Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ»

Кафедра информатики

Отчет по лабораторной работе №5

Вычисление собственных значений и векторов

Выполнил:

студент группы 053506

Ермолович Д.С

Руководитель:

доцент

Анисимов В.Я.

Минск 2022

**Содержание**

[**Цель работы** 3](#_Toc100962554)

[**Теоретические сведения** 3](#_Toc100962555)

[**Метод Якоби** 3](#_Toc100962556)

[**Свойства алгоритма** 5](#_Toc100962557)

[**Метод QR разложения** 5](#_Toc100962558)

[**QR-алгоритм** 6](#_Toc100962559)

[**Хессенберговая форма матрицы** 6](#_Toc100962560)

[**Сдвиги QR-алгоритма** 7](#_Toc100962561)

[**Программная реализация** 8](#_Toc100962562)

[**Тестовые примеры:** 10](#_Toc100962563)

[**Вывод** 12](#_Toc100962564)

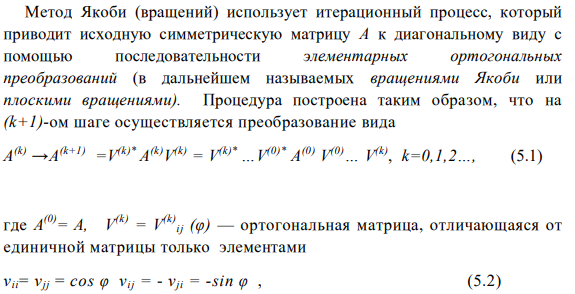
[**Литература** 12](#_Toc100962565)

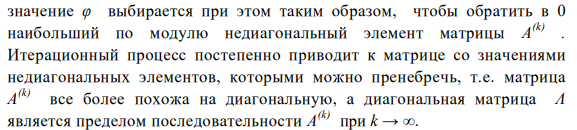
**Цель работы**

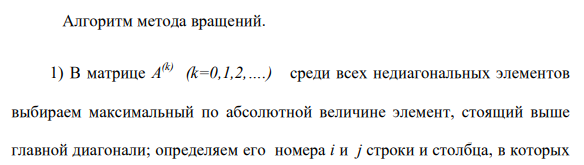
Освоить методы вычисления собственных значений и векторов

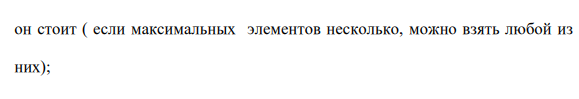
**Теоретические сведения**

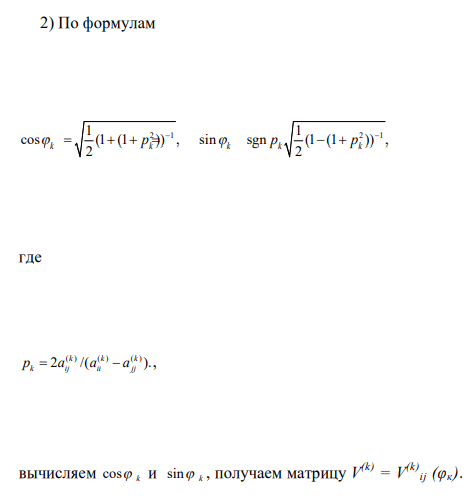
## **Метод Якоби**



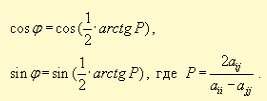




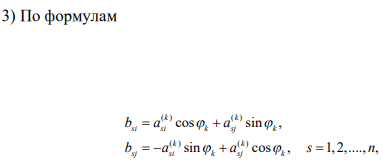




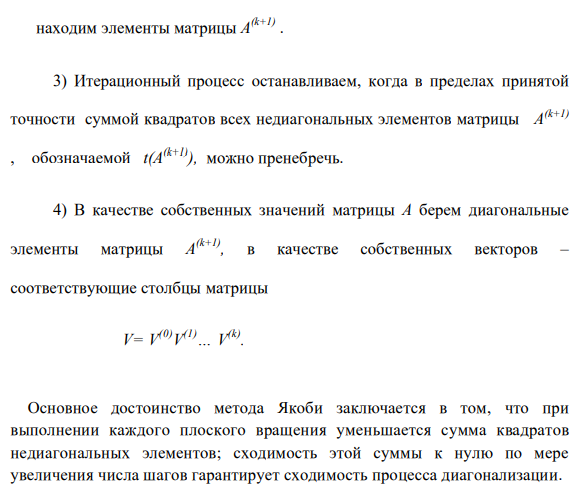
Или



Формулы взяты с сайта[[1]](#footnote-1)



