Problem

The purpose of this assignment is to give you more practice with 2D Arrays and Matrix operations. You can start with a completed version of 15L and copy the files over to start 16L since it's the continuation of your previous lab assignment.

Scenario

The organizers of an in-house software engineering conference for a small consulting company are trying to minimize scheduling conflicts by scheduling the most popular presentations at different times. First the planners survey the ten participants to determine which of the five presentations they want to attend. They then construct a matrix **A** (let's call it the ***preference matrix***) in which a 1 in entry ***ij*** means that participant ***i*** wants to attend presentation ***j***.

	Presentation				
Participant	**1**	**2**	**3**	**4**	**5**
1	1	0	1	0	1
2	0	0	1	1	1
3	1	0	0	0	0
4	0	1	1	0	1
5	0	0	0	0	0
6	1	1	0	0	0
7	0	0	1	0	1
8	0	1	0	1	0
9	1	0	1	0	1
10	0	0	0	1	0

Next the planners calculate the transpose of matrix **A** which is $\mathbf{A}^t$ and the matrix product $\mathbf{A}^t\,\mathbf{A}$.
The transpose of a matrix is formed by interchanging the matrix's rows and columns. Thus the transpose of matrix

$$\mathbf{X} = \begin{matrix} 2 & 4 \\ 6 & 8 \\ 10 & 12 \end{matrix} \qquad \mathbf{X}^t = \begin{matrix} 2 & 6 & 10 \\ 4 & 8 & 12 \end{matrix}$$

In the resulting matrix from the matrix product $\mathbf{A}^t\,\mathbf{A}$ (from above), entry ***ij*** is the number of participants wishing to attend both presentation ***i*** and presentation ***j***.

$\mathbf{A^t A}$ =	4	1	2	0	2
	1	3	1	1	1
	2	1	5	1	5
	0	1	1	3	1
	2	1	5	1	5

Notice that $\mathbf{A^t A}$ is symmetric ($\mathbf{a_{ij} = a_{ji}}$ for all $\boldsymbol{i, j}$), so the entries below the main diagonal (entries $\boldsymbol{ij}$ where $\boldsymbol{i>j}$) need not be calculated. If we supply zeroes for the unnecessary entries, the resulting matrix is termed an *upper triangular matrix*. The entries on the main diagonal ($\mathbf{a_{ii}}$) represent the total participants wanting to attend presentation $\boldsymbol{i}$.

Write a program that inputs a matrix (preference matrix) from a data file of participant preferences. The first line of this file should contain the matrix dimensions: For the preference matrix shown above, this line would be:

10 5

Subsequent lines should be the rows of the matrix. After displaying the preference matrix $\mathbf{A}$, calculate and display $\mathbf{A^t A}$ and output sentences indicating how many participants wish to attend each presentation.

Finally, find in the upper triangular matrix of $\mathbf{A^t A}$, all the pairs of presentations that the conference committee should avoid scheduling opposite one another. You will display pairs if there is more than **three** Participants attending the pair of presentations.

Input

The input data is coming from data files supplied as command line arguments (via argc, argv). You should read the data into your 2-D array, after you read the first line which contains the size of the Participant Preference array. You can assume that the maximum size of this array is going to be 25 x 25 (25 Rows for number of Participants, 25 Columns for number of Presentations)

Output

The output must first print out the preference matrix $\mathbf{A}$ that is read from the data file. Then it must print the resulting matrix $\mathbf{A^t A}$. (You should be able to use the same function print_matrix() to print each matrix at different times). Following this you must print the rest of the details outlined above.

Requirements

- You must use at least the following functions (all of them must be commented):
    - Function read_data() which reads information into the 2-D array from the data file. It takes 4 arguments, the filename, the array that needs to be filled, the actual number of rows of the preference matrix, the actual number of columns of the preference matrix (other than the filename the rest of them are output arguments).

    - Function print_matrix() which prints a matrix. It takes 3 arguments, the array that needs to be printed, the number of rows, and the number of columns (all of them are input arguments).
    - Function compute_transpose_matrix() which computes the transpose matrix. It takes 4 arguments, the array holding the preference matrix, number of rows of the preference matrix, number of columns of the preference matrix, a resulting array which is the transpose of the preference matrix.
    - Function matrix_multiply() which computes the product of two matrices. It takes 6 arguments, the first matrix (array), the second matrix (array), the number of rows of the first matrix, the number of columns of the first matrix, the number of columns of the second matrix, the resulting product matrix (array).

- The program must be split into multiple files:
    - The main must be the only function in one code file (.c)
    - All the other functions should be in a second code file (.c)
    - The other functions should be forward declared (prototype declarations) in a header file (.h) and included in both code files
    - **Be sure to submit all three files!**

- All function definitions must have a block comment describing what they do.

- Work on the program using the modularity of the functions, for example write a program that reads the data and prints it out. Once you have these working you can move on to the other functions

- As always test your program carefully and follow good programming style.

Other Details

- You must follow all the coding style rules as specified in our "coding guidelines". In particular:
    - You must put your name enclosed in a comment box at the top, and keep any other comments that are already there.
    - Keep lines to the point of making your code easily readable. It is a good idea to make your comment box lines 66-80 characters long and use this as a guide.
    - You must use good names for any variables you create (a full word that describes what it is there for).
- Details that you do not follow are penalized after other scored items are added up, so even if you got a 100 for the functionality of your program, you can still get a lower score because you did not follow all the other requirements for the assignment.

Submission

Submit this assignment with the code 16L followed by the names of all 3 files:
For example:

```
submit matrix.c functions.c header.h
```