Matrix Manipulation
15 points
Due Friday, April 4th

Manipulating matrices has many applications in computer science and mathematics. One common operation is called transposition. The transpose is a matrix obtained from a given matrix by interchanging each row and the corresponding column. Some other useful operations are flipping and rotating a matrix. For this assignment, you will use 2D arrays to code some basic matrix operations. For some matrix, you will have to flip it over the X and Y axes, and find the transpose.
Here are some small-scale examples:

original matrix
```
a  b
c  d
```

matrix flipped over x axis
```
c  d
a  b
```

matrix flipped over y axis
```
b  a
d  c
```

matrix transpose
```
a  c
b  d
```

matrix rotated 90 degrees clockwise
```
c  a
d  b
```

For this assignment, you will be provided with two files that are pre-built and have a few functions already written to help you. Included are:
`matrix_manipulation.c`
`matrix_test.h`

Much of the code has already been written for you, including doing the following:

- Declaring, and initializing the original matrix (filled with random integers between 1 and 100, inclusive)
- Declaring the 3 matrices you will create
- Printing out the resulting matrices
- Checking the results of your functions to make sure your answers are correct

Within ``matrix_manipulation.c``, you are expected to fill in some code for these 3 functions:
`flip_x_matrix`
`flip_y_matrix`
`transpose_matrix`

`flip_x_matrix` will take in the original matrix, and flip it vertically over the X axis.
`flip_y_matrix` will take in the original matrix, and flip it horizontally over the Y axis.
`transpose_matrix` will take in the original matrix, and transpose it.

Make sure the `matrix_manipulation.c` and `matrix_test.h` files are in the same directory (folder). The `.c` file requires some functions that are in the `.h` file in order to work. This is done both for your benefit and safety.

You can see all the code that has been written in the files, but here are the functions you need to write code in:

```
int** flip_x_matrix (int** orig_mat)
{
    int** result;
    result = alloc_matrix(result);
    int i, j;

    /*
     * insert code here
     */

    return result;
}
```

```
int** flip_y_matrix (int** orig_mat)
{
    int** result;
    result = alloc_matrix(result);
    int i, j;

    /*
     * insert code here
     */

    return result;
}
```

```
int** transpose_matrix (int**
orig_mat) {
    int** result;
    result = alloc_matrix(result);
    int i, j;

    /*
     * insert code here
     */

    return result;
}
```

Sample Output (note this will be handled for you, with the correct solution):

```
Original Matrix:
  98  63  85  83  97  22  23   1  58  21
   1  13  81   9 100   3  43  86  51  80
  70   8  83  20  36  57  34  90   1  90
  85  38  33  88  23  50  38  10  18  91
  31  68   1  13  80  43  80  25   7   3
  77  67  32  36  41  53  86  21   9  49
  29  97  88  44  29  90  66  72  60  79
  23  15  25  74  78  34  35  49  22  50
   6  32  38  13  66  17  67  38  62  96
  55  62   1  54  89  58  91  20  76  18
Flipped x Matrix:
  55  62   1  54  89  58  91  20  76  18
   6  32  38  13  66  17  67  38  62  96
  23  15  25  74  78  34  35  49  22  50
  29  97  88  44  29  90  66  72  60  79
  77  67  32  36  41  53  86  21   9  49
  31  68   1  13  80  43  80  25   7   3
  85  38  33  88  23  50  38  10  18  91
  70   8  83  20  36  57  34  90   1  90
   1  13  81   9 100   3  43  86  51  80
  98  63  85  83  97  22  23   1  58  21
Flipped y Matrix:
  21  58   1  23  22  97  83  85  63  98
  80  51  86  43   3 100   9  81  13   1
  90   1  90  34  57  36  20  83   8  70
  91  18  10  38  50  23  88  33  38  85
   3   7  25  80  43  80  13   1  68  31
  49   9  21  86  53  41  36  32  67  77
  79  60  72  66  90  29  44  88  97  29
  50  22  49  35  34  78  74  25  15  23
  96  62  38  67  17  66  13  38  32   6
  18  76  20  91  58  89  54   1  62  55
Transposed Matrix:
  98   1  70  85  31  77  29  23   6  55
  63  13   8  38  68  67  97  15  32  62
  85  81  83  33   1  32  88  25  38   1
  83   9  20  88  13  36  44  74  13  54
  97 100  36  23  80  41  29  78  66  89
  22   3  57  50  43  53  90  34  17  58
  23  43  34  38  80  86  66  35  67  91
```

```
   1  86  90  10  25  21  72  49  38  20
  58  51   1  18   7   9  60  22  62  76
  21  80  90  91   3  49  79  50  96  18
Matrix flip x is correct.
Matrix flip y is correct.
Matrix transpose is correct.
```

*****EXTRA CREDIT*****

(worth 5 potential extra points)

There is an extra credit opportunity available on this assignment. Aside from finding the flips and transpose of the matrix, you can also optionally figure out how to rotate the matrix 90 degrees clockwise. If you choose to tackle this additional task, use the file `` `matrix_manipulation_ec.c` ``, INSTEAD of `` `matrix_manipulation.c` ``
This includes an additional function:
`rotate_matrix`

This function is extremely similar to the others, but the solution you write will be different.

```
int** rotate_matrix (int** orig_mat) {
    int** result;
    result = alloc_matrix(result);
    int i, j;

    /*
     * insert code here
     */

    return result;
}
```

Additional Output (note this will be handled for you):

```
Rotate 90* (cw) Matrix:
  55   6  23  29  77  31  85  70   1  98
  62  32  15  97  67  68  38   8  13  63
   1  38  25  88  32   1  33  83  81  85
  54  13  74  44  36  13  88  20   9  83
  89  66  78  29  41  80  23  36 100  97
  58  17  34  90  53  43  50  57   3  22
  91  67  35  66  86  80  38  34  43  23
  20  38  49  72  21  25  10  90  86   1
  76  62  22  60   9   7  18   1  51  58
  18  96  50  79  49   3  91  90  80  21
[...]
Matrix rotate is correct.
```

`Deliverables:`

If **not** attempting extra credit:

Submit ``matrix_manipulation.c`` to WebCourses by 4/4.

If attempting extra credit:

Submit ``matrix_manipulation_ec.c`` to WebCourses by 4/4.

Do not submit both files. Only one or the other.
Do not submit `matrix_test.h`.