CSE 333 – Spring 2013
Section #2 – 4/10/13
Not a quiz! Will not count towards your grade! For learning purposes only!

Suppose we have the following `struct` definition:

```
typedef struct _Foo {
    int baz;
    int qux;
} Foo;
```

Write C statement(s) that would:

- Statically allocate a Foo

- Dynamically allocate a Foo

- Statically allocate an array of Foos

- Dynamically allocate an array of Foos

What does the following statement do? Describe and draw a diagram.

```
Foo **bar = malloc(16 * sizeof(Foo*));
```

What is the type of `**bar`? What about `bar[4]`? What do each represent?

Write a function `makefoo` that creates an array of Foo pointers (not just Foos!). Each pointer should point to a valid Foo structure, and each Foo must be initialized with `baz` being the index of that Foo structure (or its pointer) in the array, and `qux` being set to the value 333.

The function must take as parameters an integer representing the size of the array and a pointer with the correct amount of indirection (number of stars) so that the array can be passed back through this parameter. The function must also return whether it was successful or not.

The function declaration should thus look similar to the following, with the question marks replaced with the appropriate type:

```
bool makefoo(int n, ???? array_out);
```

Given the following definitions, draw a diagram including at least `main_table` and `record_x`. Make up any additional values if you feel compelled.

```
typedef struct _XRef {
    int rec_id;
    char* label;
    struct _XRef ** parent_table; // array of XRef pointers
    int table_index;
    struct _XRef * cross_ref;
} XRef;
...
XRef ** main_table = malloc(5 * sizeof(XRef*));
... (Allocate space for each XRef)
... (Initializing values)
XRef * record_x = main_table[3];
record_x->rec_id = 333;
strncpy(record_x->label, "perkhal", LABEL_LEN);
record_x->parent_table = main_table;
record_x->table_index = 3;
record_x->cross_ref = main_table[1];
... (Initialize remaining values)
```

Assuming `LABEL_LEN` is already properly defined, give an example snippet of code that would allocate space for each `XRef` in the code above.

Look at the following struct and function definitions. Describe what the struct is and what the function does as well as you can.

```
typedef struct _Myst {
    uint64_t thing0;
    struct _Myst * thing1;
    struct _Myst * thing2;
} Myst;

void function(Myst *m, uint64_t u) {
    Myst *p = NULL;
    while (m) {
        p = m;
        if (u > m->thing0)
            m = m->thing2;
        else
            m = m->thing1;
    }
    m = malloc(sizeof(Myst));
    if (u > p->thing0)
        p->thing2 = m;
    else
        p->thing1 = m;
}
```