CHEM 225 Name: _____

Programming Exercise 3

Complete the following problems.
This assignment is due by 11:59 pm on April 22, 2019.

These programming exercises build on concepts developed in previous programming exercises. You should (always!) feel free to reuse code from your previous work.

1. **PROGRAMMING EXERCISE** Write and compile a program named `pgrm_03_01.exe` that reads a symmetric matrix stored in a user-provided input file and forms the inverse square-root of the matrix returned in full N×N storage. After forming the inverse square-root matrix, the program should form the product of the inverse square-root with itself and the input matrix using the intrinsic function `MatMul` to verify it is correct (which should yield the identity matrix).

   The program should rely on the LAPack routine `SSPEV`. Note that `SSPEV` requires a symmetric packed-storage input matrix. The input matrix may be overwritten during the diagonalization process; therefore, it is important that you copy the matrix to a temporary array before calling `SSPEV`.

   An example main program for this exercise is:

   Two example input files (`input_03_1.txt` and `input_03_2.txt`) are provided in your GitHub repositories. Output files, with the `out` extension, are provided in the same directory.

2. **PROGRAMMING EXERCISE** Write and compile a program named `pgrm_03_02.exe` that reads a symmetric square matrix given in full-storage form from a user-provided input file and *symmetrizes* and *linearizes* to upper/column storage form of the matrix. This scheme will begin with a matrix such as

$$\begin{pmatrix} 4 & 2 & 1 \\ 2 & 6 & 4 \\ 1 & 4 & 3 \end{pmatrix}$$

   and converts it to a rank-1 array as

$$\begin{pmatrix} 4 \\ 2 \\ 6 \\ 1 \\ 4 \\ 3 \end{pmatrix}$$

   An example main program for this exercise is:

   Two example input files (`input_02_3.txt` and `input_02_4.txt`) are provided in your GitHub repositories. Output files, with the `out` extension, are provided in the same directory.

3. **PROGRAMMING EXERCISE** Write and compile a program named `pgrm_03_03.exe` that computes the one-electron energy contribution, two-electron energy contribution, and the density matrix

from a Hartree-Fock program. As input, the program will be given the number of electrons (the program will assume closed-shell and compute the number of occupied molecular orbitals), number of basis functions, and names for files that provide the core Hamiltonian, overlap, and Fock matrices from a converged calculation. These files will be provided by the instructor (*vide infra*).

After reading in the provided data, the program should form the inverse-square-root of the overlap matrix, $\mathbf{S}^{-1/2}$, and transformed Fock matrix based on the symmetric decomposition of the atomic orbital basis, $\widetilde{\mathbf{F}} = \mathbf{S}^{-1/2}\mathbf{F}\mathbf{S}^{-1/2}$.

Using LAPack routine `SSPEV`, the eigenvectors of $\widetilde{\mathbf{F}}$ can be found and then transformed to the canonical MO coefficient matrix.

Once the canonical MOs are known, the program print the MO energies and then form the density matrix in the atomic orbital basis and print it.

Then, the program should evaluate and print the one-electron, $\langle \mathbf{PH} \rangle$, and two-electron, $\frac{1}{2} \langle \mathbf{PG} \rangle$, contributions to the total electronic energy. Note that $\mathbf{G} = \mathbf{F} - \mathbf{H}$.

Finally, the program should compute and print the contraction of the density and overlap matrices, $\langle \mathbf{PS} \rangle$. Note that this result should equal the number of electrons in the system.

Two sets of input files (`input_03_5_*.txt` and `input_03_6_*.txt`, where the *'s are h, s, and f for the three input matrices and *=ABOUT has text giving information about the input jobs include number of electrons and number of basis functions) are provided in your GitHub repositories. Output files, with the `out` extension, are provided in the same directory.

To assist with your writing of this program, here are a few code snippets from a working code.

The top of a working version of this program could look like:

Using routines written earlier, the following is a bit of code that can form $\mathbf{S}^{-1/2}$ and $\widetilde{\mathbf{F}}$. This code snippet also demonstrates the use of the fortran intrinsic function `Reshape`, which can be used to have vectors treated as a single column or row in a matrix, etc.

Running the program with the command

```
./pgrm_03_03.exe 2 2 input_03_4_h.txt input_03_4_s.txt input_03_4_f.txt
```

should yield the output: