## ТРЕБОВАНИЯ К ПРОГРАММАМ

1. Программа должна получать **все начальные параметры в качестве аргументов командной строки**. Программа имеет 10 обязательных аргументов:

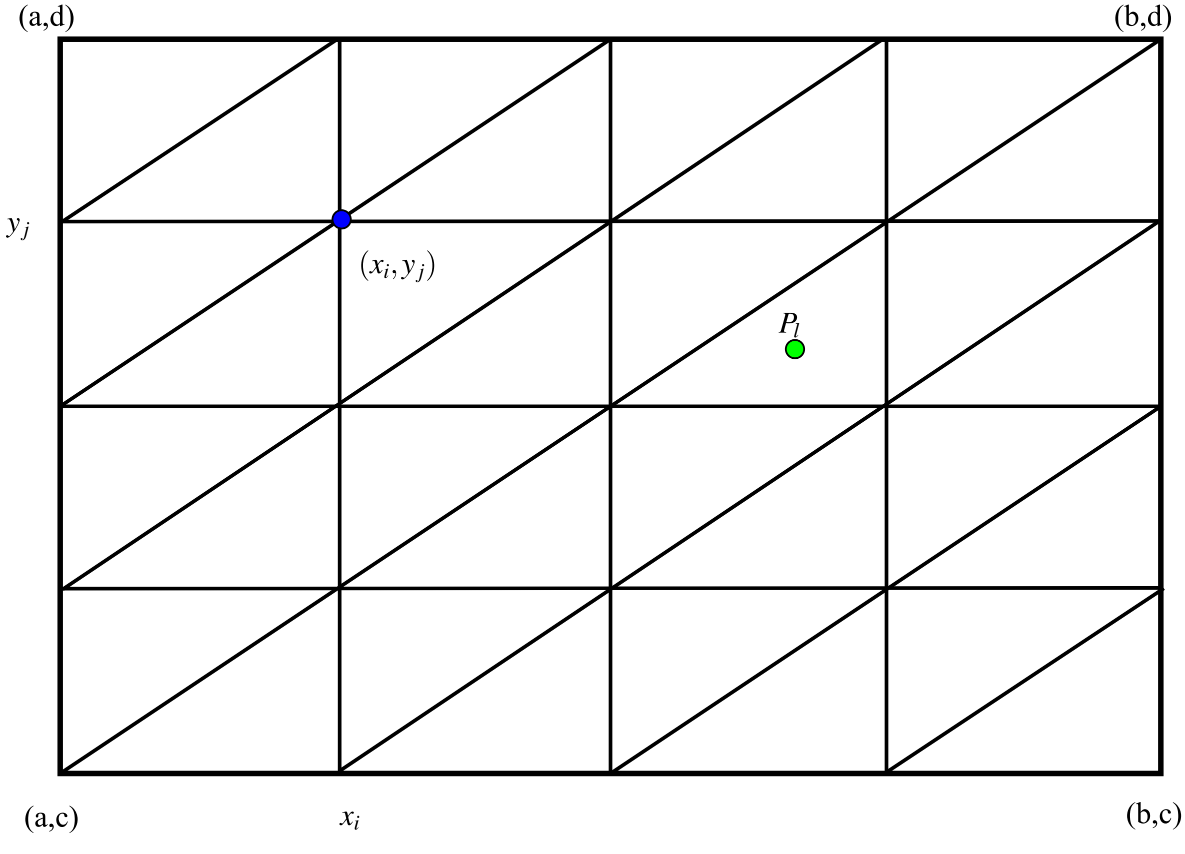
   (1–4) $a$, $b$, $c$, $d$ – спецификация области $[a,b] \times [c,d]$ (тип double),
   (5, 6) $n_x$, $n_y$ – начальное значение для числа точек интерполяции по осям $X$ и $Y$ (тип int),
   (7) $k$ – начальное значение номера приближаемой функции (тип int),
   (8) $\varepsilon$ – точность решения системы линейных уравнений (тип double),
   (9) $m_i$ – максимальное число итераций для решения системы линейных уравнений (тип int),
   (10) $p$ – число вычислительных потоков (тип int).

