CSSE 332 -- Operating Systems

Pipes

Name: _____

Question 1. (5 points) In Unix, "unnamed" pipes are **bidirectional** means of communication that are managed by the kernel.

A. True.

B. False.

Question 2. Consider the processes with the lineage relationship shown in Figure 1 below.

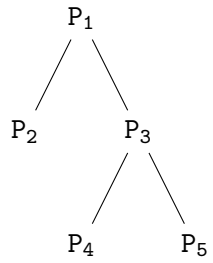

Figure 1: Lineage relationship for processes in **Question 2**

(a) Assume that before forking `P4` and `P5`, `P3` creates a Unix pipe (let's call it σ) using the `pipe` system call.

i. (5 points) `P4` and `P5` can communicate with each using σ.

A. True.

B. False.

ii. (5 points) `P4` cannot use σ to communicate with `P3`.

A. True.

B. False.

iii. (5 points) `P4` can use σ to communicate with `P1`.

A. True.

B. False.

(b) (5 points) Assume now that `P2` creates a pipe using the system call `pipe`, which of the below processes can `P2` communicate with using that pipe?

A. `P1` B. `P3` C. `P4` D. `P5` E. None of the above.

Question 3. (5 points) A process that needs to **read** from a pipe must _____ the _____ of that pipe. It can then use the _____ system call to extract bytes from the pipe.

Question 4. (5 points) Briefly describe the events that happen when a process attempts to read from a pipe that has no more writers, but whose writing ends are still open.

Question 5. Consider the following code snippet.

```c
if(pipe(fd) < 0) {
  perror("PANIC");
  exit(EXIT_FAILURE);
}
int rc = fork();
if(rc == 0) {
  // sleep for some 20 seconds, give parent time to write.
  char buff[5];
  int len;
  while((len=read(fd[0], buff, 4))) {
    buff[len] = 0;
    printf("Read %s\n", buff);
  }
  close(fd[0]);
}
// close reading end
close(fd[0]);
write(fd[1], "hello world!", strlen("hello world!"));
write(fd[1], "nice try!", strlen("nice try!"));

// done
close(fd[1]);
// do other stuff and wait for child.
```

(a) (5 points) The code above contains a bug. Find it and suggest a way to fix it.

(b) (5 points) Assume now that the bug has been fixed and that **all** of the parent's write operations finish before the child process reaches the `while` loop. What would be the output on the console when the child reads from the pipe?