

16.216: ECE Application Programming

Summer 2012

Lecture 11: Key Questions

August 14, 2012

1. Describe what a structure is in C, and how structures can be useful.
2. Explain how we can essentially declare our own types using structures.

3. Show how variables of a given structure type can be declared and initialized.

4. Show how elements within a structure can be accessed.

5. **Example:** What does the following program print?

```
#include <stdio.h>

typedef struct {
    double real;
    double imag;
} Complex;

int main() {
    Complex a = {1, 2};
    Complex b = {3.4, 5.6};
    Complex c, d, e;

    printf("A = %.2lf + %.2lfi\n", a.real, a.imag);
    printf("B = %.2lf + %.2lfi\n", b.real, b.imag);

    c = a;
    d.real = a.real + b.real;
    d.imag = a.imag + b.imag;
    e.real = a.real - b.real;
    e.imag = a.imag - b.imag;

    printf("C = %.2lf + %.2lfi\n", c.real, c.imag);
    printf("D = %.2lf + %.2lfi\n", d.real, d.imag);
    printf("E = %.2lf + %.2lfi\n", e.real, e.imag);

    return 0;
}
```

1. Explain how pointers are used to access structure variables.
2. Explain how structures are passed to and returned from functions.

3. **Example:** Write the following functions that use the `StudentInfo` structure
- Given a pointer to a single `StudentInfo` variable, print all of the student info to the screen using the following format:
 - Michael J. Geiger
 - ID #12345678
 - GPA: 1.23

 - Given an array of `StudentInfo` variables, compute and return the average GPA of all students in the list

- Prompt the user to enter 3 lines of input (using the format below), read the appropriate values into `StudentInfo` elements, and return a value of type `StudentInfo`
 - Format (user input underlined)
 - Enter name: Michael J. Geiger
 - Enter ID #: 12345678
 - Enter GPA: 1.23