CS 161
Intro to CS I

Arrays vs. Structs
What about passing structs to functions?

```cpp
#include <iostream>

using namespace std;

// This defines the user-defined type of doc_rec that contains three pieces of info of mixed types
struct doc_rec {
    char name[25];
    int age;
    float weight;
};

// We can return a doc_rec struct from a function
doc_rec set_info()
{
    doc_rec person;

    cout << "Enter name: ";
    cin >> person.name;
    cout << "Enter age: ";
    cin >> person.age;
    cout << "Enter weight: ";
    cin >> person.weight;

    return person;
}```
What about passing structs to functions?

```c++
24 } //We can pass the address of (C style) or by reference (C++ style)
25 //to change the contents of the structure austin in main
26 void set_info(doc_rec *person) {
27     cout << "Enter name: " ;
28     cin >> person->name; //same as (*person).name for dereference
29     cout << "Enter age: ";
30     cin >> person->age;
31     cout << "Enter weight: ";
32     cin >> person->weight;
33 }
34 int main() {
35     struct doc_rec austin; //create a doc_rec structure called austin
36     //austin = set_info(); //capture returned info into structure
37     set_info(&austin); //pass address of to change contents
38     cout << austin.name << endl;
39     cout << austin.age << endl;
40     cout << austin.weight << endl;
41     return 0;
42 }
```
Array of Structs...

struct contact_info {
    std::string name;
    std::string address;
    unsigned int phone;
};

...  
int main() {
    contact_info address_book[50];
    ...
    address_book[0] = create_contact();
    ...
}

contact_info create_contact() {
    contact_info contact;
    contact.name = “Jennifer”;  
    return contact;
}
Dynamic Array of Structs

```cpp
struct contact_info {
    std::string name;
    std::string address;
    unsigned int phone;
};

int main() {
    contact_info *address_book;
    cin >> n;
    address_book = new contact_info[n];
    address_book[0] = create_contact();
    delete [] address_book;
}

contact_info create_contact() {
    contact_info contact;
    contact.name = "Jennifer";
    return contact;
}
```
Demo Array of Structs, structs.cpp
struct address_book {
    std::string *name;
};

... int main() {
    address_book contacts;
    cin >> n;
    contacts.name = new string[n];
    contacts.name[0] = "Jennifer";
    delete [] contacts.name;
    ...
}
Demo Struct with Dynamic Array
Exercise #8

- Get into groups of 3-5!!!
- Make `int ****p;` point to `int i;`, using only the heap!
- Create a dynamic struct inside a function:
  ```
  struct book {
      string title;
      unsigned short year;
  }
  
  void create_struct(book **b);
  ```

Write the `main()` and `create_struct()` function, which asks a user if they want to create a book, creates the struct if needed, and sets the title and year.