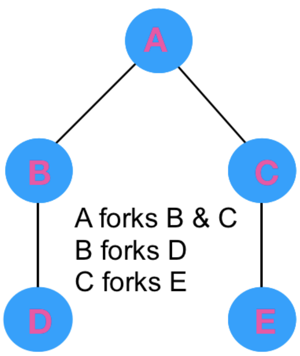
Tutorial Apr 5th
===========

Q1) Write code that create the following process tree:



Answer:

```
#include<stdio.h>
#include<unistd.h>
#include<sys/types.h>

int main(){

pid_t pid1, pid2, pid3, pid4;
printf("Parent of all: %d\n",getpid());

pid1 = fork();

if(pid1 == 0){    // A child Process. Lets say B.
    pid2 = fork();
    if(pid2 == 0){ // A child process. Lets say D.
    }
}
if(pid1 > 0){
    pid3 = fork();
    if(pid3 == 0){ // A child process. Lets say C.
        pid4 = fork();
        if(pid4 == 0){ // A child process. Lets say E.
        }
    }
}}
```

Q2) 9.9
Consider a logical address space of 256 pages with a 4-KB page size, mapped onto a physical memory of 64 frames.
a. How many bits are required in the logical address?
b. How many bits are required in the physical address?

Answer:
a.
4KB = 4048B = 2^12
256 = 2^8
12 + 8 = 20 bits.

b.
4KB = 4048B = 2^12
64 = 2^6
12 + 6 = 18 bits.

Q3) 9.10
Consider a computer system with a 32-bit logical address and 4-KB page size. The system supports up to 512 MB of physical memory. How many entries are there in each of the following?
a. A conventional, single-level page table
b. An inverted page table

Answer:
a.
2^32 - 2^12 = 2^20 entries

b.
1 Megabytes (MB) = 1,048,576 Bytes
1 KB =1024 Bytes
512 MB = 512 * 1,048,576 Bytes
4 KB = 4 * 1024 Bytes

512MB/4KB =
512 * 1,048,576/ 4 * 1024 =
536870912/4096 =
131072 Bytes = 128KB entries

Q4) 8.23
Consider a system consisting of m resources of the same type being shared by n threads. A thread can request or release only one resource at a time. Show that the system is deadlock free if the following two conditions hold:
a. The maximum need of each thread is between one resource and m resources.
b. The sum of all maximum needs is less than m + n.

Answer:

a. $\sum_{i=1}^{n} Max_i < m + n$

b. $Max_i \geq 1$ for all i
Proof: $Need_i = Max_i - Allocation_i$
If there exists a deadlock state, then:

c. $\sum_{i=1}^{n} Allocation_i = m$

Use (a) to get: $\sum Need_i + \sum Allocation_i = \sum Max_i < m + n$

Use (c) to get: $\sum Need_i + m < m + n$

Rewrite to get: $\sum_{i=1}^{n} Need_i < n$

This implies that there exists a process P_i such that $Need_i = 0$. Since $Max_i \geq 1$, it follows that P_i has at least one resource that it can release. Hence, the system cannot be in a deadlock state.

Note: We use the terminology as presented in the section 8.7.2 “Several Instances of a Resource Type”