

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ
ФЕДЕРАЦИИ**

**Федеральное государственное автономное
образовательное учреждение высшего образования
«СЕВЕРОКАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»**

**Кафедра инфокоммуникаций
Институт цифрового развития**

ОТЧЁТ
по лабораторной работе №3.2
Дисциплина: «Анализ данных»
Тема: «Основы работы с библиотекой NumPy»

Выполнил: студент 2 курса
группы ИВТ-б-о-21-1
Кочкаров Умар Ахматович

Ставрополь 2023

Цель работы: исследовать базовые возможности библиотеки NumPy языка программирования Python.

Ход работы:

1. Создать общедоступный репозиторий с лицензией MIT и языком программирования Python

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Owner *

 umarkochkarov ▾

Repository name *

/ lb3.2 ✓

Great repository names are short and memorable. Need inspiration? How about [urban-umbrella?](#)

Description (optional)



Public

Anyone on the internet can see this repository. You choose who can commit.



Private

You choose who can see and commit to this repository.

Initialize this repository with:

Skip this step if you're importing an existing repository.



Add a README file

This is where you can write a long description for your project. [Learn more.](#)

Add .gitignore


Choose which files not to track from a list of templates. [Learn more.](#)

.gitignore template: Python ▾

Choose a license

A license tells others what they can and can't do with your code. [Learn more.](#)

License: MIT License ▾

This will set  main as the default branch. Change the default name in your [settings](#).



You are creating a public repository in your personal account.

Create repository

Рисунок 1. Создание репозитория

2. Клонировать репозиторий на ПК:

```
erken@LAPTOP-ESTC60GF MINGW64 ~/Desktop/python/Ла63.2
$ git clone https://github.com/umarkochkarov/lb3.2.git
Cloning into 'lb3.2'...
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 5 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (5/5), done.
```

Рисунок 2. Клонирование репозитория на ПК

3. Организовать репозиторий в соответствии с моделью ветвления git-flow:

```
erken@LAPTOP-ESTC60GF MINGW64 ~/Desktop/python/Ла63.2/lb3.2 (main)
$ git flow init

Which branch should be used for bringing forth production releases?
- main
Branch name for production releases: [main]
Branch name for "next release" development: [develop]

How to name your supporting branch prefixes?
Feature branches? [feature/] Bugfix branches? [bugfix/]
Release branches? [release/]
Hotfix branches? [hotfix/]
Support branches? [support/]
Version tag prefix? []
Hooks and filters directory? [C:/Users/erken/Desktop/python/Ла63.2/lb3.2/.git/hooks]
```

Рисунок 3. Организация репозитория в соответствии с git-flow

4. Проработка примеров лабораторной работы:

1)

```
In [2]: import numpy as np
        m = np.matrix('1 2 3 4; 5 6 7 8; 9 1 5 7')
        print(m)

[[1 2 3 4]
 [5 6 7 8]
 [9 1 5 7]]

In [3]: m[1, 0]
Out[3]: 5

In [4]: m[1, :]
Out[4]: matrix([[5, 6, 7, 8]])

In [5]: m[:, 2]
Out[5]: matrix([[3],
                [7],
                [5]])

In [6]: m[1, 2:]
Out[6]: matrix([[7, 8]])

In [7]: m[0:2, 1]
Out[7]: matrix([[2],
                [6]])

In [8]: m[0:2, 1:3]
Out[8]: matrix([[2, 3],
                [6, 7]])

In [9]: cols = [0, 1, 3]
        m[:, cols]
Out[9]: matrix([[1, 2, 4],
                [5, 6, 8],
                [9, 1, 7]])
```

Рисунок 4. Пример 1

2)

```
In [1]: import numpy as np
m = np.matrix('1 2 3 4; 5 6 7 8; 9 1 5 7')
print(m)

[[1 2 3 4]
 [5 6 7 8]
 [9 1 5 7]]

In [2]: type(m)
Out[2]: numpy.matrix

In [3]: m = np.array(m)
type(m)
Out[3]: numpy.ndarray

In [4]: m.shape
Out[4]: (3, 4)

In [5]: m.max()
Out[5]: 9

In [6]: np.max(m)
Out[6]: 9

In [8]: m = np.matrix(m)
m.max(axis=1)
Out[8]: matrix([[4],
               [8],
               [9]])

In [9]: m.max(axis=0)
Out[9]: matrix([[9, 6, 7, 8]])

In [10]: m.mean()
Out[10]: 4.833333333333333

In [11]: m.mean(axis=1)
Out[11]: matrix([[2.5],
               [6.5],
               [5.5]])

In [13]: m.sum(axis=0)
Out[13]: matrix([[15, 9, 15, 19]])
```

