



# Интеллектуальная обработка данных

Востриков Александр  
Владимирович

К.т.н., доцент

Каб. 522

E-mail: [avostrikov@hse.ru](mailto:avostrikov@hse.ru)

# Преподаватели

- Клышинский Эдуард Станиславович – лектор.
- Востриков Александр Владимирович – практика + лабы.



# Как мы будем жить?

- 3, 4 модуль
- Язык программирования - Python
- 4 лабораторные работы и курсовая + практика
- Задания на лабы:

[docs.google.com/document/d/1PJW4Jj5d7W4QLy5MsBIRZmR1dJvKZu1J1Vjh9uLVvql/edit](https://docs.google.com/document/d/1PJW4Jj5d7W4QLy5MsBIRZmR1dJvKZu1J1Vjh9uLVvql/edit)

- Контроль активностей студентов на гугл-диске:

<https://docs.google.com/spreadsheets/d/1t9b3tlWcINX0HxrHRvOLaFx3AWPgRUEjLEEZNIkEwH0/edit?usp=sharing>

# Формула оценивания

- Оценка =  $0.2 * \text{Экзамен} + 0.1 * \text{Тест} + 0.4 * \text{Лабораторные работы} + 0.2 * \text{Проект} + 0.1 * \text{Активность}$

# Сопровождение практической части курса

- <https://edu.hse.ru/user/index.php?id=188926>



# Программное обеспечение

- Anaconda Community

Для работы нам понадобятся приложения:

- JupyterLab
- Jupyter notebook (среда является свободно распространяемой и позволяет выполнять отдельные фрагменты кода, а не программу целиком. В качестве альтернативы можно использовать полноценную среду разработки PyCharm или другие)
- VS Code
- Библиотеки можно скачать тут <https://www.lfd.uci.edu/~gohlke/pythonlibs/>



# Альтернативное ПО

- Гугл колаб: <https://colab.research.google.com>
- Для работы нужен аккаунт в гугл
- Преимущества
  - высокая скорость работы с нейронными сетями
  - Нет необходимости скачивать библиотеки

# Рекомендуемая литература

- Уэс Маккинли — Python и анализ данных - Издательство "ДМК Пресс" - 2015 - ISBN: 978-5-97060-315-4 - Текст электронный // ЭБС Лань - URL: <https://e.lanbook.com/book/73074>
- Крупномасштабное машинное обучение вместе с Python : учитесь быстро создавать мощные модели машинного обучения и развертывать крупномасштабные приложения прогнозирования, Шарден, Б., Массарон, Л., 2018



# Содержание учебной дисциплины

- Обработка данных с использованием библиотеки Pandas
  - Основные возможности библиотеки Pandas: загрузка и выборка данных, агрегирование данных, нормализация данных.
- Визуализация данных и их анализ
  - Основные виды графиков для отображения данных: диаграммы рассеяния, размаха, гистограммы, эпюры, отображение трехмерных данных, отображение последовательностей. Элементы графика: оси, легенда, надписи. Методы снижения размерности пространства признаков: метод главных компонент, многомерное шкалирование, t-SNE, UMAP.
- Кластеризация данных
  - Метод k-средних, спектральная кластеризация. Методы, основанные на оценке плотности распределения точек в пространстве. Методы оценки точности кластеризации.
- Классификация данных
  - Линейные методы классификации данных: линейная и логистическая регрессия, SMV. Методы, основанные на деревьях принятия решения, в том числе, метод случайного леса. Метод k ближайших соседей. Методы бустинга и стеккинга. Методы оценки результатов классификации: точность, полнота, f-мера, ROC-AUC, матрица ошибок.

# Содержание учебной дисциплины

## - Обработка изображений

- Основные форматы хранения изображений и их отличия: BMP, PNG, JPG, GIF. Методы внесения изменений в изображения. Методы выделения областей изображения при помощи кластеризации точек. Библиотеки Python для работы с изображениями. Библиотека OpenCV и методы обработки изображений в sklearn.