C:\Users\Dima\Downloads\Untitled.png



### **Свойства алгоритма**

Метод Якоби является самым медленным из имеющихся алгоритмов вычисления собственных значений симметричной матрицы. Тем не менее, он способен вычислять малые собственные числа и отвечающие им собственные векторы с гораздо большей точностью, чем другие методы [[1]](https://algowiki-project.org/ru/%D0%9A%D0%BB%D0%B0%D1%81%D1%81%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D0%BC%D0%B5%D1%82%D0%BE%D0%B4_%D0%AF%D0%BA%D0%BE%D0%B1%D0%B8_(%D0%B2%D1%80%D0%B0%D1%89%D0%B5%D0%BD%D0%B8%D0%B9)_%D0%B4%D0%BB%D1%8F_%D1%81%D0%B8%D0%BC%D0%BC%D0%B5%D1%82%D1%80%D0%B8%D1%87%D0%BD%D1%8B%D1%85_%D0%BC%D0%B0%D1%82%D1%80%D0%B8%D1%86_%D1%81_%D0%B2%D1%8B%D0%B1%D0%BE%D1%80%D0%BE%D0%BC_%D0%BF%D0%BE_%D0%B2%D1%81%D0%B5%D0%B9_%D0%BC%D0%B0%D1%82%D1%80%D0%B8%D1%86%D0%B5#cite_note-VVVVVV-1). Кроме того, он не предполагает первоначального приведения матрицы к трехдиагональной форме.

Соотношение последовательной и параллельной сложности в случае неограниченных ресурсов для метода Якоби является линейным.

Вычислительная мощность алгоритма линейна.

Метод Якоби не детерминирован, так как является итерационным алгоритмом с выходом по точности: число итераций зависит от входных данных и порогового значения.

## **Метод QR разложения[[2]](#footnote-2)**

### **QR-алгоритм**

**QR-алгоритм** — это [численный метод](https://ru.wikipedia.org/wiki/%D0%92%D1%8B%D1%87%D0%B8%D1%81%D0%BB%D0%B8%D1%82%D0%B5%D0%BB%D1%8C%D0%BD%D1%8B%D0%B5_%D0%BC%D0%B5%D1%82%D0%BE%D0%B4%D1%8B) в [линейной алгебре](https://ru.wikipedia.org/wiki/%D0%9B%D0%B8%D0%BD%D0%B5%D0%B9%D0%BD%D0%B0%D1%8F_%D0%B0%D0%BB%D0%B3%D0%B5%D0%B1%D1%80%D0%B0), предназначенный для решения полной проблемы собственных значений, то есть отыскания всех собственных чисел и [собственных векторов](https://ru.wikipedia.org/wiki/%D0%A1%D0%BE%D0%B1%D1%81%D1%82%D0%B2%D0%B5%D0%BD%D0%BD%D1%8B%D0%B9_%D0%B2%D0%B5%D0%BA%D1%82%D0%BE%D1%80) [матрицы](https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D1%82%D1%80%D0%B8%D1%86%D0%B0_(%D0%BC%D0%B0%D1%82%D0%B5%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0)).

Пусть *A* — вещественная матрица, для которой мы хотим найти собственные числа и векторы. Положим *A*0=*A*. На *k*-м шаге (начиная с *k* = 0) вычислим [QR-разложение](https://ru.wikipedia.org/wiki/QR-%D1%80%D0%B0%D0%B7%D0%BB%D0%BE%D0%B6%D0%B5%D0%BD%D0%B8%D0%B5) *Ak*=*QkRk*, где *Qk* — [ортогональная матрица](https://ru.wikipedia.org/wiki/%D0%9E%D1%80%D1%82%D0%BE%D0%B3%D0%BE%D0%BD%D0%B0%D0%BB%D1%8C%D0%BD%D0%B0%D1%8F_%D0%BC%D0%B0%D1%82%D1%80%D0%B8%D1%86%D0%B0) (то есть *QkT* = *Qk*−1), а *Rk* — верхняя [треугольная матрица](https://ru.wikipedia.org/wiki/%D0%A2%D1%80%D0%B5%D1%83%D0%B3%D0%BE%D0%BB%D1%8C%D0%BD%D0%B0%D1%8F_%D0%BC%D0%B0%D1%82%D1%80%D0%B8%D1%86%D0%B0). Затем мы определяем *Ak*+1 = *RkQk*.