Рисунок 5. Пример 2

3)

```
In [1]: import numpy as np
        nums = np.array([1, 2, 3, 4, 5, 6, 7, 8, 9, 10])
        letters = np.array(['a', 'b', 'c', 'd', 'a', 'e', 'b'])

In [2]: less_than_5 = nums < 5
        less_than_5

Out[2]: array([ True,  True,  True,  True, False, False, False, False, False,
               False])

In [3]: pos_a = letters == 'a'
        pos_a

Out[3]: array([ True, False, False, False,  True, False, False])

In [4]: nums[less_than_5]

Out[4]: array([1, 2, 3, 4])

In [6]: nums[nums < 5] = 10
        print(nums)

[10 10 10 10  5  6  7  8  9 10]

In [7]: m = np.matrix('1 2 3 4; 5 6 7 8; 9 1 5 7')
        print(m)

[[1 2 3 4]
 [5 6 7 8]
 [9 1 5 7]]

In [8]: mod_m = np.logical_and(m >= 3, m <= 7)
        mod_m

Out[8]: matrix([[False, False,  True,  True],
               [ True,  True,  True, False],
               [False, False,  True,  True]])

In [9]: m[mod_m]

Out[9]: matrix([[3, 4, 5, 6, 7, 5, 7]])

In [10]: m[m > 7] = 25
         print(m)

[[ 1  2  3  4]
 [ 5  6  7 25]
 [25  1  5  7]]
```

Рисунок 6. Пример 3

4)

```
In [1]: import numpy as np
        np.arange(10)

Out[1]: array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])

In [2]: np.arange(5, 12)

Out[2]: array([ 5,  6,  7,  8,  9, 10, 11])

In [3]: np.arange(1, 5, 0.5)

Out[3]: array([1. , 1.5, 2. , 2.5, 3. , 3.5, 4. , 4.5])

In [5]: A = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])
        A

Out[5]: array([[1, 2, 3],
               [4, 5, 6],
               [7, 8, 9]])

In [6]: np.ravel(A)

Out[6]: array([1, 2, 3, 4, 5, 6, 7, 8, 9])

In [7]: np.ravel(A, order='C')

Out[7]: array([1, 2, 3, 4, 5, 6, 7, 8, 9])

In [8]: np.ravel(A, order='F')

Out[8]: array([1, 4, 7, 2, 5, 8, 3, 6, 9])

In [9]: a = np.array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
        np.where(a % 2 == 0, a * 10, a / 10)

Out[9]: array([ 0. ,  0.1, 20. ,  0.3, 40. ,  0.5, 60. ,  0.7, 80. ,  0.9])

In [10]: a = np.random.rand(10)
         a

Out[10]: array([0.55642332, 0.37526446, 0.24589265, 0.95861707, 0.18235878,
                0.44916279, 0.67206023, 0.48754707, 0.09085441, 0.14104925])

In [11]: np.where(a > 0.5, True, False)

Out[11]: array([ True, False, False,  True, False, False,  True, False, False,
                False])

In [12]: np.where(a > 0.5, 1, -1)

Out[12]: array([ 1, -1, -1,  1, -1, -1,  1, -1, -1, -1])
```

Рисунок 7. Пример 4

5)

```
In [1]: import numpy as np

In [2]: x = np.linspace(0, 1, 5)
x
Out[2]: array([0. , 0.25, 0.5 , 0.75, 1.  ])

In [4]: y = np.linspace(0, 2, 5)
y
Out[4]: array([0. , 0.5, 1. , 1.5, 2. ])

In [5]: xg, yg = np.meshgrid(x, y)
xg
Out[5]: array([[0. , 0.25, 0.5 , 0.75, 1.  ],
               [0. , 0.25, 0.5 , 0.75, 1.  ],
               [0. , 0.25, 0.5 , 0.75, 1.  ],
               [0. , 0.25, 0.5 , 0.75, 1.  ],
               [0. , 0.25, 0.5 , 0.75, 1.  ]])

In [6]: yg
Out[6]: array([[0. , 0. , 0. , 0. , 0. ],
               [0.5, 0.5, 0.5, 0.5, 0.5],
               [1. , 1. , 1. , 1. , 1. ],
               [1.5, 1.5, 1.5, 1.5, 1.5],
               [2. , 2. , 2. , 2. , 2. ]])

In [7]: import matplotlib.pyplot as plt
%matplotlib inline
plt.plot(xg, yg, color="r", marker="*", linestyle="none")
Out[7]: [<matplotlib.lines.Line2D at 0x14b74887fd0>,
<matplotlib.lines.Line2D at 0x14b7482a160>,
<matplotlib.lines.Line2D at 0x14b7482a1c0>,
<matplotlib.lines.Line2D at 0x14b7482a2e0>,
<matplotlib.lines.Line2D at 0x14b7482a400>]
```

