**Problem.** Write from-scratch code to perform principal component analysis on given data. Use eigendecomposition of the correlation matrix for this purpose.

**Input.** `X`: $n \times p$ numeric matrix (rows: cases/samples, columns: variables/factors); without any missing values.

**Output.** Suppose $k = \min(n, p)$.

1. Loadings/rotations: $p \times k$ matrix.
2. Principal components/scores: $n \times k$ matrix.
3. Standard deviations: $k$-vector.

**Checks on input arguments.** Valid values in the input arguments, no missing values, etc. Treat end-cases such as `n < 2` and `p < 2` separately.

**Algorithm.**

1. Shift each column of $X$ by its own mean; i.e., $Y_{\cdot,i} = X_{\cdot,i} - \text{Mean}(X_{\cdot,i})$ for each column $i = 1, \ldots, p$. Scale each column of $Y$ by its own standard error; i.e., $Y_{\cdot,i} = Y_{\cdot,i}/\text{SE}(Y_{\cdot,i})$ for each column $i = 1, \ldots, p$.
2. Compute the $p \times p$ correlation matrix $C = Y^T Y/(n-1)$. $C$ should be symmetric, all 1s on the diagonal, and all other elements between $-1$ and $+1$.
3. Compute the eigendecomposition of $C$ using the in-built `function eigen()`. This gives a $p$-vector of eigenvalues $d$ and a $p \times p$ matrix $V$ with eigenvectors as its columns. Formally, the eigendecomposition is $C = V^T \cdot \text{diag}(d) \cdot V$.
4. (a) Check if the eigenvalues $d > 0$. If not, take an appropriate course of action.
   (b) Check if the eigenvalues $d$ are in descending order. If not, then reorder $d$ in descending order. Reorder the columns of $V$ to match the changed order.
5. Compute the output quantities:
   (a) Rotation/loading matrix $R$ is the matrix of first $k$ columns of $V$.
   (b) Scores/principal component matrix (with PCs as columns) is $YR$.
   (c) Standard deviations of the PCs are the square roots of first $k$ eigenvalues in $d$.

**Testing.**

1. At each step of the algorithm, put appropriate checks that reflect the assumptions made about the computed quantities.
2. Compare the results of your implementation with the output of the in-built `function princomp()` applied to a standard data set such as `USArrests` or `iris` without the species column. The simplest artificial test data set would be a 2-variable $(X_1, X_2)$ data set where $X_2 = mX_1 + c + \epsilon$ where the noise $\epsilon$ is normal with mean 0 and standard deviation $\sigma > 0$.
3. Demonstrate that your code produces correct results.

**Problem.** Write from-scratch code to perform principal component analysis on given data. Use singular value decomposition of the data matrix for this purpose.

**Input.** `X`: $n \times p$ numeric matrix (rows: cases/samples, columns: variables/factors); without any missing values.

**Output.** Suppose $k = \min(n, p)$.

1. Loadings/rotations: $p \times k$ matrix.
2. Principal components/scores: $n \times k$ matrix.
3. Standard deviations: $k$-vector.

**Checks on input arguments.** Valid values in the input arguments, no missing values, etc. Treat end-cases such as `n < 2` and `p < 2` separately.

**Algorithm.**

1. Shift each column of $X$ by its own mean; i.e., $Y_{\cdot,i} = X_{\cdot,i} - \text{Mean}(X_{\cdot,i})$ for each column $i = 1, \ldots, p$. Scale each column of $Y$ by its own standard error; i.e., $Y_{\cdot,i} = Y_{\cdot,i}/\text{SE}(Y_{\cdot,i})$ for each column $i = 1, \ldots, p$.
2. Compute the singular value decomposition of $Y$ using the in-built `function svd()`. SVD gives a $k$-vector of singular values $d$, a $n \times k$ matrix $U$ (left singular vectors as columns), and a $p \times k$ matrix $V$ (right singular vectors as columns). Formally, the singular value decomposition is $Y = U \cdot \text{diag}(d) \cdot V^T$.
3. (a) Check if the singular values $d > 0$. If not, take an appropriate course of action.
   (b) Check if the singular values $d$ are in descending order. If not, then reorder $d$ in descending order. Reorder the columns of $U$ and $V$ to match the changed order.
4. Compute the output quantities:
   (a) Rotation/loading matrix $R$ is the matrix $V$.
   (b) Scores/principal component matrix (with PCs as columns) is $YR$.
   (c) Standard deviations of the PCs are $d/\sqrt{n}$.

**Testing.**

1. At each step of the algorithm, put appropriate checks that reflect the assumptions made about the computed quantities.
2. Compare the results of your implementation with the output of the in-built `function princomp()` applied to a standard data set such as `USArrests` or `iris` without the species column. The simplest artificial test data set would be a 2-variable $(X_1, X_2)$ data set where $X_2 = mX_1 + c + \epsilon$ where the noise $\epsilon$ is normal with mean 0 and standard deviation $\sigma > 0$.
3. Demonstrate that your code produces correct results.