Principles of Operating Systems **Name (Print):**

Fall 2018
Final
12/13/2018
Time Limit: 1:30pm – 3:30pm

- **Don't forget to write your name on this exam.**
- **This is an open book, open notes exam. But no online or in-class chatting.**
- **Ask us if something is confusing.**
- **Organize your work**, in a reasonably neat and coherent way, in the space provided. Work scattered all over the page without a clear ordering will receive very little credit.
- **Mysterious or unsupported answers will not receive full credit**. A correct answer, unsupported by explanation will receive no credit; an incorrect answer supported by substantially correct explanations might still receive partial credit.
- If you need more space, use the back of the pages; clearly indicate when you have done this.
- **Don't forget to write your name on this exam.**

Problem	Points	Score
1	15	
2	5	
3	20	
4	5	
5	10	
6	5	
Total:	60	

1. Pipes

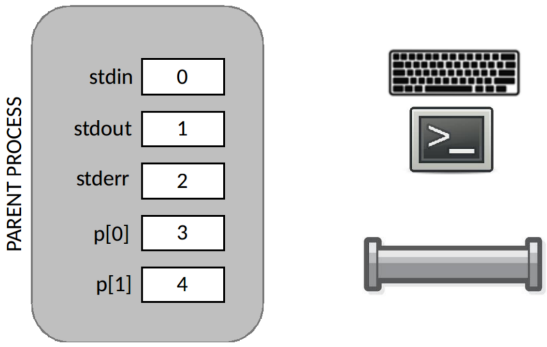
   Xv6 shell implements a pipe command (e.g., `ls | wc`) with the following code:

```
8650   case PIPE:
8651     pcmd = (struct pipecmd*)cmd;
8652     if(pipe(p) < 0)
8653       panic("pipe");
         // Point A
8654     if(fork1() == 0){
8655       close(1);
8656       dup(p[1]);
8657       close(p[0]);
8658       close(p[1]);
           // point B
8659       runcmd(pcmd->left);
8660     }
8661     if(fork1() == 0){
8662       close(0);
8663       dup(p[0]);
8664       close(p[0]);
8665       close(p[1]);
8666       runcmd(pcmd->right);
8667     }
8668     close(p[0]);
8669     close(p[1]);
         // point C
8670     wait();
8671     wait();
8672     break
```

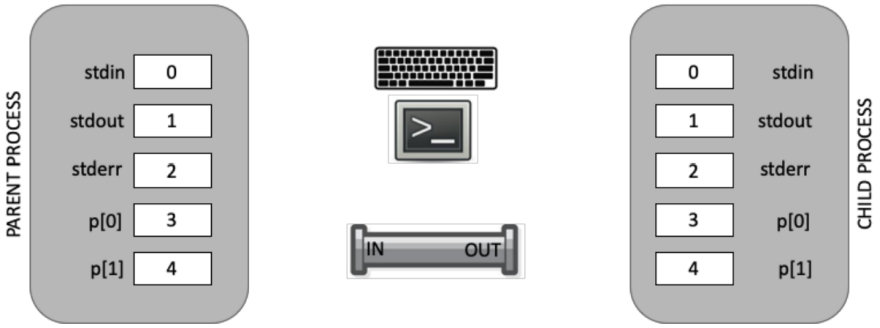
   Draw the connections between file descriptors, I/O devices and pipes at points A, B above. Connections can be depicted with lines with arrows. The error is aligned with the direction of data flow, i.e., if the file is written the error points at the file object.