Заметим, что

C:\Users\Dima\Downloads\Untitled.png

то есть все матрицы *Ak* являются [подобными](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D0%B4%D0%BE%D0%B1%D0%BD%D1%8B%D0%B5_%D0%BC%D0%B0%D1%82%D1%80%D0%B8%D1%86%D1%8B), то есть их собственные значения равны.

Пусть все диагональные [миноры](https://ru.wikipedia.org/wiki/%D0%9C%D0%B8%D0%BD%D0%BE%D1%80_(%D0%BB%D0%B8%D0%BD%D0%B5%D0%B9%D0%BD%D0%B0%D1%8F_%D0%B0%D0%BB%D0%B3%D0%B5%D0%B1%D1%80%D0%B0)) матрицы *A* не [вырождены](https://ru.wikipedia.org/wiki/%D0%92%D1%8B%D1%80%D0%BE%D0%B6%D0%B4%D0%B5%D0%BD%D0%BD%D0%B0%D1%8F_%D0%BC%D0%B0%D1%82%D1%80%D0%B8%D1%86%D0%B0). Тогда [последовательность](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D1%81%D0%BB%D0%B5%D0%B4%D0%BE%D0%B2%D0%B0%D1%82%D0%B5%D0%BB%D1%8C%D0%BD%D0%BE%D1%81%D1%82%D1%8C) матриц *Ak* при {\displaystyle k\to \infty ,} сходится по форме к клеточному правому треугольному виду, соответствующему клеткам с одинаковыми по модулю собственными значениями.[[1]](https://ru.wikipedia.org/wiki/QR-%D0%B0%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC#cite_note-1)

Для того, чтобы получить собственные векторы матрицы, нужно перемножить все матрицы *Qk*.

Алгоритм считается [вычислительно устойчивым](https://ru.wikipedia.org/wiki/%D0%92%D1%8B%D1%87%D0%B8%D1%81%D0%BB%D0%B8%D1%82%D0%B5%D0%BB%D1%8C%D0%BD%D0%B0%D1%8F_%D1%83%D1%81%D1%82%D0%BE%D0%B9%D1%87%D0%B8%D0%B2%D0%BE%D1%81%D1%82%D1%8C), т. к. производится ортогональными преобразованиями подобия.

### **Хессенберговая форма матрицы**

Использование хессенберговой формы матрицы.

Хессенбергова (почти треугольная) форма матрицы интересна в приложении к QR-алгоритму тем, что в случае, если A0=A имеет эту форму, то имеют её и все матрицы Ak. Если теперь посмотреть на методы [QR-разложения плотных неособенных матриц](https://algowiki-project.org/ru/QR-%D1%80%D0%B0%D0%B7%D0%BB%D0%BE%D0%B6%D0%B5%D0%BD%D0%B8%D1%8F_%D0%BF%D0%BB%D0%BE%D1%82%D0%BD%D1%8B%D1%85_%D0%BD%D0%B5%D0%BE%D1%81%D0%BE%D0%B1%D0%B5%D0%BD%D0%BD%D1%8B%D1%85_%D0%BC%D0%B0%D1%82%D1%80%D0%B8%D1%86), то видно, что выполнение разложения Ak=QkRk в случае плотной неособенной Akтребует (в последовательном варианте) O(n3) операций, как и последующий процесс вычисления Ak+1=RkQk. В случае же, если матрицы хессенберговы, то как [QR-разложения плотных хессенберговых матриц](https://algowiki-project.org/ru/%D0%9C%D0%B5%D1%82%D0%BE%D0%B4%D1%8B_QR-%D1%80%D0%B0%D0%B7%D0%BB%D0%BE%D0%B6%D0%B5%D0%BD%D0%B8%D1%8F_%D0%BF%D0%BB%D0%BE%D1%82%D0%BD%D1%8B%D1%85_%D1%85%D0%B5%D1%81%D1%81%D0%B5%D0%BD%D0%B1%D0%B5%D1%80%D0%B3%D0%BE%D0%B2%D1%8B%D1%85_%D0%BC%D0%B0%D1%82%D1%80%D0%B8%D1%86), так и вычисления Ak+1=RkQk потребуют уже по O(n2) операций. Это позволяет экономить если не время (критические пути графа алгоритма линейны для лучших методов QR-разложения и для плотной матрицы, и для хессенберговой), то довольно большие вычислительные ресурсы.

