ТРЕБОВАНИЯ К ПРОГРАММАМ
(MPI версия)

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

   1) n – размерность матрицы,
   2) m – размер блока,
   3) r – количество выводимых значений в матрице,
   4) s – 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 $s! = 0$.

   Например, запуск

   ```
   mpirun -np 2 ./a.out 4 3 4 0 a.txt
   ```

   означает, что матрицу `4x4` надо прочитать из файла `a.txt`, использовать блочный алгоритм с размером блока 3 и 2 MPI процесса, и выводить не более 4-х строк и столбцов матрицы, а запуск

   ```
   mpirun -np 8 ./a.out 2000 90 10 1
   ```

   означает, что матрицу `2000x2000` надо инициализировать по формуле номер 1, использовать блочный алгоритм с размером блока 90 и 8 MPI процессов, и выводить не более 10-ти строк и столбцов матрицы.

2. В задачах, где требуется правая часть b, этот вектор строится после инициализации матрицы $A = (a_{i,j})_{i,j=1,\ldots,n}$ по формуле:

$$b = (b_i)_{i=1,\ldots,n}, \quad b_i = \sum_{k=0}^{(n+1)/2} a_{i,2k+1}$$

3. Ввод матрицы должен быть оформлен в виде подпрограммы, находящейся в отдельном файле.

4. Ввод матрицы из файла. В указанном файле находится матрица в формате:

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

   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.

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

$$a_{i,j} = f(s,n,i,j), \quad i,j = 1,\ldots,n,$$

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

$$f(s,n,i,j)=\begin{cases} n-\max\{i,j\}+1 & \text{при} \quad s=1 \\ \max\{i,j\} & \text{при} \quad s=2 \\ |i-j| & \text{при} \quad s=3 \\ \dfrac{1}{i+j-1} & \text{при} \quad s=4 \end{cases}$$

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

   (a) размерность n матрицы A,

   (b) размер блока m,

   (c) номер процесса k, где работает этот "экземпляр" функции,

   (d) число процессов p,

   (e) коммуникатор `comm` (типа `MPI_Comm`),

   (f) матрицу A,

   (g) правую часть b (если стоит задача решить линейную систему)

   (h) вектор x, в который будет помещено решение системы, если стоит задача решить линейную систему, или матрицу X, в которую будет помещена обратная матрица, если стоит задача обратить матрицу,

   (i) дополнительные вектора, если алгоритму требуется дополнительная память.

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

7. Функция, реализующая задачу, возвращает ненулевое значение, если алгоритм решения неприменим к поданной на вход матрице A.

8. Функция, реализующая задачу, **не должна выделять или использовать дополнительную память**.

9. **Сложность работы функции**, реализующая задачу, не должна превышать $O(n^3)$.

10. Суммарный объем оперативной памяти, требуемой программе, не должен превышать:

    - при вычислении решения системы: $n^2/p+O(n)$ (на один процесс),
    - при вычислении обратной матрицы: $2n^2/p+O(n)$ (на один процесс),

    где p – число процессов. Для выполнения этого требования после завершения алгоритма решения (нахождения обратной матрицы) вызывается подпрограмма инициализации матрицы (из файла или по формуле) и вычисления вектора b (в задачах решения линейной системы).

11. 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 решения системы ($1\times n$ матрицы) или обратной $n\times n$ матрицы, если стоит задача обратить матрицу. Подпрограмма выводит на экран не более, чем r строк и столбцов $l\times n$ матрицы, где r – 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`).

12. Результатами работы программы являются 3 элемента:

   - Собственно вектор решения x (в задачах нахождения решения линейной системы) или обратная матрица A^{-1} (в задачах нахождения обратной матрицы).
   - Два вещественных числа r_1 и r_2, вычисляемых после вызова функции, реализующей задачу:
     - **В задачах нахождения решения линейной системы**

$$r_1 = \|Ax-b\|_1/\|b\|_1, \quad r_2 = \sum_{i=1}^{n} |x_i - (i \, mod \, 2)|, \quad \text{где } \|y\|_1 = \sum_{i=1}^{n} |y_i|, \ y = (y_i)_{i=1,\dots,n}$$

     - **В задачах нахождения обратной матрицы**

$$r_1 = \begin{cases} \|AA^{-1} - E\|_1, & N \le 11000, \\ 0, & N > 11000 \end{cases} \qquad r_2 = \begin{cases} \|A^{-1}A - E\|_1, & N \le 11000, \\ 0, & N > 11000 \end{cases}$$

     где

$$\|Y\|_1 = \max_{1 \le j \le n} \sum_{i=1}^{n} |y_{ij}|, \ Y = (y_{ij})_{i,j=1,\dots,n}$$

   Вычисление r_1 и r_2 должно быть оформлено в виде подпрограммы, вызываемой из функции `main`. Эта подпрограмма **не должна выделять или использовать дополнительную память**.

