LICENCIATURA EM ENGENHARIA DE SISTEMAS E INFORMÁTICA
LICENCIATURA EM MATEMÁTICA E CIÊNCIAS DA COMPUTAÇÃO

Sistemas Operativos I

Segunda chamada[1]

30 de Janeiro de 2007 Duração: 2h30m

I

1. Diga o que entende por fragmentação interna e externa e mostre em que medida a fragmentação é prejudicial.

2. Explique de que forma a paginação permite garantir protecção num sistema de partilha de tempo com escalonamento RR. Se alterar o escalonamento para FIFO, ainda precisa de protecção? E de paginação? Justifique.

3. Como sabe *Thrashing* é uma situação a evitar em sistemas de memória virtual. Indique causas prováveis, sintomas (i.e. como a detectaria) e descreva eventuais soluções para este problema. Acha que a mudança de rejeição local para global pode aliviar o problema? Justifique.

II

Considere um programa `controlador` que executa concorrentemente um conjunto de programas especificados (sem argumentos) na sua linha de comando. Assim que o primeiro programa terminar com sucesso, o controlador deverá avisar os restantes programas em execução que serão forçados a terminar no espaço de tempo de 5 segundos. Para este efeito será enviado o sinal `SIGTERM`. Se algum dos processos sinalizar o `controlador` (enviando-lhe o sinal `SIGUSR1`) este prolongará (uma única vez) o período de espera por mais 10 segundos. No fim deste período, todos os processos em execução serão termiados com o sinal `SIGKILL`. Apresente o código-fonte do programa controlador recorrendo às primitivas de processos e sinais estudadas nas aulas.

III

Considere uma possível modelação de um processo de produção composto por N etapas sequenciais. Nessa modelação, cada etapa é representada por um programa `etapa_i` que recebe e envia uma linha de texto respectivamente de e para o seu standard input e standard output. A ocorrência de um problema numa dada etapa é assinalada com o envio de uma linha de texto vazia para o standard output devendo o respectivo programa ser então reiniciado. A matéria-prima inicial (processada na primeira etapa) encontra-se num ficheiro `materia.txt` e o produto final (resultado da última etapa) deverá ser armazenado em `produto.txt`. Apresente o código-fonte de um programa controlador que assegure o funcionamento deste processo de produção, recorrendo às primitivas de processos e ficheiros estudadas nas aulas.

Protótipos das chamadas ao sistema relevantes

Processos

- `pid_t fork(void);`
- `void exit(int status);`
- `int execvp( const char *file, char *const argv[]);`
- `pid_t wait(int *status);`
- `pid_t waitpid(pid_t pid, int *status, int flags);`
- `WEXITSTATUS(status);`
- `int execlp( const char *file, const char *arg, ...);`

Sistema de Ficheiros

- `int open(const char *pathname, int flags, mode_t mode);`
- `int creat(const char *pathname, mode_t mode);`
- `int close(int fd);`
- `int read(int fd, void *buf, size_t count);`
- `int write(int fd, const void *buf, size_t count);`
- `int pipe(int filedes[2]);`
- `int dup(int oldfd);`
- `int dup2(int oldfd, int newfd);`

Sinais

- `void (*signal(int signum, void (*handler)(int)))(int);`
- `int kill(pid_t pid, int signum);`
- `int alarm(int seconds);`
- `int pause(void);`

[1] Cotação — 8+6+6