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

Sistemas Operativos I

Primeira chamada[1]

11 de Janeiro de 2006 Duração: 2h30m

I

1. As *system calls* são funções que podem ser invocadas pelas aplicações para solicitarem serviços ao sistema operativo. De igual modo, as bibliotecas contêm conjuntos de funções. Qual a diferença? Por que razão não estão as bibliotecas "dentro" do sistema operativo ?

2. Suponha que, ao instalar um sistema operativo num computador portátil para uso escolar, lhe aparece uma "caixa" a pedir que escolha a estratégia de escalonamento de processos. Tem a hipótese de optar por RR (Round Robin) ou MLQ (MultiLevel Queues), e pode ajustar o campo da fatia de tempo (1 .. 999), que por omissão ficará a 1ms. Explique a sua escolha e justifique.

3. Suponha que é administrador/a de um sistema, e que começou a receber telefonemas de utilizadores a queixarem-se que "o sistema está lento". No entanto, verifica-se que a utilização do CPU não excede 20%. Que atitude(s) tomaria? Justifique.

II

Scripts são ficheiros de texto que representam comandos a ser interpretados por uma *shell*. Suponha que todos eles começam por especificar na primeira linha o interpretador a ser usado (exemplo: `#!/usr/bin/ruby`), ao qual deverão ser passadas as restantes linhas através do *standard input*.
Apresente o código C de uma função que executa um determinado *script* no interpretador respectivo. A função, para além de respeitar o protótipo abaixo, deverá retornar o código de saída do interpretador e guardar toda a informação enviada para o *standard error* num ficheiro de texto.

```
int executa_script(char *nome_ficheiro, char *ficheiro_erros);
```

III

Escreva um programa **testasite** que permita validar links web de acordo com as seguintes especificações:

- O programa deverá aceitar quatro argumentos: *i)* o nome de um ficheiro de texto com os endereços (links) a testar, *ii)* o nome do binário que permite testar a validade de um link, *iii)* o número máximo de instâncias do programa de validação de links que podem estar em execução simultaneamente e *iv)* o tempo máximo de duração global dos testes. Exemplo: `testasite links.txt testalink num_inst tempo_limite`
- O ficheiro de links contêm uma entrada por linha sendo o separador de entradas o caracter '\n'.
- O programa utiliza um **binário externo** (testalink). Esse binário permite testar a validade de um link recebido através do seu *standard input* e retorna o valor 0 (zero) em caso de este ser válido ou o valor 1 (um) no caso de este ser inválido.
- O programa principal deverá terminar a sua execução quando se esgotarem os links a testar ou uma das instâncias do programa de validação de links retornar o valor 1 (um).
- Caso os links não sejam todos testados dentro do intervalo de tempo especificado, o programa principal deverá terminar todos os testes e a sua execução.

---

[1] Cotação — 8+5+7

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(stat);`
- `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);`