## - Обработка текстов на естественном языке

- Лексический и синтаксический анализ текстов. Понятия пространства признаков для текста. Задачи обработки текстов: выделение именованных сущностей, фактов. Классификация и кластеризация текстов. Технологии Word2Vec, Glove.

## - «Плотные» нейронные сети

- Бионические и искусственные нейронные сети, нейрон МакКаллока и Питтса, персептрон, сети Кохонена. Понятия порогового элемента и функции. Применение нейронных сетей для решения задач классификации и преобразования данных.

## - Глубинное обучение нейронных сетей

- Понятие свертки в нейронных сетях, сверточные нейронные сети. Введение обратной связи в нейронной сети, рекуррентные сети. Виды сверточных и рекуррентных нейронных сетей. Построение архитектуры нейронной сети для решения прикладных задач.

# Содержание практических занятий

Семинар 1 - Введение в Python и numpy

Семинар 2 - Знакомство с pandas

Семинар 3 - Обработка изображений

Семинар 4 - Векторное представление данных, виды данных

Семинар 5 - Кластеризация

Семинар 6 - Обработка видео в OpenCV

Семинар 7 - Классификация (линейная и логистическая регрессия, к ближайших соседей, дендрограммы, оценка точности классификации)

Семинар 8 - Классификация (деревья принятия решений, бустинг, ансамблирование)

Семинар 9 - Регулярные выражения

Семинар 10 - Библиотека requests

Семинар 11 - Методы обработки текстов

Семинар 12 - Методы сокращения размерности пространства признаков

Семинар 13 - Семантическое пространство при обработке текстов

Семинар 14 - Полносвязанные нейронные сети (нейрон, пороговая функция, персептрон)

Семинар 15 - Сверточные нейронные сети, рекуррентные нейронные сети

Семинар 16 - Обучение с подкреплением

Семинар 17 - Визуализация данных при помощи matplotlib и seaborn

# Циклы for и while

```
for number in range(5):  
    print(number)
```

---

```
for number in [0, 1, 2, 3, 4]:  
    print(number)
```

```
i = 0  
while i < 10:  
    print(i) i = i + 1
```

---

```
while i < 10:  
    print(i)  
    if i == 5:  
        break i += 1
```

# Списки, функции, файлы, библиотеки

```
spisok=[a, 'qwe', [12, 13], 1.5]
```

Проверить наличие  
элемента в списке (и  
не только) можно при  
помощи оператора **in**

```
'qwe' in spisok
```

