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

The struct Monster_List maintains a list of monsters using a dynamically sized array of pointers to Monster. A function prototype is given for a function initializeMonster, which takes in a pointer to a Monster **that must already be pointing to memory that is allocated**, and then fills that memory with information about a default monster. Write a function getDefaultMonsters which takes in a positive integer n, creates a pointer to a Monster_List, allocates room for it, and then fills it with n default Monsters, and then returns a pointer to the Monster_list created. (Note: You must call initializeMonster in your solution.)

```
typedef struct Monster {
    // Details not necessary to solve the problem.
} Monster;

typedef struct Monster_List {
    Monster** mArray;
    int numMonsters;
} Monster_List;

// Initializes the monster pointed to by mPtr to be the default
// monster.
void initializeMonster(Monster* mPtr);

Monster_List* getDefaultMonsters(int n) {

    Monster_List* res = malloc(sizeof(Monster_List));  // 2 pts

    res->mArray = malloc(sizeof(Monster*)*n);     // 2 pts

    res->numMonsters = n;                         // 1 pt


    for (int i=0; i<n; i++) {                     // 1 pt
        res->mArray[i] = malloc(sizeof(Monster)); // 2 pts
        initializeMonster(res->mArray[i]);        // 1 pt
    }
    return res;                                   // 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.