1)

a)

• Creates new child process by making exact copy of parent

process image

• Child process inherits resources of parent process and will

be executed concurrently with parent process

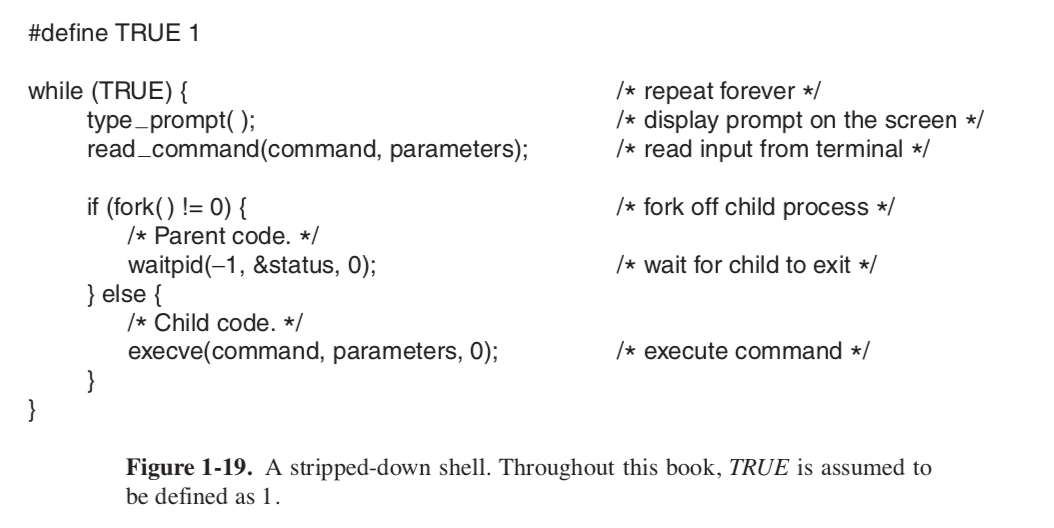
• fork() returns twice:

– In parent process: fork() returns process ID of child

– In child process: fork() returns 0

• On error, no child is created, and -1 is returned to parent

b)



c)

So that the child process knows which process to refer back to when the child process terminates (i.e. which process should I send the exit code to or the relevant data)

d)

It would be possible as long as the counter was incremented atomically or in some thread-safe way.

2)

a)

Each process has its own address space, global variables, open files, child processes and signals.

Each thread only has its own program counter, registers and stack.

2bi)

ii)

c)

di)

dii)

diii)

div)

4)

a)

lol

b)

The TLB caches recent virtual memory address translations. It needs to be flushed on a context switch.

c)

a(10 + 40) + (1-a)(10 + 40 + 40) = 60 => a = ¾

d)

* How recently the TLB was flushed.
* How large the TLB is.

e)

i)

At 64 bits with 4KB pages, you would have 2^52 pages. With 8B per entry (for example) you would need 30 million GB to store the page table.

ii)

Idea: store per frame instead of per page

* Hashed page table
* Inverted page table