2. В программе должны быть реализованы подпрограммы для задания следующих приближаемых функций $f(x)$ по аналитически заданной формуле в зависимости от параметра $k$:

   (1) для $k=0$ $f(x,y)=1$
   (2) для $k=1$ $f(x,y)=x$
   (3) для $k=2$ $f(x,y)=y$
   (4) для $k=3$ $f(x,y)=x+y$
   (5) для $k=4$ $f(x,y)=\sqrt{x^2+y^2}$
   (6) для $k=5$ $f(x,y)=x^2+y^2$
   (7) для $k=6$ $f(x,y)=e^{x^2-y^2}$
   (8) для $k=7$ $f(x,y)=1/(25(x^2+y^2)+1)$

3. Решается задача приближения функции $f(x,y)$ в прямоугольной области $[a,b]\times[c,d]$ на сетке из точек $(x_i,y_j)$,

$$x_i = a+ih_x, \quad h_x=(b-a)/n_x, \quad y_j=c+jh_y, \quad h_y=(d-c)/n_y, \quad i=0,\dots,n_x, \quad j=0,\dots,n_y$$



Приближение осуществляется непрерывными линейными на каждом треугольнике функциями методом наименьших квадратов. Ошибка приближения вычисляется в точках $P_l$ – центрах тяжести треугольников $l$, $l = 1, \ldots, 2 \cdot n_x \cdot n_y$.

4. В отдельных файлах должны быть оформлены следующие параллельные (для $p$ потоков) подпрограммы:
   - Вычисление структуры MSR матрицы
   - Вычисление матрицы Грама базиса из функций Куранта
   - Вычисление правой части системы для заданной функции $f$
   - Указанный в задании итерационный метод решения системы уравнений с разреженной матрицей
   - Построение указанного в задании предобуславливателя для разреженной матрицы и решение линейной системы с матрицей предоубуславливателя
   - Вычисление значения приближающей функции $P_f(x,y)$ в точке $(x,y)$ по полученным в результате решения линейной системы коэффициентам разложения по базису Куранта (последовательная, вызывается в каждом из потоков)
   - Вычисление приближения к $C$ норме погрешности
   $$r_1 = \max_{l=1,\ldots,2\cdot n_x\cdot n_y} |f(P_l) - P_f(P_l)| \tag{1}$$
   где $P_l$ – центр тяжести треугольника $l$, $l = 1, \ldots, 2 \cdot n_x \cdot n_y$
   - Вычисление приближения к $L_1$ норме погрешности
   $$r_2 = \sum_{l=1}^{2\cdot n_x\cdot n_y} |f(P_l) - P_f(P_l)| \cdot h_x \cdot h_y / 2 \tag{2}$$
   - Вычисление приближения к $C$ норме отличия от решения задачи линейной интерполяции
   $$r_3 = \max_{i=0,\ldots,n_x,\ j=0,\ldots n_y} |f(x_i,y_j) - P_f(x_i,y_j)| \tag{3}$$
   - Вычисление приближения к $L_1$ норме отличия от решения задачи линейной интерполяции
   $$r_4 = \sum_{i=0}^{n_x}\sum_{j=0}^{n_y} |f(x_i,y_j) - P_f(x_i,y_j)| \cdot h_x \cdot h_y \tag{4}$$
5. Вывод результата работы функции в функции `main` должен производиться по формату:

```
printf (
"%s : Task = %d R1 = %e R2 = %e R3 = %e R4 = %e T1 = %.2f T2 = %.2f\
 It = %d E = %e K = %d Nx = %d Ny = %d P = %d\n",
argv[0], task, r1, r2, r3, r4, t1, t2, it, eps, k, nx, ny, p);
```

где

- `argv[0]` – первый аргумент командной строки (имя образа программы),
- `task` – номер задачи,
- `r1` $= r_1$ – вычисленное значение $r_1$ (см. (1)),
- `r2` $= r_2$ – вычисленное значение $r_2$ (см. (2)),

