Lebanese University
Faculty of Sciences I

18 April 2013
Duration : 1 hour

INFO 324
Operating System II

Problem I — 10 points

1. Draw the tree of processes created by the following program :

```
void main(){
      if(fork() || (fork() && fork()))
            fork();
}
```

2. Display the output of the following program :

```
void main(){
      if(fork()){
            if(fork()){
                  printf("A\n");
                  wait(0);
                  printf("B\n");
            }
            else{
                  printf("C\n");
            }
      }
      else{
            printf("D\n");
      }
}
```

Problem II — 20 points

1. Write a program that creates N child processes. The processes follow the following rules:
   - The child process with even PID execute the function FP()
   - The child process with odd PID execute the function FI().
   - The parent must wait the termination of all children before exit.

   You are not requested to write the functions FP() et FI().

2. Now, suppose that all child processes shared a file and the child (odd and even) use the functions FP() and FI() for accessing the file. We assume that in a given instant, one process (even or odd) can write in the file (the other processes, wanting to write to the file, must be blocked until the end of the current writing process).
   Using the pipes of communications, add few lines to the code of part (1) in order to achieve the above described synchronization.

3. In this part, we suppose that the even children are writers and the odd children are readers. If a writer process is writing to a file using FP(), we must prevent all other processes to access the file. In contrast, several readers can read simultaneously the file using FI() without any writer. Using pipes perform the described synchronization.