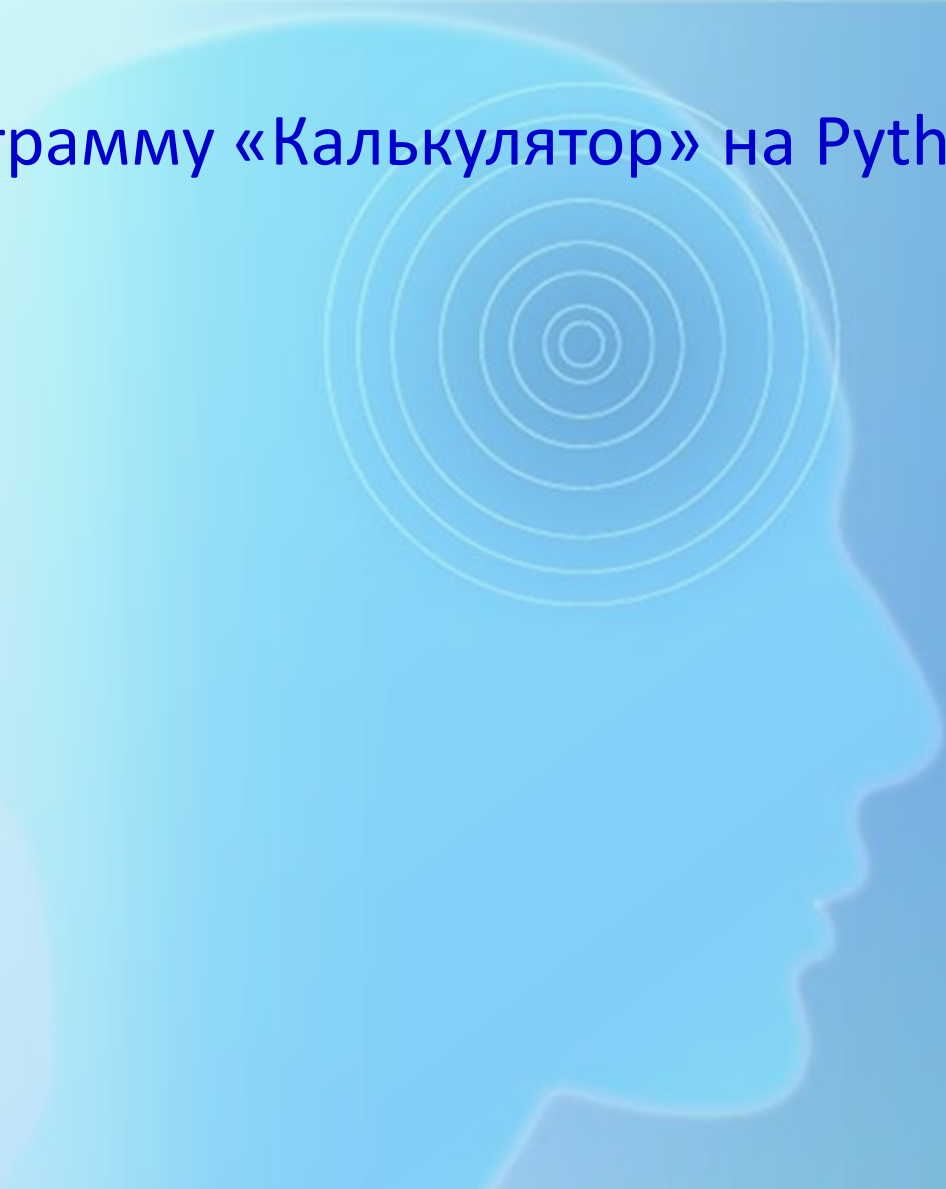
```
def sumIt(l):  
    s=0  
    for el in l:  
        s+=el  
    return s  
sumIt([1,2,3,4])
```

```
import random
```

```
fil=open("tst.txt", "wt")  
fil.write("Текстовая строка и число"+str(89))  
fil.write("И переносы только \nкогда скажем\n")  
fil.close()
```