- `r3` $= r_3$ – вычисленное значение $r_3$ (см. (3)),
- `r4` $= r_4$ – вычисленное значение $r_4$ (см. (4)),
- `t1` – время работы функции, реализующей вычисление коэффициентов приближающей функции $P_f$ (т.е. время на построение и решение системы линейных уравнений), в секундах (с точностью до сотых долей),
- `t2` – время работы функции, вычисляющей погрешности решения (см. (1), (2), (3), (4)), в секундах (с точностью до сотых долей),
- `it` – число итераций, потребовавшееся для решения системы уравнений с разреженной матрицей,
- `k`, `nx` $= n_x$, `ny` $= n_y$, `p` – аргументы командной строки.

**Вывод должен производиться в точности в таком формате**, чтобы можно было автоматизировать обработку запуска многих тестов.

6. Реализованные в программе методы интерполяции должны проходить, как минимум, следующие проверки

   - **Быть точными на многочлене "правильной" степени**. Быть точным означает, что для минимально возможного $nx, ny$ (например, $nx = ny = 5$) погрешность метода на таком многочлене имеет порядок машинной точности. Все методы, рассматриваемые в курсе, точны на многочленах степени 0 и 1. Для каждого метода из описания вытекает степень многочлена, на которой он точен.
   - **Погрешность метода должна падать в "правильное" число раз при удвоении $nx$ и $ny$**. Асимптотическое поведение точности метода указано в его описании. Асимптотику надо проверять для достаточно больших $nx, ny$, обычно 50–100.
   - **Не должно быть утечек памяти в самой программе**.
   - **Программа должна быть самостоятельно написанной**. Не должно быть сходства с вариантами из сети Интернет.

## ЗАДАЧИ

1. Построение приближения функции конечными элементами степени 1 методом наименьших квадратов для параллельных ЭВМ с общей памятью.
   - Итерационный метод: метод минимальных невязок с предобуславливателем
   - Предобуславливатель: предоблавливатель Якоби
2. Построение приближения функции конечными элементами степени 1 методом наименьших квадратов для параллельных ЭВМ с общей памятью.
   - Итерационный метод: метод минимальных невязок с предобуславливателем
   - Предобуславливатель: блочный симметричный предобуславливатель Зейделя
3. Построение приближения функции конечными элементами степени 1 методом наименьших квадратов для параллельных ЭВМ с общей памятью.
   - Итерационный метод: метод минимальных невязок с предобуславливателем
   - Предобуславливатель: блочный симметричный предобуславливатель верхней релаксации
4. Построение приближения функции конечными элементами степени 1 методом наименьших квадратов для параллельных ЭВМ с общей памятью.
   - Итерационный метод: метод минимальных невязок с предобуславливателем
   - Предобуславливатель: блочное неполное разложение Холецкого

5. Построение приближения функции конечными элементами степени 1 методом наименьших квадратов для параллельных ЭВМ с общей памятью.
   - Итерационный метод: метод минимальных ошибок с предобуславливателем
   - Предобуславливатель: предоблавливатель Якоби
6. Построение приближения функции конечными элементами степени 1 методом наименьших квадратов для параллельных ЭВМ с общей памятью.
   - Итерационный метод: метод минимальных ошибок с предобуславливателем
   - Предобуславливатель: блочный симметричный предобуславливатель Зейделя
7. Построение приближения функции конечными элементами степени 1 методом наименьших квадратов для параллельных ЭВМ с общей памятью.
   - Итерационный метод: метод минимальных ошибок с предобуславливателем
   - Предобуславливатель: блочный симметричный предобуславливатель верхней релаксации
8. Построение приближения функции конечными элементами степени 1 методом наименьших квадратов для параллельных ЭВМ с общей памятью.
   - Итерационный метод: метод минимальных ошибок с предобуславливателем
   - Предобуславливатель: блочное неполное разложение Холецкого