Operating Systems
Process Management

Group of Distributed Systems
University of Minho

1 Objectives

Familiarize yourself with and use the system calls for creating and managing processes.

2 System calls

```
#include <unistd.h>     /* system calls: essential defs and decls */
#include <sys/wait.h>   /* wait*() call and related macros */

pid_t getpid(void);
pid_t getppid(void);
pid_t fork(void);
void _exit(int status);
pid_t wait(int *status);
pid_t waitPID(pid_t pid, int *status, int options);
int WIFEXITED(int status); /* macro */
int WEXITSTATUS(int status); /* macro */
```

3 Exercises

1. Implement a program that prints its process identifier and the identifier of its parent process. Prove – by invoking the command `ps` – that the parent of the process is the command interpreter you used to run it.
2. Implement a program that creates a child process. Parent and child should print their process identifier and the identifier of their parent processes. The parent process should also print the PID of its child.
3. Implement a program that creates ten child processes that should execute sequentially. For this purpose, the children can print their PID and the PID of their parent. Child processes should finish their execution with an output value equal to their order number (e.g.: first child created ends with the value `1`). The parent must print the output code of each of its children.
4. Implement a program that creates ten child processes that should execute concurrently. The parent must print the output code of each of its children.
5. We want to determine the existence of a given integer in the rows of a matrix of integers, where the number of columns is much larger than the number of rows. Implement, using processes, a program that determines the existence of a given number, received as an argument, in a randomly generated matrix.

6. Based on the scenario described in the previous exercise, adapt it to print in ascending order the row numbers where there are occurrences of the number being searched for.

4 Extra exercises

1. Implement a new version of the program made in the previous exercises that operates on a matrix persisted in a file (in binary format). The matrix must be randomly generated by the parent process and written to the file at the start of the program's execution. After this step it should be possible to search for the existence of a given number using multiple processes.