CS 170 Week 7

Winter 2026

https://sites.cs.ucsb.edu/~rich/class/cs170/

Agenda

1. Practice exec(), fork(), wait()
    a. Board Presentations
2. Questions

Execve()

```
char* path = "argtest";
char* argv[10] = {"argtest", "A", "cat", "sat", "on", "a", "mat",
    NULL };
printf("printf 0x%x, 0x%x %s\n", argv, argv[0], argv[0]);
execve(path, argv);
```

- What is the type of path?
  - How about proc->registers[5]?
  - Argv?
  - proc->registers[6]?
- Let main_memory start at 500. Proc->base = 40. Proc->registers[5] = 80.
  - Respective to the user, what is the value of path?
  - Respective to the OS, where is the value of path located in main_memory array? KOS memory?
- Let proc->registers[6] = 72.
  - What is the value of argv?
  - Suppose argv[2] = 96. Where is “cat” located in main_memory array? KOS memory?

fork() - Find the errors.

```c
void syscall_return(pcb* proc, int ret) {
    proc->reg[PCReg] = proc->reg[NextPCReg];
    proc->reg[2] = ret;
    add_to_ready_queue(proc);
    kt_exit();
}
void do_fork(pcb *proc) {
    pcb* child = (pcb*)malloc(sizeof(pcb));

    child->pid = proc->pid;
    child->base = proc->base;
    child->limit = proc->limit;
    for(int i = 0; i < NumTotalRegs; i++) {
        child->registers[i] = proc->registers[i];
    }

    memcpy(child->base, proc->base, proc->limit);

    syscall_return(child, 0);
    syscall_return(proc, child->pid);
}
```

Wait() Find Errors

```
void do_exit(pcb *proc) {
    ...
    add(proc->parent->exited_children, proc);
    V_kt_sem(proc->parent->wait_sem);
    ...
}
void do_wait(pcb *proc) {

    P_kt_sem(proc->wait_sem);

    pcb* child = first_item(pcb->exited_children);

    ret_val = proc->pid;

    int status = proc->registers[5];
    if(status > 0){
        memcpy(&main_memory[status], &child->exit_status, sizeof(int));
    }
    FreePCB(child);
    syscall_return(proc, ret_val);
}

```

Questions