Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования «Южно-Уральский государственный университет (национальный исследовательский университет)»

Высшая школа экономики и управления

Кафедра Информационных технологий в экономике

Программирование на языке Python

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА К КУРСОВОЙ РАБОТЕ

по дисциплине «ПРОГРАММИРОВАНИЕ»

ЮУрГУ– 380305. 2022.123.ПЗ КР

|  |  |
| --- | --- |
|  |  |
| Рецензент, \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_    \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /\_\_\_\_\_\_\_\_\_\_\_\_/    «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_ г. | Руководитель, \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_    \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /\_\_\_\_\_\_\_\_\_\_\_\_\_/    «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_ г. |
| Нормоконтролёр, \_\_\_\_\_\_\_\_\_\_\_\_\_\_    \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /\_\_\_\_\_\_\_\_\_\_\_\_/    «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_ г. | Надеин Дмитрий Денисович ЭУ - 120    \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/\_\_\_\_\_\_\_\_\_\_\_\_\_\_/ |
|  | Работа защищена с оценкой  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/ |

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_ г.

Челябинск 2020

# Аннотация

НадеинД.Д.   
Программирование на языке Python – Челябинск: ЮУрГУ, ЭУ - 120, 2020

Данная курсовая работа состоит из трех частей.

В рамках первой части этой курсовой работы были написаны и исследованы на эффективность 3 основные метода сортировки: сортировка включением (insert), обменная сортировка (bubble) и сортировка выбором (select).

Во второй части курсовой работы была рассмотрена библиотека numpy и при помощи её решена 31 задача на работу с матрицами.

Третья часть курсовой работы состоит из решения систем линейных уравнений методом Гаусса.

Аннотация. 2

Введение. 3

Курсовая работа — часть 1: "Методы сортировки". 5

1.1. Алгоритмы сортировки. 5

1.1.1. Сортировка включением (insert). 5

1.1.2. Обменная сортировка (bubble). 6

1.1.3. Сортировка выбором (select). 7

1.2. Анализ эффективности. 8

Курсовая работа — часть 2: Матричная математика и работа с пакетами. 9

Курсовая работа — часть 3: Решение слау. 10

Заключение. 11

# Введение

Актуальность темы. На данный момент одним из самых популярных языков программирования является python. В данной курсовой работе рассмотрено его применение для сортировки массивов, также рассмотрено применение одной из его библиотек для решения примеров линейной алгебры и систем линейных уравнений методом Гаусса.

Благодаря разнообразию библиотек при помощи python мы можем решать множество различных задач, что очень важно бля программиста.

Цели работы:

1. Методом анализа выявить наиболее эффективный метод сортировки.
2. С помощью библиотеки numpy решить 31 задачу.
3. При помощи библиотеки numpy решить системы линейных уравнений методом Гаусса.

# Курсовая работа — часть 1: "Методы сортировки"

## Алгоритмы сортировки.

«… Сортировка к тому же, еще и сама достаточно хороший пример задачи, которую можно решать с помощью многих различных алгоритмов. Каждый из них имеет и свои достоинства, и свои недостатки, и выбирать алгоритмы нужно исходя из конкретной постановки задачи.

В общем, под сортировкой мы будем понимать процесс перегруппировки заданного множества объектов в некотором определенном порядке. Цель сортировки – облегчить последующий поиск элементов в таком отсортированном множестве. Это почти универсальная, фундаментальная деятельность. Мы встречаемся с отсортированными объектами в телефонных книгах, в списках подоходных налогов, в оглавлениях книг, в библиотеках, в словарях, на складах – почти везде, где нужно искать хранимые объекты. Даже малышей учат держать свои вещи «в порядке», и они уже сталкиваются с некоторыми видами сортировок задолго до того, как познакомятся с азами арифметики».

Н. Вирт — Алгоритмы + данные = программы

### Сортировка включением (insert).

Одним из наиболее простых и естественных методов внутренней сортировки является сортировка простыми включениями. Идея алгоритма очень проста. Пусть имеется массив ключей Arr0, Arr1, ..., ArrN‑1. Для каждого элемента массива, начиная со второго, производится сравнение с элементами с меньшим индексом. Элемент Arri последовательно сравнивается с элементами Arrj, где jЄ[i‑1;0], т.е. изменяется от i‑1 до 0. До тех пор, пока для очередного элемента Arrj выполняется соотношение Arrj > Arri, Arri и Arrj меняются местами. Если удается встретить такой элемент Arrj, что Arrj ≤ Arri, или если достигнута нижняя граница массива, производится переход к обработке элемента Arri+1. Так продолжается до тех пор, пока не будет достигнута верхняя граница массива.

Легко видеть, что в лучшем случае, когда массив уже упорядочен для выполнения алгоритма с массивом из N элементов потребуется N‑1 сравнение и 0 пересылок. В худшем случае, когда массив упорядочен в обратном порядке потребуется N(N‑1)/2 сравнений и столько же пересылок. Таким образом, можно оценивать сложность метода простых включений как O(N2).

Можно сократить число сравнений, применяемых в методе простых включений, если воспользоваться тем, что при обработке элемента Arri массива элементы Arr0, Arr1, ..., Arri‑1 уже упорядочены, и воспользоваться для поиска элемента, с которым должна быть произведена перестановка, методом двоичного деления. В этом случае оценка числа требуемых сравнений становится O(N\*Log(N)). Заметим, что поскольку при выполнении перестановки требуется сдвижка на один элемент нескольких элементов, то оценка числа пересылок остается O(N2). Алгоритм сортировки включением, оформленный в виде функции приведен ниже.