# Задание


- Напишите программу «Калькулятор» на Python



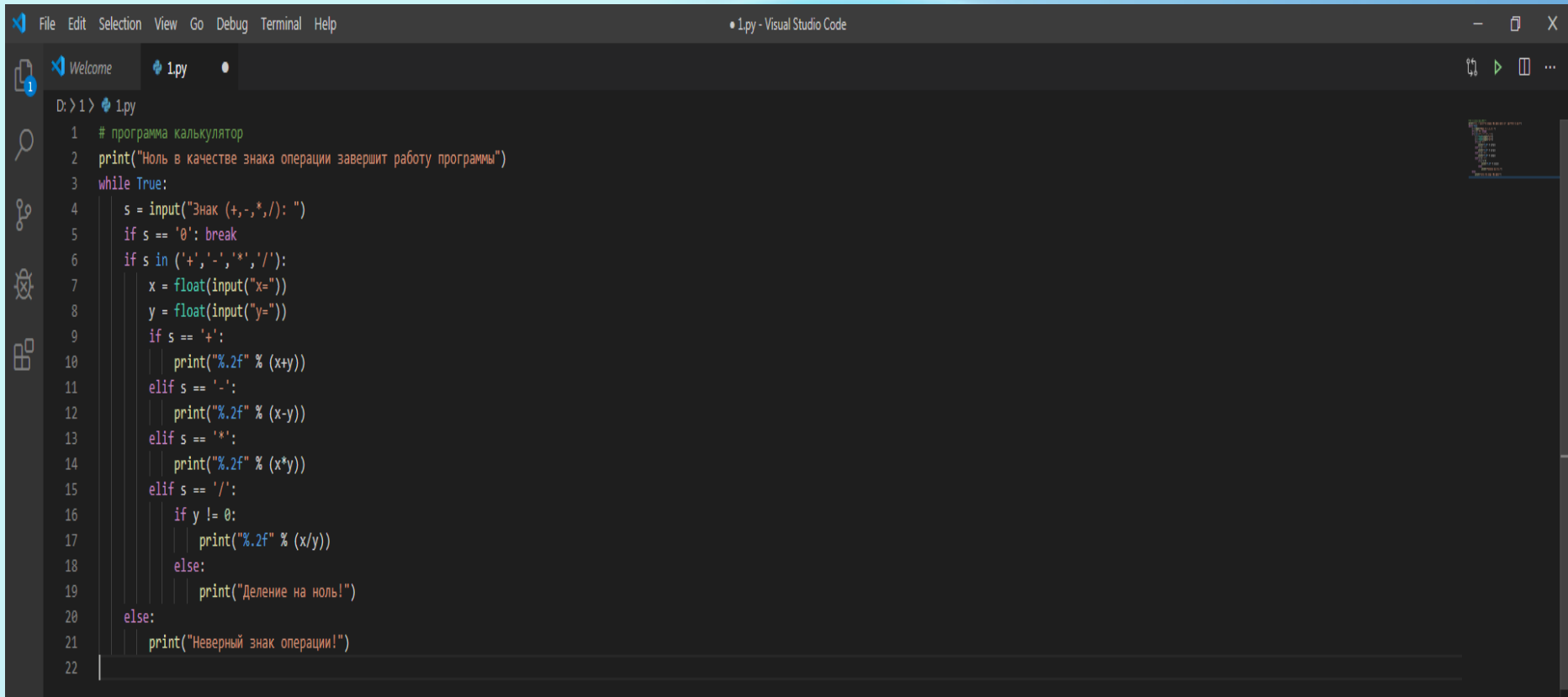


# Программа калькулятор

```
# программа калькулятор
print("Ноль в качестве знака операции завершит работу программы")
while True:
    s = input("Знак (+,-,*,/): ")
    if s == '0': break
    if s in ('+', '-', '*', '/'):
        x = float(input("x="))
        y = float(input("y="))
        if s == '+':
            print("%.2f" % (x+y))
        elif s == '-':
            print("%.2f" % (x-y))
        elif s == '*':
            print("%.2f" % (x*y))
        elif s == '/':
            if y != 0:
                print("%.2f" % (x/y))
            else:
                print("Деление на ноль!")
    else:
        print("Неверный знак операции!")
```



# Программа калькулятор



The image shows a screenshot of the Visual Studio Code editor interface. The top menu bar includes File, Edit, Selection, View, Go, Debug, Terminal, and Help. The title bar indicates the file is '1.py' in the 'Visual Studio Code' workspace. The editor window displays a Python script for a calculator. The code is as follows:

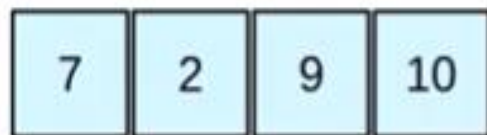
```
D:\> 1> 1.py
1 # программа калькулятор
2 print("Ноль в качестве знака операции завершит работу программы")
3 while True:
4     s = input("Знак (+,-,*,/): ")
5     if s == '0': break
6     if s in ('+', '-', '*', '/'):
7         x = float(input("x="))
8         y = float(input("y="))
9         if s == '+':
10             print("%.2f" % (x+y))
11         elif s == '-':
12             print("%.2f" % (x-y))
13         elif s == '*':
14             print("%.2f" % (x*y))
15         elif s == '/':
16             if y != 0:
17                 print("%.2f" % (x/y))
18             else:
19                 print("Деление на ноль!")
20     else:
21         print("Неверный знак операции!")
22
```

