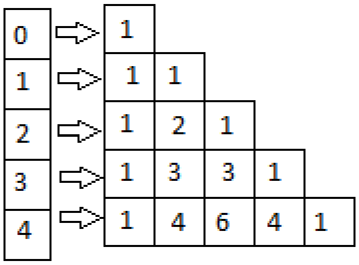
1) (10 pts) DSN (Dynamic Memory Management in C)

Using 0-based indexing, on row i of Pascal's Triangle, there are i+1 positive integer values. One way we can efficiently store the triangle is to allocate the correct amount of memory for each row. Here is a picture of the first five rows of the triangle (rows 0 through 4, inclusive.):



If the name of the array is **tri**, then the rule to fill in the entries in the table are as follows:

```
tri[i][0] = 1, for all positive ints i
tri[i][i] = 1, for all positive ints i
tri[i][j] = tri[i-1][j-1]+tri[i-1][j], for all ints j, 0 < j < i
```

Write a function that takes in an integer, ***n***, dynamically allocates an array of ***n*** arrays, where the ith array (0-based) is allocated to store exactly i+1 ints, fills the contents of the array with the corresponding values of Pascal's Triangle as designated above, and returns a pointer to the array of arrays. **You may assume that 1 < n < 31.**

```
int** getPascalsTriangle(int n) {


    int** tri = malloc(sizeof(int*)*n);                 // 2 pts

    for (int i=0; i<n; i++) {                           // 1 pt

        tri[i] = malloc(sizeof(int)*(i+1));             // 2 pts
        tri[i][0] = tri[i][i] = 1;                      // 1 pt

        for (int j=1; j<i; j++)                         // 1 pt
            tri[i][j] = tri[i-1][j-1] + tri[i-1][j];    // 2 pts
    }

    return tri;                                         // 1 pt


}
```

Grading Notes: Take off an integer number of points. For two small errors that you believe are each worth less than a point, take off 1 pt total. It there's only one tiny error (say one dot instead of arrow) correct it and give full credit.