MCS 360 Exercise Set #1
(to be turned in Tuesday, Sep 11, at the discussion section)

1. Given the declarations

```
double a[9] = {0, 1, 4, 9, 16, 25, 36, 49, 64};
double *p = &a[2], *q = &a[5];
```

find the value of each expression below.   Assume that a pointer occupies 4 bytes and a double occupies 8 bytes.

a) `*(p + 4)`

b) `q[2]`

c) `q - p`

d) `*q - *p`

e) `(unsigned)q - (unsigned)p`

f) `sizeof(a) - sizeof(p)`

g) `*a`

h) `*(q - 4)`

2. Given the declarations

```
int b[3][4] = {10, 20, 30, 40,
               55, 45, 35, 25,
               21, 18, 15, 12};
int *r = &b[1][3], *s = &b[2][1];
```

find the value of each expression below.

a) `r - s`

b) `*r - *s`

c) `*(s + 2)`

d) `*(*(b+2) + 1)`

e) `s - b[2]`

3. Write a C language function

```
double *new_d_array( int size, double val);
```

that will create a new dynamic array of size `size` and element type `double`, in which each entry is initialized to `val`. Your function should return the address of the array. If memory is not available, terminate with an error message. You may invoke `checked_malloc()`.

4. Write a C language function

```
int **pascal_triangle( int n)
```

that will create a new 2-dimensional *dynamic* array with element type `int` that is triangular. The array will have `n+1` rows, and row `i` will contain `i+1` elements, for `i` = 0, 1, ..., n. Element `i,j` of the array should be initialized to the binomial coefficient $\binom{i}{j}$; please see below.

For example, if $n = 8$, the array would look like this.

```
1
1   1
1   2   1
1   3   3   1
1   4   6   4   1
1   5   10  10  5   1
1   6   15  20  15  6   1
1   7   21  35  35  21  7   1
1   8   28  56  70  56  28  8   1
```
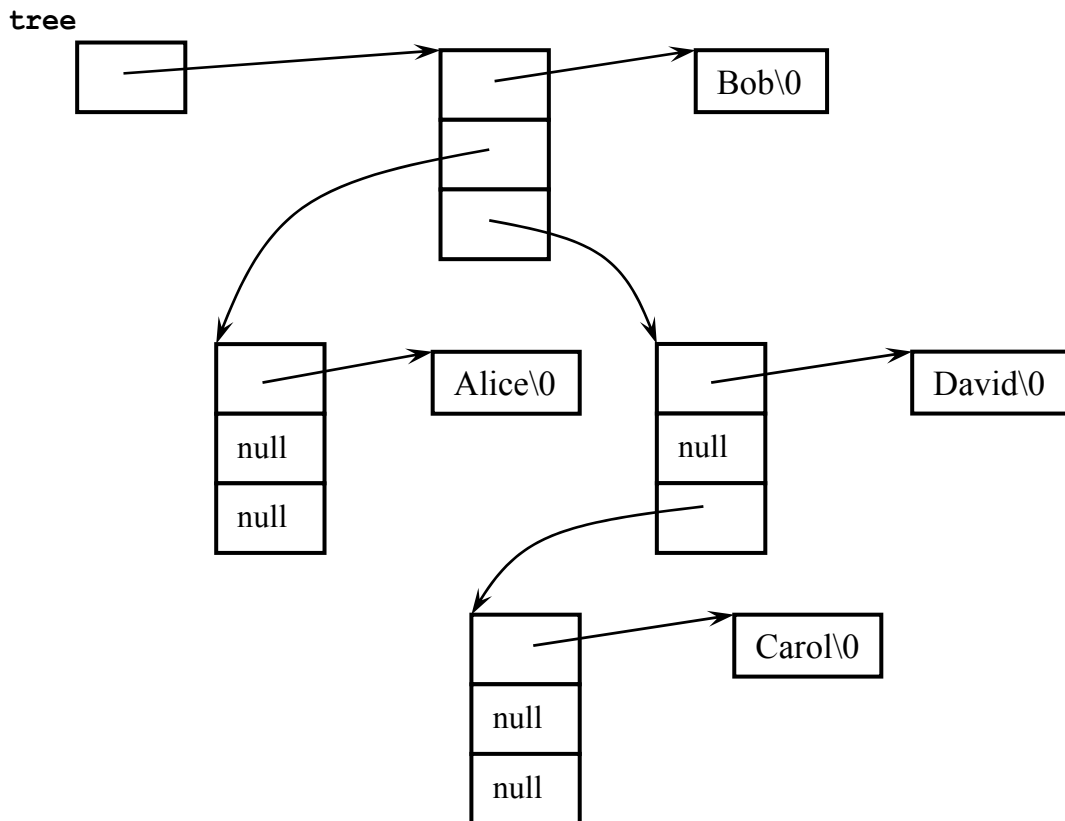
Note that, except in row 0, each entry is the sum of the entry straight above and the entry above and to the left. A missing entry is treated as 0.

Your function should return the address of the array. If memory is not available, terminate with an error message. You may invoke `checked_malloc()`.

5. Given the declaration

```
struct BTNode {
    char *name;
    struct BTNode *left;
    struct BTNode *right;
}
typedef struct BTNode BTNode;
BTNode *tree;
```

write code to allocate and initialize memory as shown in the diagram below. Apart from `tree`, everything is in dynamic memory. You may use `checked_malloc()`.

6. Without introducing any new variables, write code to change "Carol" to "Cindy" in the configuration created in problem 5.

7. Given the declarations

```
struct Name {
    char *first_name;
    char middle_initial;
    char *last_name;
};
typedef struct Name Name;

struct Person {
    Name name;
    int year_born;
};
typedef struct Person Person;

struct Book {
    char *title;
    Person *firstAuthor;
    int year_published;
};
typedef struct Book Book;

Book *textbook;
```

Write C language code to allocate a `Book` to which textbook points and initialize the `Book` to the textbook for this course (see course information sheet on the website; you may leave the `year_born` field of `Person` uninitialized). Note initializing a `Book` will require additional allocations of dynamic memory.