# Библиотека NumPy

- NumPy это open-source модуль для python, который предоставляет общие математические и числовые операции в виде пре-скомпилированных, быстрых функций.
- Функционал можно сравнить с функционалом MatLab.
- NumPy (Numeric Python) предоставляет базовые методы для манипуляции с большими массивами и матрицами. SciPy (Scientific Python) расширяет функционал numpy огромной коллекцией полезных алгоритмов, таких как минимизация, преобразование Фурье, регрессия, и другие прикладные математические техники.

# Библиотека NumPy

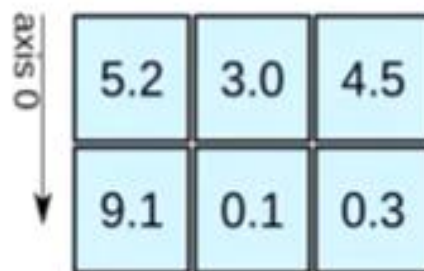
1D array



axis 0 →

shape: (4,)

2D array

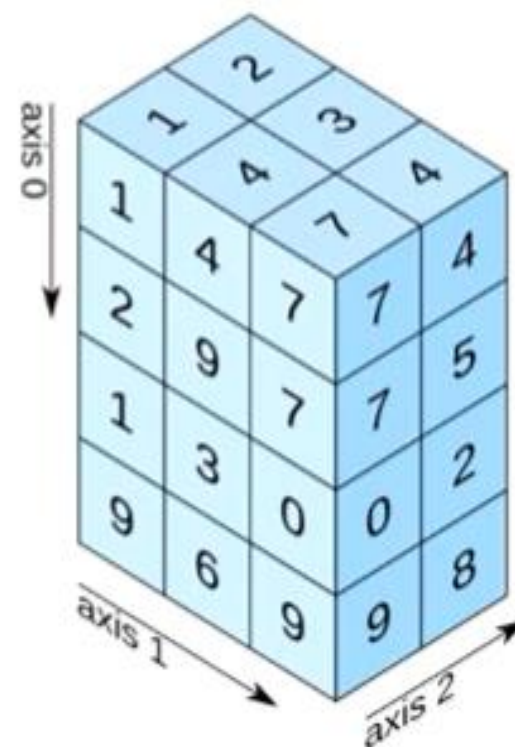


axis 0 ↓

axis 1 →

shape: (2, 3)