Поэтому естественным приёмом ускорения итераций QR-алгоритма является начальное [унитарно подобное приведение матрицы к хессенбергову виду](https://algowiki-project.org/ru/%D0%A0%D0%B0%D0%B7%D0%BB%D0%BE%D0%B6%D0%B5%D0%BD%D0%B8%D1%8F,_%D1%81%D0%BE%D0%B4%D0%B5%D1%80%D0%B6%D0%B0%D1%89%D0%B8%D0%B5_%D1%85%D0%B5%D1%81%D1%81%D0%B5%D0%BD%D0%B1%D0%B5%D1%80%D0%B3%D0%BE%D0%B2%D1%83_%D0%BC%D0%B0%D1%82%D1%80%D0%B8%D1%86%D1%83,_%D1%83%D0%BD%D0%B8%D1%82%D0%B0%D1%80%D0%BD%D0%BE_%D0%BF%D0%BE%D0%B4%D0%BE%D0%B1%D0%BD%D1%83%D1%8E_%D0%B8%D1%81%D1%85%D0%BE%D0%B4%D0%BD%D0%BE%D0%B9). После этого все процедуры QR-алгоритма следует проводить уже с матрицами хессенберговой формы.

В случае, если исходная матрица эрмитова (в вещественном случае - симметричная), то унитарно подобное приведение к хессенберговому виду становится [приведением к трёхдиагональному симметричному виду](https://algowiki-project.org/ru/%D0%A0%D0%B0%D0%B7%D0%BB%D0%BE%D0%B6%D0%B5%D0%BD%D0%B8%D1%8F,_%D1%81%D0%BE%D0%B4%D0%B5%D1%80%D0%B6%D0%B0%D1%89%D0%B8%D0%B5_%D1%82%D1%80%D1%91%D1%85%D0%B4%D0%B8%D0%B0%D0%B3%D0%BE%D0%BD%D0%B0%D0%BB%D1%8C%D0%BD%D1%83%D1%8E_%D0%BC%D0%B0%D1%82%D1%80%D0%B8%D1%86%D1%83,_%D1%83%D0%BD%D0%B8%D1%82%D0%B0%D1%80%D0%BD%D0%BE_%D0%BF%D0%BE%D0%B4%D0%BE%D0%B1%D0%BD%D1%83%D1%8E_%D0%B8%D1%81%D1%85%D0%BE%D0%B4%D0%BD%D0%BE%D0%B9). В этом случае QR-алгоритм становится ещё быстрее, благодаря существенно большему количеству нулей в структурах образующихся матриц..

### **Сдвиги QR-алгоритма**

QR-алгоритм со сдвигами позволяет сократить количество итераций, необходимых для сходимости[[7]](https://algowiki-project.org/ru/QR-%D0%B0%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC#cite_note-bahvalov-7). Пусть у нас есть матрица Ak, тогда процесс перехода к матрице Ak+1 выглядит следующим образом:

* На каждом шаге подбирается число νk и ищется следующее QR-разложение: Ak−νkE=QkRk.
* Вычисляется матрица Ak+1: Ak+1=RkQk+νkE.

При этом сохраняется свойство подобия матриц Ak и Ak+1:

Ak+1=RkQk+νkE=QTkQkRkQk+νkE=QTk(Ak−νkE)Qk+νkE=QTkAkQk−QTk(νkE)Qk+νkE=QTkAkQk−νkE+νkE=QTkAkQk.

Подбор параметра νk осуществляется в зависимости от элементов матрицы Ak. При алгоритме подбора параметров, известном, как *правило Фрэнсиса*, среднее количество итераций для большинства матриц получается порядка 2 на одно собственное значение. Существуют, однако, матрицы, для которых правило Фрэнсиса не даёт сходимости, поэтому если по истечении 10 итераций с обычным сдвигом не получилось сходимости, применяют так называемые "особые сдвиги". Для симметрического же случая используется стратегия Уилкинсона[[6]](https://algowiki-project.org/ru/QR-%D0%B0%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC" \l "cite_note-Dem-6).

