1 Gram-Schmidt Algorithm

1. Implement the Gram-Schmidt algorithm as a function `QR()`, which takes a matrix A as input and returns the matrices Q and R. Test your algorithm by computing $Q^\top Q$ and $QR - A$, where the first should yield the identity and the latter a zero-matrix.

   Hint: If you use `np.round(...,2)` on $Q^\top Q$ and $QR - A$ it will be easier to check your results.

2. Implement the QR-Eigenvalue algorithm (see lecture page 68) as a function `eig()`. The function shall take a matrix A as input and return the diagonal of the last iterate A_n. Test your results against `np.linalg.eig()` with some symmetric, positive semi-definite matrix A as input. Note that such a matrix has only nonnegative eigenvalues so that there are no 2×2 blocks on the diagonal (as they can occur for complex eigenvalues within the Schur decomposition).

   Hint: You can directly access the diagonal of a `numpy.array` by `A.diagonal()`.

Solution: