

CSE 333 – SECTION 1

C Review and problems

A bit about us

- James Okada
 - Undergrad, CSE.
 - Contact: jyo2@uw.edu jyo2@cs.washington.edu
 - Office hours: TBD
- Renshu Gu
 - PhD student, Department of Electrical Engineering.
 - Contact: renshugu@uw.edu
 - renshugu@u.washington.edu
 - Office hours: TBD

Sections Format

- Some lecture material/discussion of projects.
- Try to go through examples each week pertaining to the exercise/project and material learned in class
- We're likely to do exercises in section. On two or three instances. They will be graded as a quiz, but mostly they won't. We will let you know which day those quizzes will be by marking them prominently on the calendar.

Ex0/hw0

- Some suggestions for exercises
 - “Good style” for this class is based on the Google Style guide, so follow it when in doubt, later on use `clint`, `cpplint`
 - Keep it short and simple— dense code with a few comments sprinkled in
- Get in to the habit of using man pages.
- Expect exercise grades/feedback prior to the next lecture after turning them in (no promises!)

Structs

- Used for encapsulating data
- Can contain primitive types (int, double, etc.), arrays, other structs, and unions, among other types
- Accesses are made through the ‘->’ operator for pointers to structs and ‘.’ for values.
- More on this later;.

Structs

- Example

```
struct Sample {  
    int a, b;  
};
```

```
int main(int argc, char* argv[]) {  
    struct Sample s;  
    s.a = 10;  
    s.b = 5;  
    struct Sample *s_ptr = &s;  
    printf("s.a is %d and s.b is %d\n", s.a, s.b);  
    printf("s_ptr->a is %d and s_ptr->b is %d\n", s.a, s.b);  
    return 0;  
}
```

Arrays

- Just a block of data of a particular type and size
- Raw pointers can be treated as arrays and vice versa, with some minor caveats
- Pointer variables can be treated as arrays but, don't forget to allocate space for the array!

```
int* a = (int*) malloc(sizeof(int) * 3);
int* b = (int*) malloc(sizeof(int));
int c[5] = {0}; // stack allocated array
a[2] = 6;
b[0] = 4;
c[2] = 2;
*a = c[2]; // what does this do?
free(a);
free(b);
```

Quick Example!

- Lets do a quick example to recap what we learned!