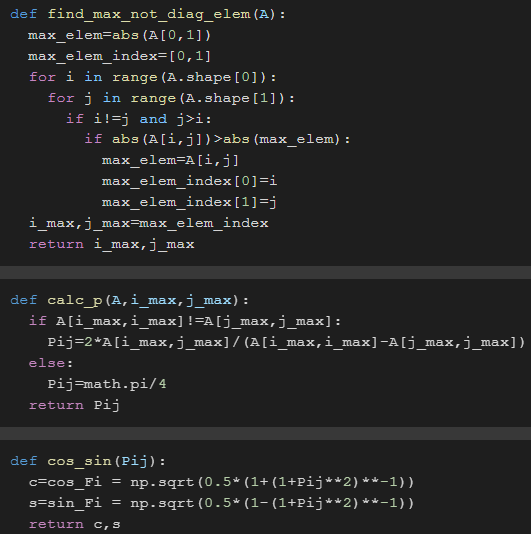
# **Программная реализация**

Функции:

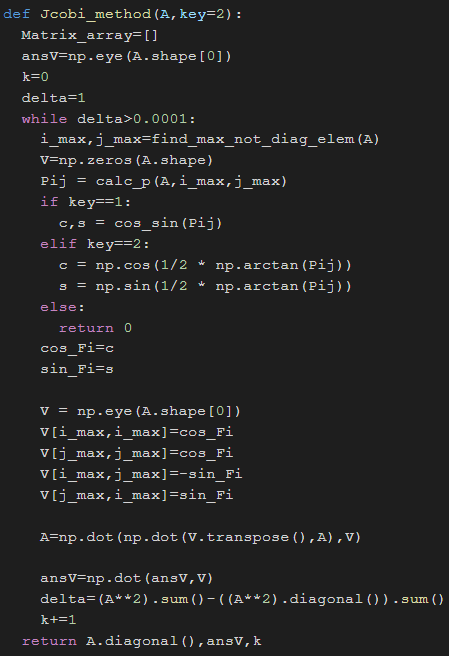
find\_max\_not\_diag\_elem – вычисляет наибольший по модулю не диагональный элемент.

