Lista 2 Arquitetura Sistemas operacionais

**4**-- Dois motivos. Primeiro, não é desejável que o sistema operacional pare de funcionar. Segundo, se o kernel deixa a area stack data na área de memória de um programa em uso sobre o retorno de uma chamada de sistema, um programa maléfico pode ter acesso a dados de outros programas

**18**— Implementar threads pode fazer com que um programa seja executado no topo de qualquer S.O, mesmo se ele não suportar threads. A maior desvantagem seria permitir que threads fizessem system calls que normalmente causam bloqueios feitos pelo S.O que não parariam o processo inteiro, visto que isso vai contra o princípio dos threads.

**26** -- Não, o problema não irá ocorrer em round-robin pois processos só podem usar uma certa quantidade de tempo antes de ser forçado a esperar o próximo processo.

**27**-- Each thread calls procedures on its own, so it must have its own stack for the local variables, return addresses, and so on. This is equally true for user-level threads as for kernel-level threads.

39 – Dois processos filhos e dois pais

40—Se um processo ocorre mais de uma vez num round-robin, então ele terá dois turnos por passagem na lista. Uma razão para permitir isso seria a implementação de um sistema de prioridade primitiva, pois quanto mais um processo aparece na lista, maior a porcentagem de tempo que a CPU gastará nesse processo

42 – Num algoritmo round Robin cada processo é rotulado com um intervalo de tempo chamado time quantum, cada processo ganha acesso ao controle da CPU por este intervalo de tempo e executa suas tarefas

45 --