16.216: ECE Application Programming

Fall 2015

Lecture 30: Key Questions
November 18, 2015

1. Describe how to print hexadecimal values.

2. Describe what a structure is in C, and how structures can be useful.

3. Explain how we can essentially declare our own types using structures.

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

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

6. **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;
}
```

7. Explain how pointers are used to access structure variables.

8. Explain how structures are passed to and returned from functions.

9. **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`