### Обменная сортировка (bubble)

Простая обменная сортировка, называемая «методом пузырька», для массива Arr0, Arr2, ..., ArrN‑1 работает следующим образом. Начиная с конца массива сравниваются два соседних элемента ArrN‑1 и ArrN‑2. Если выполняется условие ArrN‑2 > ArrN‑1, то они меняются местами. Процесс продолжается для ArrN‑2 и ArrN‑3 и т.д., пока не будет произведено сравнение Arr1 и Arr0. Понятно, что после этого на месте Arr0 окажется элемент с наименьшим значением. На втором шаге процесс повторяется, но последними сравниваются Arr2 и Arr1. И так далее. На последнем шаге будут сравниваться только текущие значения ArrN‑1 и ArrN‑2. Понятна аналогия с пузырьком, поскольку наименьшие элементы, самые «легкие», постепенно «всплывают» к верхней границе массива.

Для метода обменной сортировки требуется число сравнений N(N‑1)/2, минимальное число пересылок 0, а среднее и максимальное число пересылок − O(N2).

Метод пузырька допускает три простых усовершенствования. Во-первых, если на некотором шаге не было произведено ни одного обмена, то выполнение алгоритма можно прекращать. Во-вторых, можно запоминать наименьшее значение индекса массива, для которого на текущем шаге выполнялись перестановки. Очевидно, что верхняя часть массива до элемента с этим индексом уже отсортирована, и на следующем шаге можно прекращать сравнения значений соседних элементов при достижении такого значения индекса. В-третьих, метод пузырька работает неравноправно для «легких» и «тяжелых» значений. Легкое значение попадает на нужное место за один шаг, а тяжелое на каждом шаге опускается по направлению к нужному месту на одну позицию.

### Сортировка выбором (select)

При сортировке массива Arr0, Arr2, ..., ArrN‑1 методом простого выбора среди всех элементов находится элемент с наименьшим значением Arri, и Arr0 и Arri обмениваются значениями. Затем этот процесс повторяется для получаемого подмассива Arr1, Arr2, ..., ArrN‑1, ... Arrj, Arrj+1, ..., ArrN‑1 до тех пор, пока мы не дойдем до подмассива ArrN‑1, содержащего к этому моменту наибольшее значение.

Для метода сортировки простым выбором оценка требуемого числа сравнений – N(N‑1)/2. Порядок требуемого числа пересылок, которые требуются для выбора минимального элемента, в худшем случае составляет O(N2). Однако порядок среднего числа пересылок есть O(N\*Lg(N)), что в ряде случаев делает этот метод предпочтительным.

## Анализ эффективности.

При помощи программы замеряем количество сравнений и перестановок для каждого метода сортировки, которые мы применяем к упорядоченному массиву, обратно упорядоченному массиву, и исследуем количество перестановок со случайными массивами. Конечные данные записываем в отдельный файл.

Исходя из полученный данных можем сделать вывод, что наиболее эффективным методом сортировки является сортировка выбором (select).

# Курсовая работа — часть 2: Матричная математика и работа с пакетами.

Сам по себе «чистый» Python пригоден только для несложных вычислений. Ключевая особенность Python — его расширяемость. Это, пожалуй, самый расширяемый язык из получивших широкое распространение. Как следствие этого для Python не только написаны и приспособлены многочисленные библиотеки алгоритмов на C и Fortran, но и имеются возможности использования других программных средств и математических пакетов, в частности, R и SciLab, а также графопостроителей, например, Gnuplot и PLPlot.

Одной из самых значимых библиотек для превращения Python в математический пакет являются Numpy и Matplotlib.

Numpy — это библиотека (модуль, в действительности, набор модулей) языка Python, добавляющая поддержку больших многомерных массивов и матриц, вместе с большим набором высокоуровневых (и очень быстрых) математических функций для операций с этими массивами.

Matplotlib — это библиотека (модуль, в действительности, набор модулей) на языке программирования Python для визуализации данных двумерной (2D) графикой (3D графика также поддерживается). Получаемые изображения могут быть использованы в качестве иллюстраций в публикациях.

Кроме Numpy и Matplotlib популярен модуль scipy для специализированных математических вычислений (поиск минимума и корней функции многих переменных, аппроксимация сплайнами, вейвлет-преобразования), sympy для символьных вычислений (аналитическое взятие интегралов, упрощение математических выражений), ffnet для построения искусственных нейронных сетей, pyopencl/pycuda для вычисления на видеокартах и некоторые другие. Возможности numpy и scipy покрывают практически все потребности в математических алгоритмах. Но далее речь пойдет о Numpy.

# Курсовая работа — часть 3: Решение слау

Метод Гаусса при решении СЛАУ позволяет ответить на вопросы о совместности или несовместности, определенности или неопределенности системы линейных уравнений, а также отыскать все решения совместной системы. В основе метода лежит идея последовательного исключения неизвестных с помощью подстановок, суть которой состоит в приведении данной системы к другой, равносильной ей, но более простой системе. Это приведение одной системы к другой осуществляется путем элементарных преобразований, которые производятся над уравнениями системы или, что удобнее, над строками расширенной матрицы. Элементарному преобразованию системы линейных уравнений соответствует одноименное элементарное преобразование строк ее расширенной матрицы.

# Заключение.

1. Мы ближе познакомились с языком программирования python.
2. Методом анализа выявили наиболее эффективный метод сортировки.
3. Изучили библиотеку numpy и с её помощью решили 31 задачу.
4. При помощи библиотеки numpy решили системы линейных уравнений методом Гаусса.

В итоге можно сделать вывод, что python достаточно прост в понимании, а благодаря библиотекам имеет очень большой круг применения .