CIS 415 * Spring 2010 Worksheet: Fork and Exec

1. Basic Fork and Exec (assume one process running initially)

a. After execution of `pid=fork()` how many processes are executing?
b. What value is in `pid`? For the parent? For the child?
c. What line of code placed immediately after the fork(), would the parent execute to print the pid of the child?
d. What line of code placed immediately after the fork(), would the child execute to print its own pid?
e. What lines of code place immediately after the fork(), would the child execute to print its parent’s pid?

2. Simple Fork and Exec

Write pseudo-code for a parent that creates two children. One child should run the code in "/bin/A" and the second child should run the code in "/bin/B". The parent should stay alive throughout and can just run the original code but do nothing. (In real life, it might sleep until the children terminate.)
You don't need to declare variables or write accurate C code.

3. Edible Fork and Exec

(a) Draw the tree of processes that results from execution of the following code.
(b) Label the processes that are executing “/bin/chocolate” code.

```c
main() {
    int j;
    pid_t pid1, pid2;       /* two pid variables */

    for (j = 1; j <=10; j++) {
        pid1 = fork();
        if (pid1 == 0 && j is even) {
            pid2 = fork();
            if (pid2 == 0)    exec("/bin/chocolate");
            else if (pid2 > 0) exec("/bin/vanilla");
        } else if (pid1 == 0 && j is odd) exec("/bin/strawberry");
        else if (pid1 > 0) continue;  /* continue looping */
    }

    wait for all children to complete; /* only the parent will execute this code since all the descendants exec-ed off new code. */
}
```