

ТРЕБОВАНИЯ К ПРОГРАММАМ

1. Программа должна получать все параметры в качестве аргументов командной строки. Аргументы командной строки:

- 1) n – размерность матрицы,
- 2) m – количество выводимых значений в матрице,
- 3) ϵ – точность нахождения собственных значений матрицы,
- 4) k – задает номер формулы для инициализации матрицы, должен быть равен 0 при вводе матрицы из файла
- 5) `filename` – имя файла, откуда надо прочитать матрицу. Этот аргумент **отсутствует**, если $k \neq 0$.

Например, запуск

```
./a.out 4 4 1e-15 0 a.txt
```

означает, что матрицу 4×4 надо прочитать из файла `a.txt`, выводить не более 4-х строк и столбцов матрицы, и решить задачу с точностью 10^{-15} , а запуск

```
./a.out 2000 6 1e-14 1
```

означает, что матрицу 2000×2000 надо инициализировать по формуле номер 1, выводить не более 6-ти строк и столбцов матрицы, и решить задачу с точностью 10^{-14} .

2. Ввод матрицы должен быть оформлен в виде подпрограммы, находящейся в отдельном файле.
3. Ввод матрицы из файла. В указанном файле находится матрица в формате:

$$\begin{array}{ccc} a_{1,1} & \dots & a_{1,n} \\ a_{2,1} & \dots & a_{2,n} \\ \dots & \dots & \dots \\ a_{n,1} & \dots & a_{n,n} \end{array}$$

где n - указанный размер матрицы, $A = (a_{i,j})$ - матрица. Программа должна выводить сообщение об ошибке, если указанный файл не может быть прочитан, содержит меньшее количество данных или данные неверного формата.

4. Ввод матрицы и правой части по формуле. Элемент $a_{i,j}$ матрицы A полагается равным

$$a_{i,j} = f(k, n, i, j), \quad i, j = 1, \dots, n,$$

где $f(k, n, i, j)$ - функция, которая возвращает значение (i, j) -го элемента $n \times n$ матрицы по формуле номер k (аргумент командной строки). Функция $f(k, n, i, j)$ должна быть оформлена в виде отдельной подпрограммы.

$$f(k, n, i, j) = \begin{cases} n - \max\{i, j\} + 1 & \text{при } k = 1 \\ 2 \text{ при } i = j, \quad -1 \text{ при } |i - j| = 1, \quad 0 \text{ иначе} & \text{при } k = 2 \\ 1 \text{ при } i = j < n, \quad i \text{ при } j = n, \quad j \text{ при } i = n, \quad 0 \text{ иначе} & \text{при } k = 3 \\ \frac{1}{i + j - 1} & \text{при } k = 4 \end{cases}$$

5. Решение системы должно быть оформлено в виде подпрограммы, находящейся в отдельном файле и получающей в качестве аргументов

- 1) размерность n матрицы A ,
- 2) матрицу A ,
- 3) вектор x , в который будут помещены найденные собственные значения,
- 4) точность ε ,
- 5) дополнительные вектора, если алгоритму требуется дополнительная память.

Получать в этой подпрограмме дополнительную информацию извне через глобальные переменные, включаемые файлы и т.п. запрещается.

6. Программа должна содержать подпрограмму вывода на экран прямоугольной матрицы $l \times n$ матрицы. Эта подпрограмма используется для вывода исходной $n \times n$ матрицы после ее инициализации, а также для вывода на экран найденных s собственных значений системы ($1 \times s$ матрицы). Подпрограмма выводит на экран не более, чем m строк и столбцов $l \times n$ матрицы, где m – параметр этой подпрограммы (аргумент командной строки). Каждая строка матрицы должна печататься на новой строке, каждый элемент матрицы выводится в строке по формату " %10.3e" (один пробел между элементами и экспоненциальный формат %10.3e).

7. Если требуется найти все n собственных значений или найдены все n собственных значений на указанном отрезке, то программа должна содержать подпрограмму вычисления норм невязок, т.е.

- невязку в первом инварианте: модуль разности следа исходной матрицы и суммы собственных значений,
- невязку во втором инварианте: модуль разности длины исходной матрицы как вектора размера n^2 и суммы квадратов собственных значений,

и выводить эти нормы невязок на экран (в экспоненциальном формате).

8. Программа должна выводить на экран время, затраченное на решение системы (обращение матрицы, если стоит задача обратить матрицу). Допустимо представлять время в сотых долях секунды, не преобразовывая в обычный формат чч.мм.сс:тт.

9. Суммарный объем оперативной памяти, требуемой программе, не должен превышать $n^2 + O(n)$.

10. Время работы программы не должно превышать $O(n^3)$.

МЕТОДЫ НАХОЖДЕНИЯ СОБСТВЕННЫХ ЗНАЧЕНИЙ

1. Степенной метод нахождения максимального по модулю собственного значения и соответствующего собственного вектора.
2. Метод вращений Якоби с выбором в качестве обнуляемого элемента максимального по модулю среди внедиагональных элементов.
3. Метод вращений Якоби с циклическим выбором обнуляемого элемента.

