Exercise 13
Fortran, Linear Algebra

1. Task Recursion (3 P)
   Recursion can be easily and clearly demonstrated for the example of calculating the factorial \( n! \). A small Fortran program should read in from STDIN an INTEGER number and calculate for it the factorial with help of a recursive function call.

2. Task Solving systems of linear equations (10 P)
   Let
   \[
   A = \begin{pmatrix}
   \pi & \pi & \ldots & \pi \\
   0 & \pi & \ldots & \pi \\
   0 & 0 & \pi & \ldots \\
   0 & 0 & 0 & \ldots \\
   \end{pmatrix} \in \mathbb{R}^{n \times n}
   \text{ and } \quad
   b = \begin{pmatrix}
   n \\
   \vdots \\
   \vdots \\
   n \\
   \end{pmatrix} \in \mathbb{R}^n
   \] (1)

   for different values of \( n \geq 1000 \).
   Write a program (C/C++, better in Fortran) that solves \( Ax = b \) for \( x \) numerically
   a) column-wise
   b) row-wise

   and measure the runtime of both versions with help of, e.g., `omp_get_wtime()`. Make \( n \) sufficiently large to get significant different runtimes. Explain the difference.
   \textit{Hint:} While the row-wise version might be straightforward to program (outer loop over first index \( i \) from \( n - 1 \) to 1), the column-wise implementation (outer loop over second index \( j \) from \( n - 1 \) to 1) looks in pseudo code like that:

   \begin{verbatim}
   for j = n-1 ... 1
       for i = 1 ... j
           b[i] = b[i] - a[i][j+1] * x[j]
           x[j] = b[j] / a[j][j]
   \end{verbatim}