   Hint: pay attention to close() dup() calls before and after the point

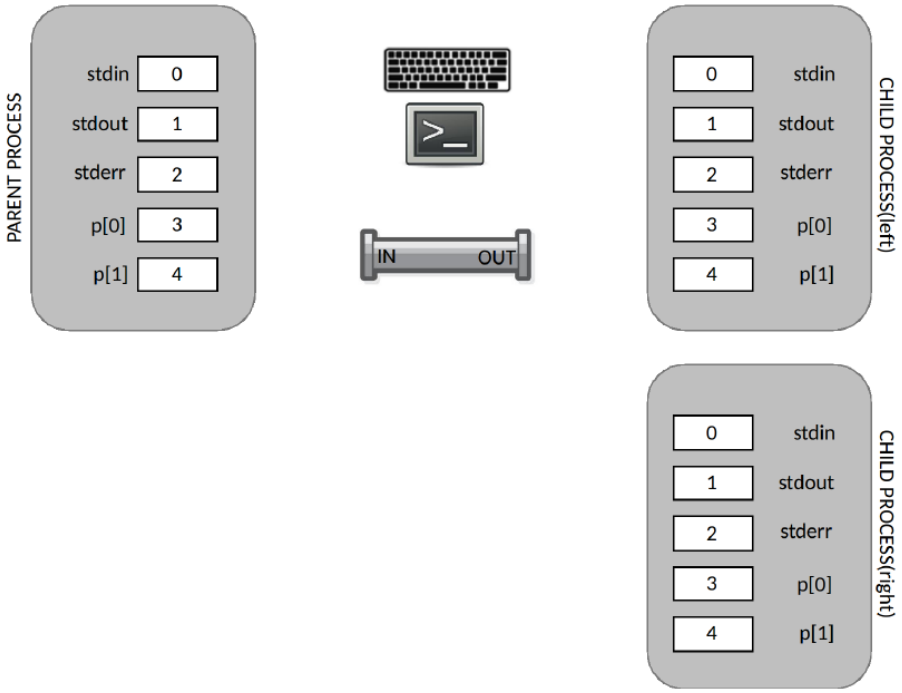
(a) (5 points) Point A



(b) (5 points) Point B



(c) (5 points) Point C



2. Processes and system calls

(a) (5 points) What is the first system call executed by xv6? Explain your answer.

3. Interrupts and context switch

   (a) (5 points) When a user-program (a program that executes at current privilege level 3) is preempted with an interrupt five registers are saved by the hardware: ESP, SS, EFLAGS, CS, EIP. Why these five registers have to be saved, but others, e.g., EAX, ECX, etc., don't?

   (b) (5 points) During the context switch the code of the `swtch()` function visibly does not save the EIP register. How is it saved and restored then during the context switch?

(c) (5 points) The `fork()` system call returns "0" inside the child process. This return value is passed to the child process from inside the `fork()` system call with the following line:

```
np->tf->eax = 0
```

Explain how does this work, i.e., how the "0" value ends up being returned by the fork() inside the user process.

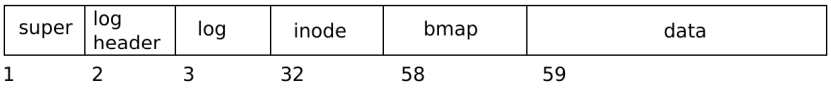
(d) (5 points) What does the stack look inside the `bar()` function. Draw a diagram, provide a short description for every value on the stack.

```
int bar(int a, void *buffer, int size) {


};
```

4. File system

   Xv6 lays out the file system on disk as follows:



   Block 1 contains the super block. Blocks 2 through 31 contain the log header and the log. Blocks 32 through 57 contain inodes. Block 58 contains the bitmap of free blocks. Blocks 59 through the end of the disk contain data blocks.

   (a) (5 points) Every file system transaction that changes the file system write one disk block twice. What is this block (what's its block number) and why is it written twice?

5. Synchronization

   (a) (5 points) Sleep has to check `lk != &ptable.lock` to avoid a deadlock. Suppose the special case when the following lines

   ```
   if(lk != &ptable.lock) {
     acquire(&ptable.lock);
     release(lk);
   }
   ```

   are replaced with

   ```
   release(lk);
   acquire(&ptable.lock);
   ```

   Doing this would break sleep. How?

   (b) (5 points) Now Alice decides to put the following code instead of the original `xchg()` loop in the `acquire()` function

   ```
   for(;;) {
     if(!lk->locked)
     {
       lk->locked = 1;
       break;
     }
   }
   ```

   She boots xv6 on a multi-processor machine, explain what happens?

6. cs143A. I would like to hear your opinions about cs143A, so please answer the following questions. (Any answer, except no answer, will receive full credit.)

   (a) (1 point) Grade cs143A on a scale of 0 (worst) to 10 (best)?

   (b) (2 points) Any suggestions for how to improve cs143A?

   (c) (1 point) What is the best aspect of cs143A?

   (d) (1 point) What is the worst aspect of cs143A?