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

Consider allocating space for an array of arrays, where each of the individual lengths of the different small arrays may differ. For example, we might want 5 arrays, which have lengths 10, 5, 20, 100 and 50, respectively. Write a function `makeArray` that takes in an array of integers itself and the length of that array (so for the example above the first parameter would be the array storing 10, 5, 20, 100 and 50 and the second parameter would have a value of 5) and allocates space for an array of arrays where each of the individual arrays have the lengths specified by the values of the input array. Before returning a pointer to the array of arrays, the function should store 0 in every element of every array allocated.

```
int** makeArray(int* lengths, int numarrays) {

    int** array = malloc(sizeof(int*)*numarrays);
    int i;
    for (i=0; i<numarrays; i++)
        array[i] = calloc(lengths[i], sizeof(int));
    return array;

}
```

Grading:

3 pts initial malloc, (1 pt for **, 1 pt for sizeof(int*), 1 pt *numarrays) could use calloc as well.

2 pts for loop

4 pts for allocating each array (2 pts) and zeroing it out (2 pts), obviously calloc does this faster, but you can also malloc and run a loop inside a loop…

1 pt for the return