Consider the problem: Find the solution $u = u(x, y)$ of the following boundary value problem:

$$- \nabla \cdot (k \nabla u) = f(x, y), \quad (x, y) \in \Omega = (0, 1) \times (0, 1), \quad u(x, y) = 0 \quad (x, y) \in \partial \Omega.$$ 

Here $\partial \Omega$ is the boundary of $\Omega$, $k = k(x, y)$, and $f$, are given functions of their arguments.

Use 5-point finite difference approximation of the above problem on a square mesh with a step-size $h$. Namely, if $(x_i, y_j) = (ih, jh)$, $i, j = 0, ..., N$ are the grid points and $h = 1/N$ is the step-size, the approximation $U_{i,j}$ of $u(x_i, y_j)$ is a solution to the system of linear equations (five-point finite differences):

$$d_{i,j}U_{i,j} - k_{i-1/2,j}U_{i-1,j} - k_{i+1/2,j}U_{i+1,j} - k_{i,j-1/2}U_{i,j-1} - k_{i,j+1/2}U_{i,j+1} = h^2 f(x_i, y_j),$$

$$d_{i,j} = k_{i-1/2,j} + k_{i+1/2,j} + k_{i,j-1/2} + k_{i,j+1/2}, \quad i, j = 1, ..., N - 1,$$

and $U_{i,j} = 0$, for $(x_i, y_j) \in \partial \Omega$.

Here $k_{i-1/2,j} = k(x_i - \frac{h}{2}, y_j)$, $k_{i+1/2,j} = k(x_i + \frac{h}{2}, y_j)$, $k_{i,j-1/2} = k(x_i, y_j - \frac{h}{2})$, and $k_{i,j+1/2} = k(x_i, y_j + \frac{h}{2})$.

As always, we write this system in the form $Ax = b$. For a preconditioner use the diagonal $D$ of the matrix $A$ and use a subroutine for evaluating the action of the matrix $A$ on a given vector $x$.

Consider the following a problem with:

$$k(x, y) = \begin{cases} 
1, & \text{for } 0 < x < 0.5, \\
K, & \text{for } 0.5 \leq x < 1,
\end{cases}$$

Run two different cases of $K = 1$ and $K = 100$ and $f(x, y) = 1$.

Use the iteration methods that you have programmed in this class: namely: Jacobi, Gauss-Seidel, SOR, Chebyshev acceleration with 32 parameters, steepest descent, and conjugate gradient. Use the preconditioned variants whenever possible and relevant stopping criteria with the same tolerance say $10^{-7}$.

Report the number of iterations for all six methods for a problem on a $80 \times 80$ mesh and two different $K$, namely, $K = 1, 100$. If you are able to clock, report also the computational time.