ТРЕБОВАНИЯ К ПРОГРАММАМ

1. Программа должна получать **все начальные параметры в качестве аргументов командной строки**. Программа имеет 12 обязательных аргументов:

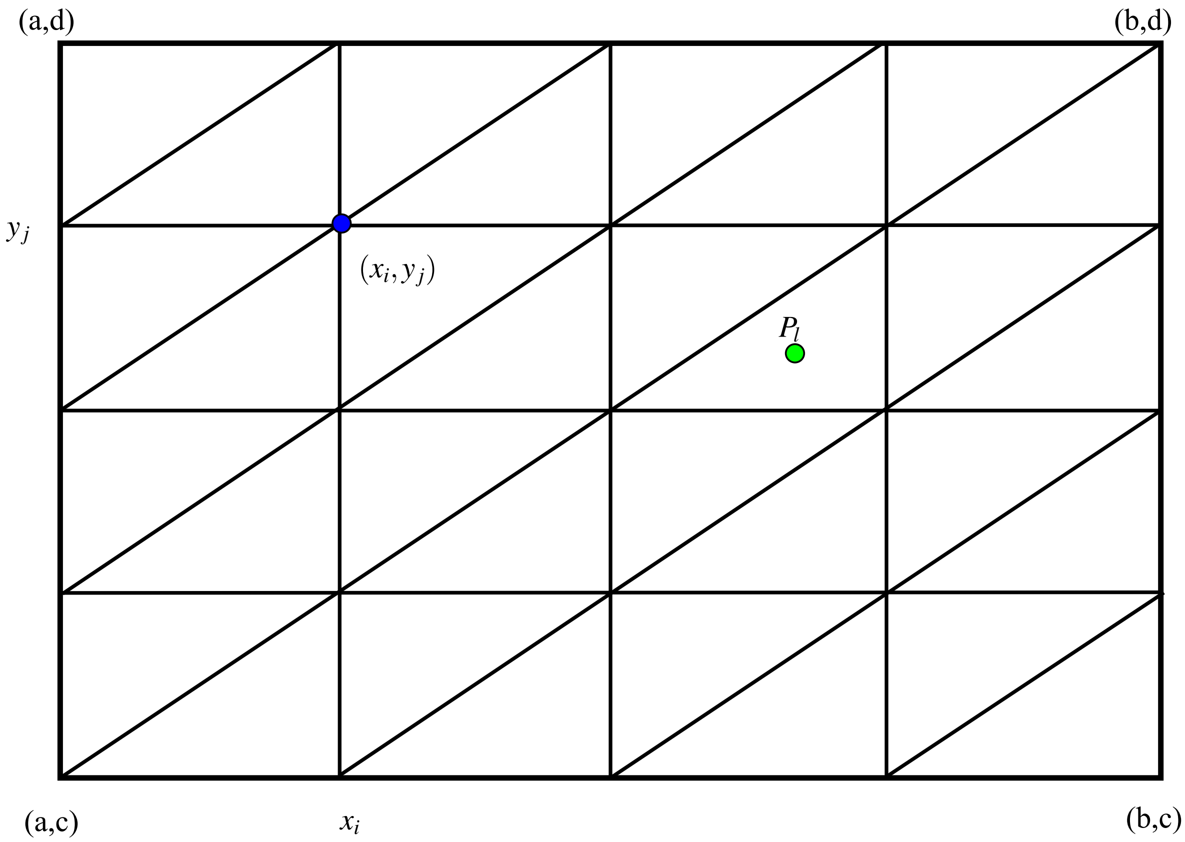
   (1–4) a, b, c, d – спецификация области $[a,b]\times[c,d]$ (тип double),
   (5, 6) n_x, n_y – начальное значение для числа точек интерполяции по осям X и Y (тип int),
   (7, 8) m_x, m_y – начальное значение для числа точек визуализации по осям X и Y (тип int),
   (9) k – начальное значение номера приближаемой функции (тип int),
   (10) ε – точность решения системы линейных уравнений (тип double),
   (11) m_i – максимальное число итераций для решения системы линейных уравнений (тип int),
   (12) 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 предобуславливателя
   - 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. После завершения работы алгоритма построения приближающей функции, в консоль должны выводиться следующие параметры по указанному ниже 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` $= n_y$ – текущее значение k, n_x, n_y,
- `eps` $= \varepsilon$, `p` – аргументы командной строки.

6. Программа должна содержать подпрограмму графического представления заданной функции в окне приложения, разработанного с помощью **библиотеки Qt5**. Эта подпрограмма должна находиться в отдельном файле. Функция должна:

   - осуществлять **отображение визуализируемой части области** $[\widehat{a},\widehat{b}]\times[\widehat{c},\widehat{d}]$, $a \le \widehat{a} < \widehat{b} \le b$, $c \le \widehat{c} < \widehat{d} \le d$, **на область рисования** $[1;W]\times[1;H]$, где W, H – ширина и высота области, например, при коэффициенте масштабирования (zoom), равном 1, отображать всю область $[a,b]\times[c,d]$ на область рисования (т.е. при zoom=1 $\widehat{a}=a$, $\widehat{b}=b$, $\widehat{c}=c$, $\widehat{d}=d$);
   - представлять значение функции цветом на 2D области рисования;
   - область рисования представляет из себя образ "визуализационной" сетки из точек $(\widehat{x}_i,\widehat{y}_j)$ на визуализируемой части области $[\widehat{a},\widehat{b}]\times[\widehat{c},\widehat{d}]$,