calc\_p – вычисляет p

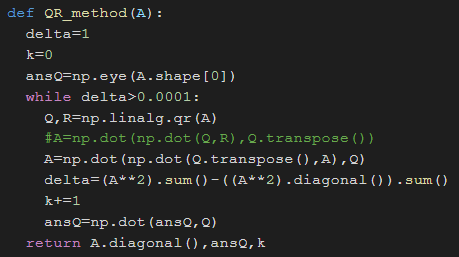
cos\_sin – вычисляет cos sin



Jcobi\_method – Якоби метод

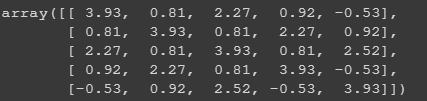


QR - метод

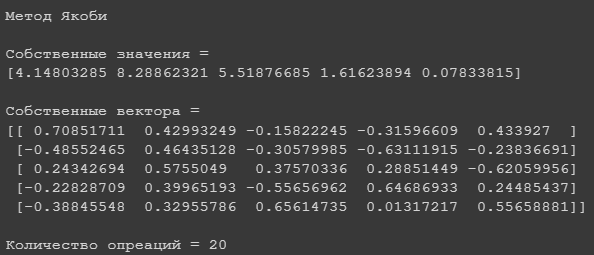


# **Тестовые примеры:**

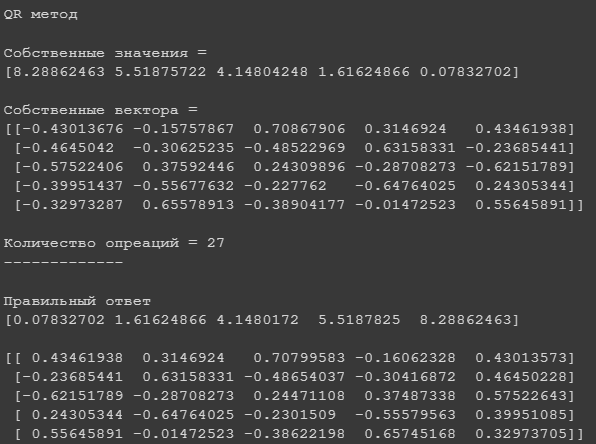
Вариант 8



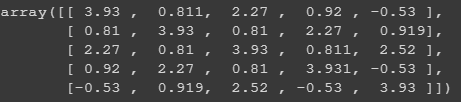
Метод Якоби



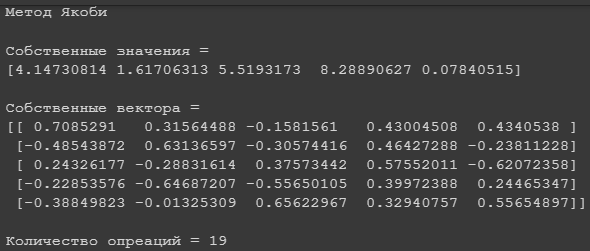
QR алгоритм

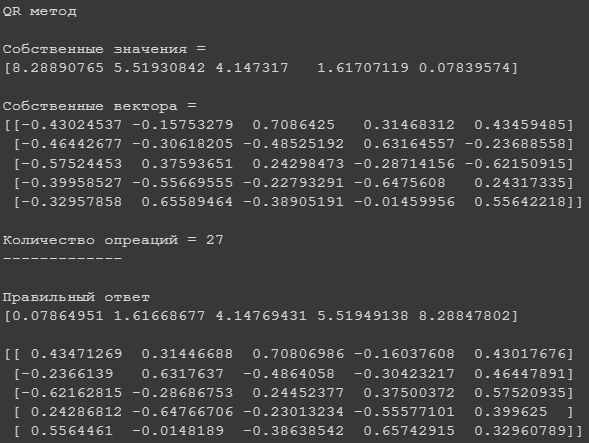


Изменим коэффициенты

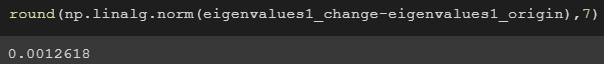


Метод Якоби

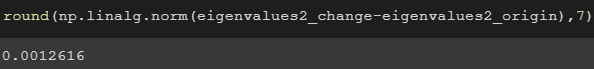




Погрешность собственных значений на методе вращений Якоби



Погрешность собственных значений на методе вращений Якоби



# **Вывод**

В работе были исследованы методы нахождения собственных значений и векторов двумя методами: методом вращений Якоби и QR метод. После проделанной работы я могу заключить, что QR метод сходится медленнее чем метод Якоби, что и требовалось доказать. Однако существуют вариации QR метода, которые приближают его к методу Якоби. Все поставленные цели выполнены, лабораторная работа успешно завершена.

# **Литература**

<https://old.math.tsu.ru/EEResources/cm/text/6_2_3.htm> (Метод вращений)

Классический метод Якоби (вращений) для симметричных матриц с выбором по всей матрице

<https://algowiki-project.org/ru/%D0%9A%D0%BB%D0%B0%D1%81%D1%81%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D0%BC%D0%B5%D1%82%D0%BE%D0%B4_%D0%AF%D0%BA%D0%BE%D0%B1%D0%B8_(%D0%B2%D1%80%D0%B0%D1%89%D0%B5%D0%BD%D0%B8%D0%B9)_%D0%B4%D0%BB%D1%8F_%D1%81%D0%B8%D0%BC%D0%BC%D0%B5%D1%82%D1%80%D0%B8%D1%87%D0%BD%D1%8B%D1%85_%D0%BC%D0%B0%D1%82%D1%80%D0%B8%D1%86_%D1%81_%D0%B2%D1%8B%D0%B1%D0%BE%D1%80%D0%BE%D0%BC_%D0%BF%D0%BE_%D0%B2%D1%81%D0%B5%D0%B9_%D0%BC%D0%B0%D1%82%D1%80%D0%B8%D1%86%D0%B5>

QR – алгоритм

<https://algowiki-project.org/ru/QR-%D0%B0%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC#.D0.9C.D0.B0.D1.82.D0.B5.D0.BC.D0.B0.D1.82.D0.B8.D1.87.D0.B5.D1.81.D0.BA.D0.BE.D0.B5_.D0.BE.D0.BF.D0.B8.D1.81.D0.B0.D0.BD.D0.B8.D0.B5>

<https://ru.wikipedia.org/wiki/QR-%D0%B0%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC>

1. https://old.math.tsu.ru/EEResources/cm/text/6\_2\_3.htm [↑](#footnote-ref-1)
2. QR-алгоритм [↑](#footnote-ref-2)