4. Метод вращений Якоби с выбором в качестве обнуляемого элемента максимального по модулю среди внедиагональных элементов столбца с наибольшей суммой квадратов внедиагональных элементов (т.н. оптимальный выбор).
5. Метод бисекции нахождения k -го по величине собственного значения симметричной матрицы с приведением ее к трехдиагональному виду методом вращений и вычислением числа перемен знака в последовательности главных миноров с помощью LU-разложения.
6. Метод бисекции нахождения k -го по величине собственного значения симметричной матрицы с приведением ее к трехдиагональному виду методом вращений и вычислением числа перемен знака в последовательности главных миноров с помощью рекуррентных формул.
7. Метод бисекции нахождения k -го по величине собственного значения симметричной матрицы с приведением ее к трехдиагональному виду методом отражений и вычислением числа перемен знака в последовательности главных миноров с помощью LU-разложения.
8. Метод бисекции нахождения k -го по величине собственного значения симметричной матрицы с приведением ее к трехдиагональному виду методом отражений и вычислением числа перемен знака в последовательности главных миноров с помощью рекуррентных формул.
9. Метод бисекции нахождения всех собственных значений симметричной матрицы на заданном интервале с приведением ее к трехдиагональному виду методом вращений и вычислением числа перемен знака в последовательности главных миноров с помощью LU-разложения.
10. Метод бисекции нахождения всех собственных значений симметричной матрицы на заданном интервале с приведением ее к трехдиагональному виду методом вращений и вычислением числа перемен знака в последовательности главных миноров с помощью рекуррентных формул.
11. Метод бисекции нахождения всех собственных значений симметричной матрицы на заданном интервале с приведением ее к трехдиагональному виду методом отражений и вычислением числа перемен знака в последовательности главных миноров с помощью LU-разложения.
12. Метод бисекции нахождения всех собственных значений симметричной матрицы на заданном интервале с приведением ее к трехдиагональному виду методом отражений и вычислением числа перемен знака в последовательности главных миноров с помощью рекуррентных формул.
13. Метод бисекции нахождения всех собственных значений симметричной матрицы с приведением ее к трехдиагональному виду методом вращений и вычислением числа перемен знака в последовательности главных миноров с помощью LU-разложения.
14. Метод бисекции нахождения всех собственных значений симметричной матрицы с приведением ее к трехдиагональному виду методом вращений и вычислением числа перемен знака в последовательности главных миноров с помощью рекуррентных формул.
15. Метод бисекции нахождения всех собственных значений симметричной матрицы с приведением ее к трехдиагональному виду методом отражений и вычислением числа перемен знака в последовательности главных миноров с помощью LU-разложения.

16. Метод бисекции нахождения всех собственных значений симметричной матрицы с приведением ее к трехдиагональному виду методом отражений и вычислением числа перемен знака в последовательности главных миноров с помощью рекуррентных формул.
17. LR - алгоритм нахождения всех собственных значений матрицы с приведением ее к почти треугольному виду методом вращений.
18. LR - алгоритм нахождения всех собственных значений матрицы с приведением ее к почти треугольному виду методом отражений.
19. Метод Холецкого нахождения собственных значений симметричной положительно определенной матрицы с приведением ее к трехдиагональному виду методом вращений.
20. Метод Холецкого нахождения собственных значений симметричной положительно определенной матрицы с приведением ее к трехдиагональному виду методом отражений.
21. QR - алгоритм нахождения всех собственных значений матрицы с приведением ее к почти треугольному виду методом вращений и нахождением QR-разложения на каждом шаге методом вращений.
22. QR - алгоритм нахождения всех собственных значений матрицы с приведением ее к почти треугольному виду методом вращений и нахождением QR-разложения на каждом шаге методом отражений.
23. QR - алгоритм нахождения всех собственных значений матрицы с приведением ее к почти треугольному виду методом отражений и нахождением QR-разложения на каждом шаге методом вращений.
24. QR - алгоритм нахождения всех собственных значений матрицы с приведением ее к почти треугольному виду методом отражений и нахождением QR-разложения на каждом шаге методом отражений.
25. QR - алгоритм нахождения всех собственных значений симметричной матрицы с приведением ее к трехдиагональному виду методом вращений и нахождением QR-разложения на каждом шаге методом вращений.
26. QR - алгоритм нахождения всех собственных значений симметричной матрицы с приведением ее к трехдиагональному виду методом вращений и нахождением QR-разложения на каждом шаге методом отражений.
27. QR - алгоритм нахождения всех собственных значений симметричной матрицы с приведением ее к трехдиагональному виду методом отражений и нахождением QR-разложения на каждом шаге методом вращений.
28. QR - алгоритм нахождения всех собственных значений симметричной матрицы с приведением ее к трехдиагональному виду методом отражений и нахождением QR-разложения на каждом шаге методом отражений.