

# stdio.h, varargs, unistd.h

3 lectures left!!

---

CS 2130: Computer Systems and Organization 1

November 30, 2022

# Announcements

- Homework 8-10 posted, due last day of class at 11pm
  - Homework 10 is **optional** - replaces lowest homework grade
- There **will** be an optional Quiz 9 this weekend - replaces lowest quiz grade
- Final Exam: December 15 at 7-10pm
  - Cumulative, see practice tests
  - Please schedule with SDAC if needed

# printf

```
int printf(const char *format, ...);  
int fprintf(FILE *stream, const char *format, ...);
```

# printf

```
int printf(const char *format, ...);
```

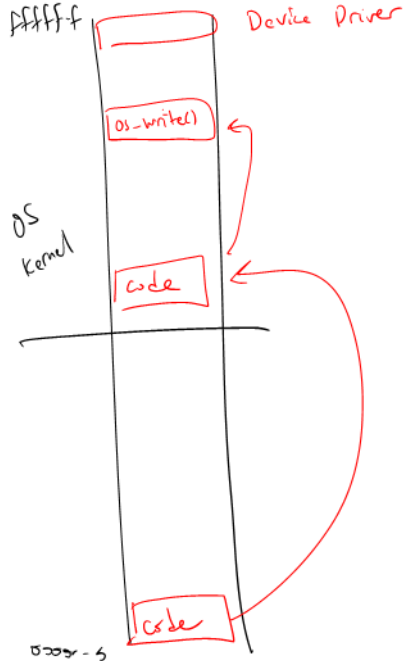
```
printf("hi: %s and %d\n", mystr, myint);
```

The diagram illustrates the argument passing mechanism in the printf function. It shows the format string "hi: %s and %d\n" and the arguments mystr and myint. Blue brackets are drawn under "hi:", "%s", and "%d\n" in the format string. Arrows point from these brackets to the arguments: one arrow from the "%s" bracket to mystr, and another from the "%d\n" bracket to myint. A double underline is drawn under "hi:".

move  
copy (str → FILE\*)

Backing up...

# Syscalls



Write :

- [ argument checking
- syscall
- [ return value checking
- ret

`write:`

- Argument checking
- `syscall`
- Return value checking
- `ret`

# Processes

Process - approximately what we think of as a “running program”

- Operating System effectively has a giant array of processes started since computer turned on
- Try `ps -A`
- Has access to all memory (but only its own!)
- Operating System maintains data structure about each process
  - What program is running, who ran it, when it started, ...
  - Array of “file like objects”



# Processes

Using write