True or false? (unless otherwise specified.)

Answer grid

	1	2	3	4	5	6	7	8	9	10	11
True	☐	☐	☒	☐	☐	☒	☐	☐	☒	☒	☒
False	☒	☒	☐	☒	☒	☐	☒	☒	☐	☐	☐

1. **Direct methods.** The computational cost of the forward substitution algorithm, for a matrix of size $n \times n$, scales as $\mathcal{O}(n)$.

   False. Computing the i-th unknown requires $\mathcal{O}(i)$ operations, so the total cost scales as $\sum_{i=1}^{n} \mathcal{O}(i) = \mathcal{O}(n^2)$.

2. **Direct methods.** The computational cost to calculate the LU decomposition of an invertible matrix scales as $\mathcal{O}(n^2)$.

   False. The computational cost scales as $\frac{2}{3}n^3 + \mathcal{O}(n^2)$ in general.

3. **Direct methods.** Suppose that A is a symmetric, positive definite matrix. Calculating the Cholesky decomposition of A requires fewer floating point operations than calculating the LU decomposition of A.

   True. The computational cost of the Cholesky decomposition is $\frac{1}{3}n^3 + \mathcal{O}(n^2)$, about half that of the LU decomposition.

4. **Conditioning.** The condition number $\kappa_2(\mathsf{A})$ of a square invertible matrix A is always strictly larger than 1.

   False. The condition number is always greater than *or equal to* 1. For the identity matrix, the condition number is 1.

5. **Spectral radius.** The spectral radius of a square matrix A is zero if and only if A is zero.

   False. For example, the only eigenvalue of the nonzero matrix `A = [0 1; 0 0]` is 0, so its spectral radius is 0.

6. **Conditioning.** If the condition number of a matrix A is very large ($\gg 1$), then it is possible that rounding errors arising from floating point arithmetic will have a large impact on the accuracy of the numerical solution to the linear system $\mathsf{A}\boldsymbol{x} = \boldsymbol{b}$ (calculated by LU decomposition followed by forward and backward substitution, for example).

   True. The condition number bounds the amplification of relative perturbations of the data, such as those caused by roundoff errors.

7. **Richardson's method.** Suppose that $\mathsf{A} \in \mathbf{R}^{n\times n}$ is symmetric and positive definite, and let $\boldsymbol{b} \in \mathbf{R}^n$. Consider the following iterative method for solving the linear system $\mathsf{A}\boldsymbol{x} = \boldsymbol{b}$:

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} + \omega \left(\boldsymbol{b} - \mathsf{A}\boldsymbol{x}^{(k)} \right). \tag{1}$$

   This iteration converges to the exact solution of the linear system for all $\omega \in \mathbf{R}$.

   False. For example, for $\omega = 0$, it holds that $\boldsymbol{x}^{(k)} = \boldsymbol{x}^{(0)}$ for all k. Therefore, unless $\boldsymbol{x}^{(0)}$ coincides with the solution of the linear system, the iteration does not converge to the solution.

8. **Richardson's method.** The convergence speed of the iteration (1), for the optimal value of ω, is independent of the condition number $\kappa_2(\mathsf{A})$.

   False. For the optimal value of ω, the error is multiplied at each iteration by at most $\rho(\mathsf{I} - \omega_*\mathsf{A}) = \frac{\kappa_2(\mathsf{A})-1}{\kappa_2(\mathsf{A})+1}$: the larger the condition number, the slower the convergence.

9. **Richardson's method.** If A is symmetric positive definite, there always exists $\omega \in \mathbf{R}$ such that the iteration (1) converges.

   True. Any $\omega \in \left(0, 2/\lambda_{\max}(\mathsf{A})\right)$ works, since then $|1 - \omega\lambda| < 1$ for every eigenvalue λ of A.

10. **Norms.** If A is a nonzero matrix, then its norm $\|\mathsf{A}\|_2$ is strictly positive.

    True. If this was false, then $\|\cdot\|_2$ would not be called a norm.

11. **Direct methods.** Suppose that we want to solve the linear systems $\mathsf{A}\boldsymbol{x} = \boldsymbol{b}$ and $\mathsf{A}\boldsymbol{x} = \boldsymbol{c}$ by a direct method. To this end, the LU decomposition of A can be calculated only once.

    True. This is the advantage of calculating the LU decomposition, compared to Gaussian elimination with corresponding operations on the right-hand side.

12. **Julia.** Assuming that the matrix `A` and the vector `b` are already defined, write (on paper) Julia code implementing 100 iterations of (1).

    For example:

```julia
w = .01
x = zeros(length(b))  # Initial guess
for i in 1:100
    x += w * (b - A*x)
end
```