> echo $USER
Lawrence_Angrave

> cat "CS241 Learning Objectives.txt".
You will be able to ...

- Interact with OS in C via system calls
- Understand how OS allocate, deallocates and accesses memory
- Understand how virtual memory works
- Create, use, manipulate processes and threads
- Understand how OS schedules processes and threads
- Communicate and synchronize between threads and processes
- Determine when deadlock and race conditions may occur and how to avoid them
- Manipulate filesystem structures (inodes etc.)
- Communicate across networks

> man -S 2 " The Experience CS241"

Not your regular course. This is a UIUC-and-by-Angrave course. A byte of CS241 every day is good for you.
Class: Lecture MWF. Thursday Section. CBTF Examlets.

> Grades

> Why do we need an O/S ?

> Program vs Process

> Fun stuff:
Low level! UIUC programmers don't just program in python/js, they could *write* python/js
Powerful! Create things that others will use. Make programs that others can only dream of.

> Master...

- Know your tools! C Programming / System programming is brutal if you don't know the details.
- Concurrency (muli-threading, multi-process)
- Synchronization
- Signals
- Critical Section
- Race Conditions
- Deadlock
- Analysis of Reader-Writer, Dining Philosphers, Producer Consumer

> Process memory

- Environment
- Program Arguments
- Stack
- Heap
- Unitialized vars
- Initialized vars
- Code

+ Dynamically linked library functions + Guard pages + Multiple threads...

-1. Pointers hold a memory address. (useful...)

```
char *ptr1;
char* * * * *ptr2; // is just another pointer.

ptr =

*ptr =
```

0. Spot the difference

```
    char* a = "Arghhh";
    char b[] = "Pieces of 8";

    *b = 0;
    *a = 0;
```

1. c library vs system calls.

```
    printf("Hello %d",cs241);
    puts("World");

    const char*ptr = "World\n");

    write( 1, ____ , _______ );
//write(int fildes, const void *buf, size_t nbyte);
```

2. Truncate a string to four letters.

```
char[] mesg = "Once upon";
?_____

printf("%d:%s, strlen(mesg), mesg); // Prints 4:Once
```

3. Implement strcpy (copies a C string from src to dst)

```
char * strcpy(char * dst, const char * src){







}
```

4. Implement strdup (create a copy of the string in heap memory)?

```
char * strdup(const char * src){







}
```

5. Questions!

Your turn: Honors course?

cs241.cs.illinois.edu | Navigate to the wiki HW0 ; bring to your lab tomorrow.