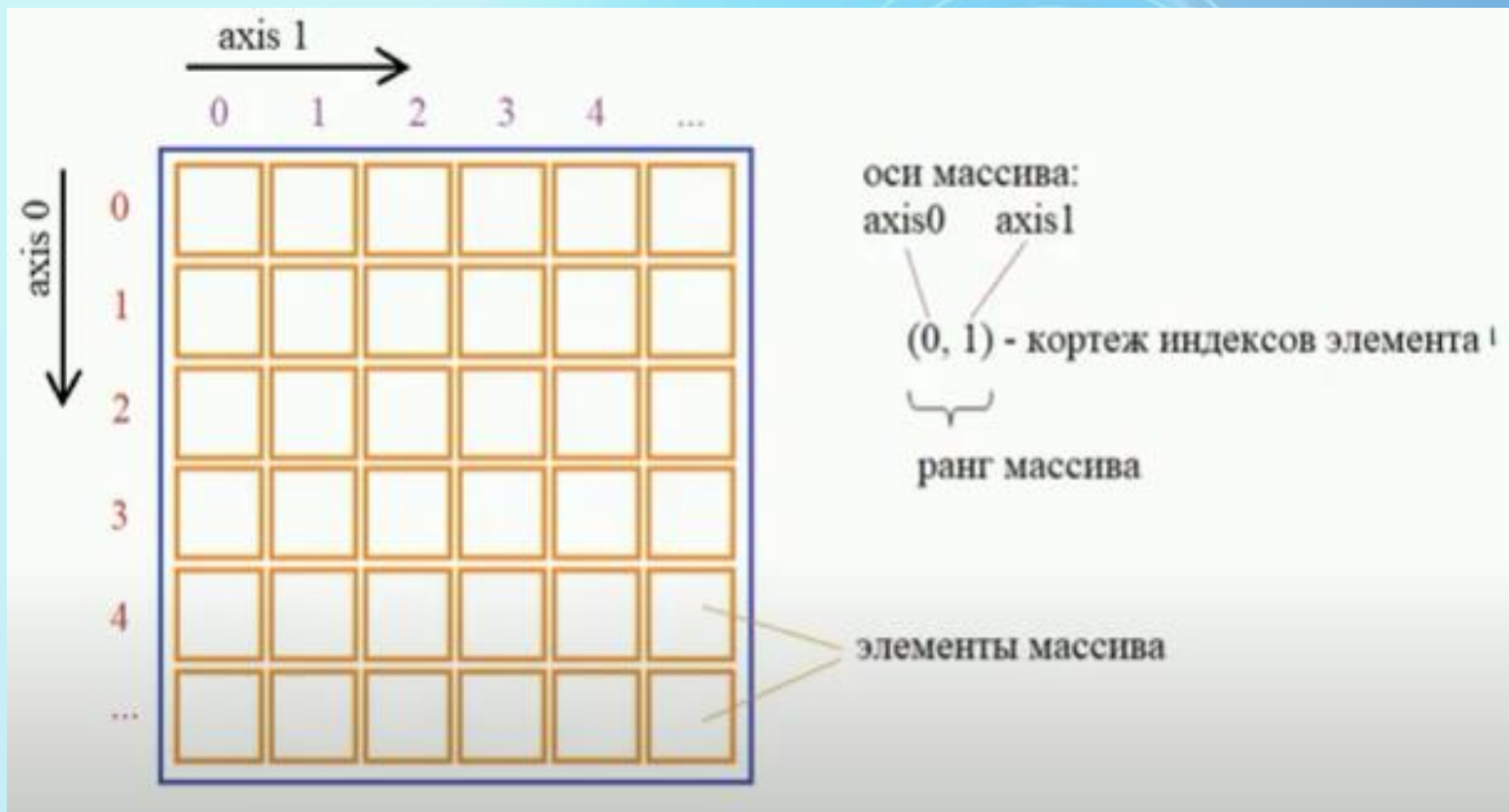
3D array



shape: (4, 3, 2)

Четырехмерный, пятимерный и более массив — это **тензор**

# Многомерный массив



# Пример работы с NumPy

- <https://colab.research.google.com/drive/1UWqWcYOxJQKn-5h6ZmLOpsiOsufVY6br?usp=sharing>





# Функционал NumPy

```
>>> import numpy as np
>>> a = np.array([1, 2, 3])
>>> a
array([1, 2, 3])
```

```
>>> type(a)
<class 'numpy.ndarray'>
```

```
>>> np.zeros((3, 5))
array([[ 0.,  0.,  0.,  0.,  0.], [ 0.,  0.,  0.,  0.,  0.], [ 0.,  0.,  0.,  0.,  0.]])
```

```
>>> np.ones((2, 2, 2))
array([[[ 1.,  1.], [ 1.,  1.]], [[ 1.,  1.], [ 1.,  1.]])
```

```
>>> np.linspace(0, 2, 9)
# 9 чисел от 0 до 2 включительно
array([ 0. , 0.25, 0.5 , 0.75, 1.  , 1.25, 1.5 , 1.75, 2.  ])
```

# Самостоятельная работа

1. Произвести произвольное арифметическое действие с двумя матрицами с шагом элементов 2.
  2. Сложить результат с единичной матрицей.
  3. Найти корень произведения всех элементов результирующей матрицы.
- 