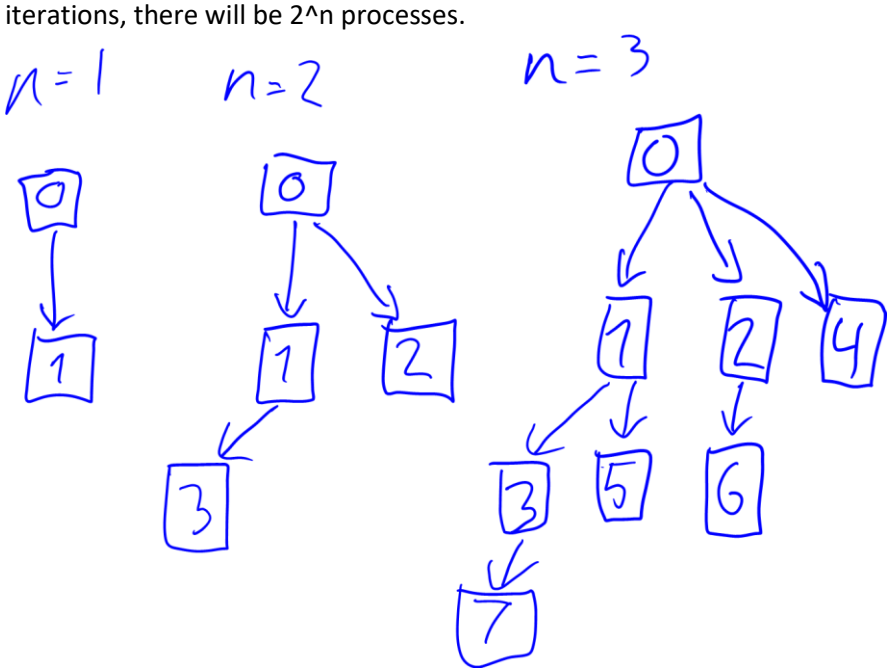
2.1 a) A child process of init is either a getty process which has gained access to the terminal login provided by init, or a process is a descendant from the getty process, but has lost all of its ancestors up to the init, so that the init process “takes over custody”

b) The exec() commands can replace the instructions to a process. This is for example useful when we fork a process but want it to do something completetley different. The execvp() in specific takes in a array of pointers to an assigned prosess, and a second array which is the new process. Then it overwrites the original prosess with its new process.

c) ls sends a list of all files and directories through the pipeline to the grep command. The grep command gets the list from ls, and does a counting operation -c on all files that end in .pdf. So it prints out the number of pdf files in the current directory.

2.2 a) Infinite recursive call on fork() so each node make a new child on each iteration. After N iterations, there will be 2^n processes.




b) The biggest challenge with this is that each process requires its own process addresses. This can be solved by having a copy on write policy, where we keep all the new processes pointing to the original as long as they are exactly the same. (Of course after a short while we will only have pointers pointing to all the new 2^n processes and we will no longer have sufficient memory to add another pointer).