Homework 2

CS211 - Fall 2024

(You should try to answer first and then compare your solution with the AI tool solution)

This homework is designed to help you become familiar with key concepts related to processes and the programming interface for process management in an operating system. All questions are adapted from textbooks and online sources.

Question 1: Open a terminal and install all necessary packages to ensure you can run the “ls” and “exec” commands. Then, try running the command “exec ls” and explain what you observe.

Question 2: How many processes are created if the following program is run

```
void main (int argc, char ** argv) {
    forkthem(6);
}

void forkthem (int n) {
    if (n > 0) {
        fork();
        forkthem(n-1);
    }
}
```

Question 3: How many processes are created when the program below is run, and what is the value of the variable x in each process when it finishes?

```
void main (int argc, char** argv) {
    int child = fork();
    int x = 5;

    if (child == 0) {
        x += 5;
    }
    else {
        child = fork();
        x += 10;
        if (child) {
            x += 5;
        }
    }
}
```

Question 4: Guess what is the output of the following programs. Then, compile and run them, and explain why these outputs were observed.

```
// program 1
void main() {
    int val = 5;
    int pid;
    if (pid = fork())
        wait(pid);
    val++;
    printf("%d\n", val);
}

// program 2
void main() {
    int val = 5;
    int pid;
    if (pid = fork())
        wait(pid);
    else
        exit(val);
    val++;
    printf("%d\n", val);
}

// program 3
void main() {
    fork();
    fork();
    fork();
    printf("hello\n");
    return 0;
}

// program 4
void main(){
    printf("A\n");
    fork();
    printf("B\n");
    fork();
    printf("C\n");
}
```

Question 5: Read the related documentation about execl, errno, and perror, and explain the output of the program below.

```
#include <stdio.h>
#include <unistd.h>
#include <errno.h>
int main(int argc,char *argv[]) {
    printf("Try to execute lss\n");
    execl("/bin/lls","lls",NULL);
    printf("execl returned! errno is [%d]\n",errno);
    perror("The error message is :");
    return 0;
}
```

Question 6: Use fork, the exec family, and wait/waitpid to write code that executes ls in a new process properly without creating a zombie process.