Exercise 3.1. Let $A$ be the $n \times n$ block tridiagonal matrix

$$A = \begin{pmatrix}
T & -I & & \\
-I & T & -I & \ \\
& \ddots & \ddots & \ddots \\
& & -I & T & -I \\
& & & -I & T
\end{pmatrix}$$

where $T$ is the $n \times n$ tridiagonal matrix

$$T = \begin{pmatrix}
4 & -1 & & \\
-1 & 4 & -1 & \\
& \ddots & \ddots & \ddots \\
& & -1 & 4 & -1 \\
& & & -1 & 4
\end{pmatrix}$$

Let $A = D - L - L^t$, where $D = 4I$, and $L$ is strictly lower triangular. Write a program to solve $Ax = b$ by the following methods:

- SSOR
- Jacobi
- SSOR-CG
- Jacobi-CG

I suggest that you organize the calculation as follows: Write a conjugate gradient routine in which the matrix multiply (form $y = Ax$ given $x$) and the preconditioning (solve $Mx = y$ given $y$) are separate subroutines. Then write subroutines for Jacobi and SSOR preconditioning. You can then recover plain Jacobi or SSOR by setting $\alpha = 1, \beta = 0$ in your CG routine (obviously not the most efficient, but it should involve the least amount of coding).

To carry out experiments, one can follow this general procedure: Choose your favorite vector $x$ (a random vector with entries on $(-1,1)$ is a good choice). Form a right hand side $b = Ax$. Next, compute the approximate solution of $A\tilde{x} = b$ by the iterative method, with initial guess $\tilde{x}_0 = 0$. Since you know the solution ($\tilde{x} = x$), you can monitor the error, and determine the rate of convergence. Use the energy norm $\|x\|^2 = x'Ax$ to measure the error.

An equally good procedure is to solve $Ax = 0$ (with solution $x = 0$) starting with a nonzero initial guess $x_0$. $x_0$ could be a random vector with entries on $(-1,1)$. In this case the error is just $x_k$ itself.
Exercise 3.2. First compare the Jacobi and Jacobi-CG methods. For \( n = 10, 20, 40, 80 \), compute the number of iterations to reduce the initial error by \( 10^{-6} \). Compute the average error reduction factor per step \( \gamma \). Use this to form an estimate of the generalized condition number. Compare with the exact generalized condition number.

Exercise 3.3. Now compare the SSOR and SSOR-CG methods as in exercise 2.2. Use the same values of \( n \) and try \( \omega = 1 \) and \( \omega = 1.5 \). You do not have to compute the exact generalized condition number.

Exercise 3.4. For a fixed \( n \), say \( n = 80 \), study the rate of convergence of SSOR-CG as a function of \( \omega \). Solve the same problem using SSOR-CG and \( \omega = 1.0, 1.1, 1.2, ..., 1.9 \) to get some idea of the sensitivity of the rate of convergence to the choice of \( \omega \).