
	Politechnika Bydgoska im. J. J. Śniadeckich Wydział Telekomunikacji, Informatyki i Elektrotechniki		
Przedmiot	Skryptowe języki programowania		
Prowadzący	mgr inż. Martyna Tarczewska		
Temat	<i>Numpy</i>		
Student	Marcin Ogórkiewicz		
Nr ćw.	9	Data wykonania	28.11.2023
Ocena		Data oddania spr.	28.11.2023

Zadanie 1.

```

#!C:\Users\koku\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()

```

```
C:\Users\koku\PycharmProjects\Skr  
Macierz pierwotna:  
[[1 0 3]  
 [0 5 0]  
 [6 7 0]]  
  
Macierz po zamianie zer na 10 :  
[[ 1 10 3]  
 [10 5 10]  
 [ 6 7 10]]  
  
Process finished with exit code 0
```

Zadanie 2.

```
#!C:\Users\koku\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()
```

```
Tablica pierwotna:
[1 2 3 4 5]

Tablica po użyciu funkcji medianize:
[-2. -1.  0.  1.  2.]

Process finished with exit code 0
```

Zadanie 3.

```
#!C:\Users\koku\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()
```

```

C:\Users\koku\PycharmProjects\Skrypto
Macierz pierwotna:
[[ 0  1  2  3  4  5  6  7  8  9]
 [10 11 12 13 14 15 16 17 18 19]
 [20 21 22 23 24 25 26 27 28 29]
 [30 31 32 33 34 35 36 37 38 39]
 [40 41 42 43 44 45 46 47 48 49]
 [50 51 52 53 54 55 56 57 58 59]
 [60 61 62 63 64 65 66 67 68 69]
 [70 71 72 73 74 75 76 77 78 79]
 [80 81 82 83 84 85 86 87 88 89]
 [90 91 92 93 94 95 96 97 98 99]]

Element największy globalnie: 99

Największy element w każdym wierszu:
[ 9 19 29 39 49 59 69 79 89 99]

Największy element w każdej kolumnie:
[90 91 92 93 94 95 96 97 98 99]

Process finished with exit code 0

```

Zadanie 4.

```

#!C:\Users\koku\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."""
```

```
Macierz pierwotna:
[[1 2 3]
 [4 5 6]]

Macierz po użyciu reshape z liczbą -1 jako pierwszym parametrem:
[[1 2]
 [3 4]
 [5 6]]

Macierz po użyciu reshape z liczbą -1 jako drugim parametrem:
[[1 2 3]
 [4 5 6]]

Process finished with exit code 0
```

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()

```

```

C:\Users\koku\PycharmProjects\SkrptoweJęzykiProgramowania\venv\Scripts\python.exe C:
Najniższa ocena z laboratoriów dla każdego studenta:
[3.  3.5 3.5 5.  2.  3.5 3.5 4.5 3.5 2. ]
Średnia ocena z egzaminu: 3.45
liczba 2 z egzaminu: 2
Czy jest student, który miał same 5 z laboratoriów?: True
Czy jest student, który miał 2 z LAB2 i LAB3?: False
Ilu studentów dostało wyższą ocenę z egzaminu niż ich średnia ocen z laboratoriów?: 5
Liczba piątek, którą uzyskał student mający najwięcej 5 w całej grupie: 5

Process finished with exit code 0

```

Zadanie 6.

```

#!C:\Users\koku\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()

```

```

C:\Users\koksu\PycharmProjects\SkryptoweJęzykiProgramowania\venv\Sc
Wylosowana tablica:
[0.96980751 0.69908805 0.60662602 0.60332412 0.68794656 0.50931443
 0.02833158 0.79665623 0.18464279 0.19648556]

Tablica posortowana rosnąco:
[0.02833158 0.18464279 0.19648556 0.50931443 0.60332412 0.60662602
 0.68794656 0.69908805 0.79665623 0.96980751]

Tablica posortowana malejąco:
[0.96980751 0.79665623 0.69908805 0.68794656 0.60662602 0.60332412
 0.50931443 0.19648556 0.18464279 0.02833158]

Process finished with exit code 0

```

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()

```

```
C:\Users\koku\PycharmProjects\SkrptoweJzykiProgramowania\
Macierz:
[[0.66464333 0.99904827 0.49519769 0.91114824 0.46615311]
 [0.68521972 0.32653331 0.3036088 0.40560483 0.03816393]
 [0.95885282 0.71920252 0.5916857 0.30847223 0.18072151]
 [0.74301397 0.59313179 0.05992627 0.14751893 0.30118599]
 [0.29750995 0.58999922 0.90803322 0.34772325 0.22859867]]

Średnia ważona dla każdego wiersza:
[0.71519806 0.34427626 0.5522201 0.30058669 0.56951703]

Process finished with exit code 0
```

Zadanie 8.

```
#!C:\Users\koku\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()
```



```
C:\Users\koku\PycharmProjects\Skryp
```

Tablica:

```
[[ 7  7 10  9  1  1  8  5  6  5]
 [ 8  3  7  3  5  5  5  3  9  4]
 [ 3  3  5  6  5  3  1  2  0  2]
 [ 2  0  2  8  4  7  9  8  1  0]
 [ 3 10  8  8  3  2  2  4  6  8]
 [ 1  1  2  8  3 10  3  6  9  3]
 [10  7  1  9  8  6  6  9  6  8]
 [ 2  5  9  6  4  9  2  7  7  6]
 [ 9  8  4  6  5  6  6  3  6  5]
 [10  2  4  7  3  5  0  5  2  1]]
```

Ilość wystąpień liczby 0: 4

Ilość wystąpień liczby 1: 8

Ilość wystąpień liczby 2: 11

Ilość wystąpień liczby 3: 13

Ilość wystąpień liczby 4: 6

Ilość wystąpień liczby 5: 12

Ilość wystąpień liczby 6: 13

Ilość wystąpień liczby 7: 8

Ilość wystąpień liczby 8: 11

Ilość wystąpień liczby 9: 9

Ilość wystąpień liczby 10: 5

Process finished with exit code 0

Zadanie 9.

```
#!C:\Users\koku\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."""

```

```

C:\Users\koksu\PycharmProjects\Skrypto
*:
[[ 5 12]
 [21 32]]

@:
[[19 22]
 [43 50]]

Process finished with exit code 0

```

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."""

```

```
C:\Users\koku\PycharmProjects\SkryptoweJęzykiProgramowania\venv\Scripts
Domyślne opcje wyświetlania:
[1.12345679 2.3456789 3.98765432]

Przykładowo dostosowane opcje wyświetlania (4 miejsca po przecinku):
[1.1235 2.3457 3.9877]

Process finished with exit code 0
```

Wnioski:

Laboratorium nauczyło mnie pracy z modułem numpy.