13. Вывод результата работы функции в функции `main` должен производиться по формату:

   - Непосредственно вывод вектора решения x или обратной матрицы A^{-1}:
     - в задачах нахождения решения линейной системы вывод вектора решения x производится вызовом подпрограммы печати матрицы (см. пункт 11) размера $1 \times n$ (т.е. в строку и **по указанному там формату**)
     - в задачах нахождения обратной матрицы вывод обратной матрицы A^{-1} производится вызовом подпрограммы печати матрицы (см. пункт 11) размера $n \times n$ (**по указанному там формату**)

     Вывод не производится, если алгоритм решения неприменим к поданной на вход матрице A.
   - Отчет о результате и времени работы:

```
printf (
"%s : Task = %d Res1 = %e Res2 = %e T1 = %.2f T2 = %.2f S = %d N = %d M = %d P = %d\n",
argv[0], task, r1, r2, t1, t2, s, n, m, p);
```

     где

     - `argv[0]` – первый аргумент командной строки (имя образа программы),
     - `task` – номер задачи,
     - `r1` $= r_1$ – вычисленное значение $r1$ (см. пункт 12), выводится $r_1 = -1$, если алгоритм решения неприменим к поданной на вход матрице A,
     - `r2` $= r_2$ – вычисленное значение $r2$ (см. пункт 12), выводится $r_2 = -1$, если алгоритм решения неприменим к поданной на вход матрице A,
     - `t1` – время работы функции, реализующей решение этой задачи, в секундах (с точностью до сотых долей),

- `t2` – время работы функции, вычисляющей невязки решения (см. пункт 12), в секундах (с точностью до сотых долей), выводится $\texttt{t2} = 0$, если алгоритм решения неприменим к поданной на вход матрице A,
- `s`, `n`, `m`, `p` – аргументы командной строки.

Вывод должен производиться в точности в таком формате, чтобы можно было автоматизировать обработку запуска многих тестов. **Вывод отчета о результате и времени работы должен производится всегда**, даже если алгоритм решения неприменим к поданной на вход матрице A.

МЕТОДЫ РЕШЕНИЯ СИСТЕМ ЛИНЕЙНЫХ УРАВНЕНИЙ

1. Метод Гаусса решения линейной системы.
2. Метод Гаусса нахождения обратной матрицы.
3. LU-разложение для решения линейной системы.
4. LU-разложение для нахождения обратной матрицы.
5. Метод Холецкого решения линейной системы с симметричной матрицей.
6. Метод Холецкого нахождения обратной матрицы для симметричной матрицы.
7. Метод Жордана решения линейной системы.
8. Метод Жордана нахождения обратной матрицы.
9. Метод Гаусса решения линейной системы с выбором главного элемента по столбцу.
10. Метод Гаусса решения линейной системы с выбором главного элемента по строке.
11. Метод Гаусса решения линейной системы с выбором главного элемента по всей матрице.
12. Метод Гаусса нахождения обратной матрицы с выбором главного элемента по столбцу.
13. Метод Гаусса нахождения обратной матрицы с выбором главного элемента по строке.
14. Метод Гаусса нахождения обратной матрицы с выбором главного элемента по всей матрице.
15. Метод Жордана решения линейной системы с выбором главного элемента по столбцу.
16. Метод Жордана решения линейной системы с выбором главного элемента по строке.
17. Метод Жордана решения линейной системы с выбором главного элемента по всей матрице.
18. Метод Жордана нахождения обратной матрицы с выбором главного элемента по столбцу.
19. Метод Жордана нахождения обратной матрицы с выбором главного элемента по строке.
20. Метод Жордана нахождения обратной матрицы с выбором главного элемента по всей матрице.
21. Метод вращений решения линейной системы.
22. Метод вращений нахождения обратной матрицы.
23. Метод отражений решения линейной системы.
24. Метод отражений нахождения обратной матрицы.