Operating Systems Design Lab
Computer Engineering Department
Spring 2023/2024
Lab 1: `fork`, `exec`, and `wait` systems calls

Objectives

1. To understand the "`forking`" process and the "`fork`" system call.
2. To understand the "`exec`" system call and its variants from the `unistd` library.
3. To understand the "`wait`" and "`waitpid`" system calls.

Prelab

1. Read chapter 3 of the textbook.
2. Read the manual pages of the following systems calls and functions.

```
pid_t fork(void);
pid_t wait(int *status);
pid_t waitpid(pid_t pid, int *status, int options);
int execl(const char *path, const char *arg, ...);
int execlp(const char *file, const char *arg, ...);
int execle(const char *path, const char *arg,..., char * const envp[]);
int execv(const char *path, char *const argv[]);
int execvp(const char *file, char *const argv[]);
int execvpe(const char *file, char *const argv[], char *const envp[]);
```

Experiment

1. Write a simple program to print the PID (process ID) and PPID (parent process ID) of the underlying process. Repeat the execution multiple times (from the shell), then answer the following questions:

   (a) Is the PID the same in all runs? Why/why not?

   (b) Is the PPID the same in all runs? Why/why not?

   Hint: use the `getpid` and `getppid` systems calls to get the PID and PPID of a process respectively.

2. Modify the program in part 1 so that the main process creates a child process, and each of the two processes prints its PID and PPID. The parent process must **wait until the child process terminates**.

   (a) Verify the parent-child relationship between the two processes.

3. Write a program that continuously asks the user to enter a command (with no arguments). When the user enters a command, the program forks a child process to execute it (using `execlp`) while it waits until the child finishes execution. The program quits when the user enters "`exit`" as a command.

4. Write a program to draw the following process tree. Verify the correctness of your solution using the PIDs and PPIDs of the generated processes.