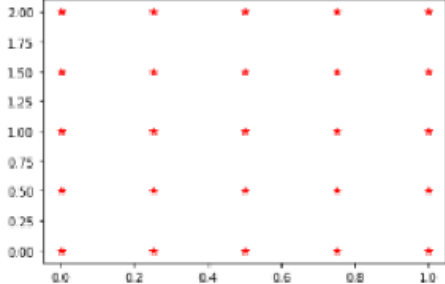


Рисунок 8. Пример 5

6)

```
In [1]: import numpy as np

In [2]: np.random.permutation(7)
Out[2]: array([4, 6, 5, 3, 2, 1, 0])

In [3]: a = ['a', 'b', 'c', 'd', 'e']
         np.random.permutation(a)
Out[3]: array(['a', 'e', 'c', 'd', 'b'], dtype='<U1')

In [5]: arr = np.linspace(0, 10, 5)
         arr
Out[5]: array([ 0. ,  2.5,  5. ,  7.5, 10. ])

In [6]: arr_mix = np.random.permutation(arr)
         arr_mix
Out[6]: array([ 2.5, 10. ,  5. ,  0. ,  7.5])

In [8]: index_mix = np.random.permutation(len(arr_mix))
         index_mix
Out[8]: array([4, 1, 0, 3, 2])

In [9]: arr[index_mix]
Out[9]: array([10. ,  2.5,  0. ,  7.5,  5. ])
```

Рисунок 9. Пример 6

5. Индивидуальное задание:

1) Создать ноутбук, в котором выполнить решение индивидуального задания. Ноутбук должен содержать условие индивидуального задания. При решении индивидуального задания не должны быть использованы условный оператор if, а также операторы циклов while и for, а только средства библиотеки NumPy. Привести в ноутбуке обоснование принятых решений.

Вариант 6. Для заданной матрицы размером 8 на 8 найти такие k , что k -я строка матрицы совпадает с k -м столбцом. Найти сумму элементов в тех строках, которые содержат хотя бы один отрицательный элемент.


```

Ввод [1]: import random

matrix = [[random.randint(-10, 10) for j in range(8)] for i in range(8)]

print("Матрица 8 на 8:")
for row in matrix:
    print(row)

matching_indexes = [i for i in range(8) if matrix[i][i] == matrix[0][i]]

if len(matching_indexes) > 0:
    print(f"Строки и столбцы совпадают в строках {' '.join(str(i) for i in matching_indexes)}")
else:
    print("Совпадений строк и столбцов не найдено.")

negative_rows_sum = 0

for row in matrix:
    if any(i < 0 for i in row):
        negative_rows_sum += sum(row)

print(f"Сумма элементов в строках с отрицательными элементами: {negative_rows_sum}")

```

Матрица 8 на 8:

```

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

```

Строки и столбцы совпадают в строках 0
Сумма элементов в строках с отрицательными элементами: 4

Рисунок 10. Индивидуальное задание 1

2) Индивидуальное задание 2: Дан маятник массой m и длиной l . Начальная скорость маятника равна 0, а начальный угол отклонения от вертикали составляет θ_0 . Не учитывая сопротивление воздуха, определить максимальный угол отклонения маятника от вертикали и время, за которое он достигнет этого угла.

```
Ввод [1]: import math
```

```
Ввод [3]: m=1  
l=1  
x=30  
g = 9.81
```

```
Ввод [5]: x = math.radians(x)
```

```
Ввод [6]: T = 2*math.pi*math.sqrt(l/g)
```

```
Ввод [7]: print("Период колебаний маятника: {:.2f} секунд".format(T))
```

Период колебаний маятника: 2.01 секунд

Рисунок 11. Индивидуальное задание 2

Контрольные вопросы:

1. Каково назначение библиотеки NumPy?

Numpy – это библиотека для языка программирования Python, которая предоставляет в распоряжение разработчика инструменты для эффективной работы с многомерными массивами и высокопроизводительные вычислительные алгоритмы.

2. Что такое массивы ndarray?

Этот объект является многомерным однородным массивом с заранее заданным количеством элементов.

3. Как осуществляется доступ к частям многомерного массива?

В квадратных скобках указывается номер строки – первой цифрой и номер столбца – второй. Двоеточие означает “все элементы”, первый элемент – это номер строки, второй – указание на то, что необходимо взять элементы всех столбцов матрицы.

4. Как осуществляется расчет статистик по данным?

Для расчета той или иной статистики, соответствующую функцию можно вызвать как метод объекта, с которым вы работаете. Если необходимо найти максимальный элемент в каждой строке, то для этого нужно передать в качестве аргумента параметр axis=1. Для вычисления статистики по столбцам, передайте в качестве параметра аргумент axis=0.

5. Как выполняется выборка данных из массивов ndarray?

Использование boolean массивов для доступа к данным порой является более лучшим вариантом, чем использование численных индексов. Как вы знаете, в Python есть такой тип данных – boolean. Переменные этого типа принимают одно из двух значений: True или False. Такие переменные можно создать самостоятельно, либо они могут являться результатом какого-то выражения. Самым замечательным в использовании boolean массивов при работе с ndarray является то, что их можно применять для построения выборок. Если мы переменную `less_than_5` передадим в качестве списка индексов для `nums`, то получим массив, в котором будут содержаться элементы из `nums` с индексами равными индексам True позиций массива `less_than_5`.

Вывод: исследованы базовые возможности библиотеки NumPy для языка программирования Python.