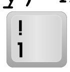
     $$\widehat{x}_i=\widehat{a}+i\widehat{h}_x,\ \widehat{h}_x=(\widehat{b}-\widehat{a})/m_x,\quad \widehat{y}_j=\widehat{c}+j\widehat{h}_y,\ \widehat{h}_y=(\widehat{d}-\widehat{c})/m_y,\quad i=0,\dots,m_x,\ j=0,\dots,m_y,$$

     каждый треугольник этой сетки заливается цветом, вычисляемым отображением значения визуализируемой функции в центре тяжести треугольника на палитру;
   - палитра представляет из себя отображение отрезка вещественных чисел на набор цветов, например,

     $$\alpha\in[0;1]\to(\mathrm{r,g,b})=(\alpha*255,\alpha*255,\alpha*255)$$

     представляет из себя отображение отрезка $[0;1]$ на оттенки серого при использовании (r,g,b)=(red,green,blue) нотации для обозначения цвета;
   - вычислять максимальное значение функции на области рисования и **осуществлять масштабирование значений функции** на палитру рисования для того, чтобы значения функции использовали все значения палитры;
   - выводить **на графический экран и в консоль** максимальное по модулю значение функции.

7. Интерфейсная часть программы по нажатию указанной клавиши **должна**:

   - По нажатию клавиши  циклически **менять номер k приближаемой функции** и **перерисовывать новый график**. Значение номера приближаемой функции k, а также текстовое представление функции **должно выводиться** в графическом окне (например, выводится `k=3 f(x,y)=x+y`).
   - По нажатию клавиши  циклически **менять состав отображаемых графиков** и **перерисовывать новый график**:

(a) показывать график функции;
(b) показывать график ее приближения;
(c) показывать график погрешности приближения.

Каждый из графиков отображается **в своем масштабе, причем разном для осей XY и палитры**, так, чтобы значения отображаемого в текущий момент графика использовали все значения палитры. Значение величины $\max\{|F_{min}|, |F_{max}|\}$ **должно выводиться** как в графическом окне, так и на текстовой консоли, где F_{min} – минимальное значение визуализируемого графика в области, F_{max} – максимальное значение визуализируемого графика в области.

- По нажатию клавиши 2 увеличивать, а по нажатию клавиши 3 уменьшать масштаб текущего графика, осуществляя **двукратное растяжение/сжатие** осей XY относительно центра тяжести области и **перерисовку графика в новом масштабе**. Например, если s раз нажать клавишу 2, то визуализируемый график отображается **в своем масштабе, причем разном для осей XY и палитры**, так, чтобы значения отображаемого в текущий момент графика над $1/2^s$ частью заданной области использовали все значения палитры. Значение величины $\max\{|F^{(s)}_{min}|, |F^{(s)}_{max}|\}$, а также значение величины текущего масштаба s **должно выводиться** в графическом окне, где $F^{(s)}_{min}$ – минимальное значение визуализируемого графика в $1/2^s$ части области, $F^{(s)}_{max}$ – максимальное значение визуализируемого графика в $1/2^s$ части области.
- По нажатию клавиши 4 увеличивать, а по нажатию клавиши 5 уменьшать в 2 раза **число точек приближения** n_x, n_y и **перерисовывать графики для нового числа точек приближения**. Значение текущего числа точек n_x, n_y **должно выводиться** в графическом окне.
- По нажатию клавиши 6 прибавлять, а по нажатию клавиши 7 вычитать к/от вычисленному значению функции $f_{n/2} = f(x_{n_x/2}, y_{n_y/2})$ одну десятую максимума функции f в исходной области, моделируя погрешность измерения, и **перерисовывать новый график**. Например, если p раз нажать клавишу 6, то все приближения и графики строятся не для функции $f(x,y)$, а для функции $\hat{f}(x,y)$, где

$$\hat{f}(x,y) = \begin{cases} f(x,y) & (x,y) \neq (x_{n_x/2}, y_{n_y/2}) \\ f(x,y) + p * 0.1 * \max\limits_{\Omega} |f| & (x,y) = (x_{n_x/2}, y_{n_y/2}) \end{cases}$$

  Значение текущего возмущения p **должно выводиться** в графическом окне.
- По нажатию клавиши 8 увеличивать, а по нажатию клавиши 9 уменьшать в 2 раза **число точек визуализации** m_x, m_y и **перерисовывать графики для нового числа точек визуализации**. Значение текущего числа точек m_x, m_y **должно выводиться** в графическом окне.

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

   - **Быть точными на многочлене "правильной" степени**. Быть точным означает, что для минимально возможного n_x, n_y (например, $n_x = n_y = 5$) 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** n_x **и** n_y. Асимптотическое поведение точности метода указано в его описании. Асимптотику надо проверять для достаточно больших n_x, ny, обычно 50–100.
- **Методы кусочно многочленной аппроксимации должны практически мгновенно работать для** $n_x * n_y = 10^7$. Как время работы метода, так и время обновления экрана не должны превышать 1 секунды даже на компьютерах десятилетней давности для $n_x * n_y = 10^7$ (десять миллионов точек интерполяции).
- **Скорость перерисовки экрана при изменении размера окна для** $n_x * n_y = 10^7$ **должна быть практически мгновенной**. Время обновления окна не должно превышать 1 секунды даже на компьютерах десятилетней давности без аппаратного графического ускорителя для $n_x * n_y = 10^7$ (десять миллионов точек интерполяции).
- **Не должно быть утечек памяти в самой программе**. Утечки в используемой библиотеке Qt5 допустимы.
- **Программа должна быть самостоятельно написанной, как метод, так и графический интерфейс**. 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