1. Here is the beginning of a program involving structs. You will need to fill in missing bits. If you can work with a partner with a machine and actually compile your program at each step, do that. If not, work on paper.

```c
#define MAX_POSITION_SIZE 16
#include <stdio.h>
#include <string.h>
#include <stdlib.h>

struct player {
    char *name;
    char position[MAX_POSITION_SIZE];
    int home_runs;
    float avg;
};

int main() {
    // Declare a struct player called p1.
    struct player p1;
    // Initialize it to represent the third baseman Josh Donaldson,
    // whose batting average is 0.270. He has hit 33 home runs.
}
```

2. Here we have added a declaration for a pointer to a struct player. Allocate space for the struct on the heap and have `p2` point to that memory.

**Error-checking:** Look at the man page for `malloc` to see what it returns if it is unable to allocate memory. Now add code to check if the memory allocation for `p2` was successful, and if it failed, exit the program with a non-zero exit status.

```c
struct player *p2;

// Set the values of p2 to represent shortstop Troy Tulowitzki. His batting average is .249
// and he has hit 7 home runs.
```
3. Write a function `out_of_the_park` that increments the home-run count for the player passed as the function’s argument. Think carefully about what the type of the function parameter should be.

4. Show how to make calls to `out_of_the_park` using `p1` and `p2`.

5. Suppose we have the following function declaration.
   
   ```c
   void f(struct player p) { // Body hidden }
   ```
   
   Now suppose we call it from `main` using `f(p1)`. Draw the memory diagram of the program immediately after `f` is called, but before it starts executing. For extra practice, include `p2` and related memory in your diagram.

6. Something to think carefully about: can the body of `f` affect the local `p1` of `main`? In other words, after `f(p1)` exits, can any data associated with `p1` have changed?

7. On a new sheet of paper, repeat the previous two questions when you call `f(*p2)` instead.