | **Data oddania spr.** | 28.11.2023 |

**Zadanie 1.**

#!C:\Users\koksu\AppData\Local\Programs\Python\Python310  
  
# importy  
import typing  
import numpy as np  
# stałe i zmienne globalne  
  
# funkcje  
  
  
def replace\_zeros(A, x):  
 A[A == 0] = x  
 return A  
  
  
def main() -> None:  
 matrix = np.array([[1, 0, 3], [0, 5, 0], [6, 7, 0]])  
 print("Macierz pierwotna:")  
 print(matrix)  
 replacement\_value = 10  
 result\_matrix = replace\_zeros(matrix, replacement\_value)  
 print("\nMacierz po zamianie zer na ", replacement\_value, ":")  
 print(result\_matrix)  
  
  
main()

**Obraz zawierający tekst, zrzut ekranu, Czcionka, design

Opis wygenerowany automatycznie**

**Zadanie 2.**

#!C:\Users\koksu\AppData\Local\Programs\Python\Python310  
  
# importy  
import typing  
import numpy as np  
# stałe i zmienne globalne  
  
# funkcje  
  
  
def medianize(A):  
 mean\_value = np.mean(A)  
 medianized\_array = A - mean\_value  
 return medianized\_array  
  
  
def main() -> None:  
 input\_array = np.array([1, 2, 3, 4, 5])  
 result\_array = medianize(input\_array)  
 print("Tablica pierwotna:")  
 print(input\_array)  
 print("\nTablica po użyciu funkcji medianize:")  
 print(result\_array)  
  
  
main()

**Obraz zawierający tekst, Czcionka, zrzut ekranu, design

Opis wygenerowany automatycznie**

**Zadanie 3.**

#!C:\Users\koksu\AppData\Local\Programs\Python\Python310  
  
# importy  
import typing  
import numpy as np  
# stałe i zmienne globalne  
  
# funkcje  
  
  
def print\_matrix\_and\_max\_values(matrix):  
 print("Macierz pierwotna:")  
 print(matrix)  
 global\_max = np.max(matrix)  
 print("\nElement największy globalnie:", global\_max)  
 row\_max\_values = np.max(matrix, axis=1)  
 print("\nNajwiększy element w każdym wierszu:")  
 print(row\_max\_values)  
 col\_max\_values = np.max(matrix, axis=0)  
 print("\nNajwiększy element w każdej kolumnie:")  
 print(col\_max\_values)  
  
  
def main() -> None:  
 matrix\_size = (10, 10)  
 numbers = np.arange(0, 100, 1)  
 matrix = np.reshape(numbers, matrix\_size)  
 print\_matrix\_and\_max\_values(matrix)  
  
  
main()

**Obraz zawierający tekst, zrzut ekranu, Czcionka, dokument

Opis wygenerowany automatycznie**

**Zadanie 4.**

#!C:\Users\koksu\AppData\Local\Programs\Python\Python310  
  
# importy  
import typing  
import numpy as np  
# stałe i zmienne globalne  
  
# funkcje  
  
  
def reshape\_test(matrix, shape1, shape2):  
 matrix\_backup = matrix  
 print("Macierz pierwotna:")  
 print(matrix)  
 reshaped\_matrix1 = np.reshape(matrix, (shape1, -1))  
 print("\nMacierz po użyciu reshape z liczbą -1 jako pierwszym parametrem:")  
 print(reshaped\_matrix1)  
 reshaped\_matrix2 = np.reshape(matrix\_backup, (-1, shape2))  
 print("\nMacierz po użyciu reshape z liczbą -1 jako drugim parametrem:")  
 print(reshaped\_matrix2)  
  
  
def main() -> None:  
 original\_matrix = np.array([[1, 2, 3], [4, 5, 6]])  
 reshape\_test(original\_matrix, 3, 3)  
  
  
main()  
"""Parametr -1 wskazuje, że ta wymiarowa wartość powinna być automatycznie obliczona na podstawie rozmiaru oryginalnej   
macierzy i innych wymiarów, aby zachować spójność danych."""

**Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie**

**Zadanie 5.**

#!C:\Users\koksu\AppData\Local\Programs\Python\Python310  
  
# importy  
import typing  
import numpy as np  
# stałe i zmienne globalne  
  
# funkcje  
  
  
def main() -> None:  
 grades\_data = np.genfromtxt('oceny.csv', delimiter='\t', skip\_header=1)  
 min\_lab\_grades = np.min(grades\_data[:, :5], axis=1)  
 print("Najniższa ocena z laboratoriów dla każdego studenta:")  
 print(min\_lab\_grades)  
 avg\_exam\_grade = np.mean(grades\_data[:, 5])  
 print("Średnia ocena z egzaminu:", avg\_exam\_grade)  
 num\_2\_exam\_grades = np.sum(grades\_data[:, 5] == 2)  
 print("liczba 2 z egzaminu:", num\_2\_exam\_grades)  
 has\_all\_5\_lab\_grades = np.any(np.all(grades\_data[:, :5] == 5, axis=1))  
 print("Czy jest student, który miał same 5 z laboratoriów?:", has\_all\_5\_lab\_grades)  
 has\_2\_in\_lab2\_and\_lab3 = np.any((grades\_data[:, 1] == 2) & (grades\_data[:, 2] == 2))  
 print("Czy jest student, który miał 2 z LAB2 i LAB3?:", has\_2\_in\_lab2\_and\_lab3)  
 students\_higher\_exam\_than\_avg\_lab = np.sum(grades\_data[:, 5] > avg\_exam\_grade)  
 print("Ilu studentów dostało wyższą ocenę z egzaminu niż ich średnia ocen z laboratoriów?:",  
 students\_higher\_exam\_than\_avg\_lab)  
 max\_5\_count\_student\_idx = np.argmax(np.sum(grades\_data[:, :5] == 5, axis=1))  
 num\_5\_grades\_max\_student = np.sum(grades\_data[max\_5\_count\_student\_idx, :5] == 5)  
 print("Liczba piątek, którą uzyskał student mający najwięcej 5 w całej grupie:", num\_5\_grades\_max\_student)  
  
  
main()

**Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie**

**Zadanie 6.**

#!C:\Users\koksu\AppData\Local\Programs\Python\Python310  
  
# importy  
import typing  
import numpy as np  
# stałe i zmienne globalne  
  
# funkcje  
  
  
def main() -> None:  
 random\_array = np.random.rand(10)  
 print("Wylosowana tablica:")  
 print(random\_array)  
 sorted\_array\_ascending = np.sort(random\_array)  
 print("\nTablica posortowana rosnąco:")  
 print(sorted\_array\_ascending)  
 sorted\_array\_descending = np.sort(random\_array)[::-1]  
 print("\nTablica posortowana malejąco:")  
 print(sorted\_array\_descending)  
  
  
main()

**Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie**

**Zadanie 7.**

#!C:\Users\koksu\AppData\Local\Programs\Python\Python310  
  
# importy  
import typing  
import numpy as np  
# stałe i zmienne globalne  
  
# funkcje  
  
  
def main() -> None:  
 matrix = np.random.rand(5, 5)  
 weights = np.array([1, 2, 3, 2, 1])  
 weighted\_average = np.average(matrix, axis=1, weights=weights)  
 print("Macierz:")  
 print(matrix)  
 print("\nŚrednia ważona dla każdego wiersza:")  
 print(weighted\_average)  
  
  
main()

**Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie**

**Zadanie 8.**

#!C:\Users\koksu\AppData\Local\Programs\Python\Python310  
  
# importy  
import typing  
import numpy as np  
# stałe i zmienne globalne  
  
# funkcje  
  
  
def main() -> None:  
 array = np.random.randint(0, 11, size=(10, 10))  
 print("Tablica:")  
 print(array, "\n")  
 numbers, counts = np.unique(array, return\_counts=True)  
 for number, count in zip(numbers, counts):  
 print(f"Ilość wystąpień liczby {number}: {count}")  
  
  
main()

**Obraz zawierający tekst, zrzut ekranu, menu, Czcionka

Opis wygenerowany automatycznie**

**Zadanie 9.**

#!C:\Users\koksu\AppData\Local\Programs\Python\Python310  
  
# importy  
import typing  
import numpy as np  
# stałe i zmienne globalne  
  
# funkcje  
  
  
def main() -> None:  
 array1 = np.array([[1, 2], [3, 4]])  
 array2 = np.array([[5, 6], [7, 8]])  
 star\_operator = array1 \* array2  
 print("\*:")  
 print(star\_operator)  
 asterisk\_operator = array1 @ array2  
 print("\n@:")  
 print(asterisk\_operator)  
  
  
main()  
"""Operator \* jest używany do mnożenia element-wise, co oznacza, że każdy odpowiadający sobie element w dwóch macierzach   
jest mnożony przez siebie.  
Operator @ jest używany do mnożenia macierzowego, co oznacza, że wykonuje iloczyn skalarny, a wynikowa macierz zawiera   
sumy iloczynów odpowiednich elementów."""

**Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie**

**Zadanie 10.**

#!C:\Users\koksu\AppData\Local\Programs\Python\Python310  
  
# importy  
import typing  
import numpy as np  
# stałe i zmienne globalne  
  
# funkcje  
  
  
def main() -> None:  
 array = np.array([1.123456789, 2.345678901, 3.987654321])  
 print("Domyślne opcje wyświetlania:")  
 print(array)  
 np.set\_printoptions(precision=4)  
 print("\nPrzykładowo dostosowane opcje wyświetlania (4 miejsca po przecinku):")  
 print(array)  
  
  
main()  
"""Funkcja set\_printoptions w bibliotece NumPy umożliwia dostosowanie sposobu, w jaki tablice NumPy są wyświetlane."""

**Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie**

**Wnioski:**

Laboratorium nauczyło mnie pracy z modułem numpy.