

Sistemas Operativos

Recurso

19 de Junho de 2023

Duração: 2h

Por favor responda ao grupo I e a **cada** exercício do grupo II em folhas de teste **separadas**. Obrigado.

I

1 Para efetuar o escalonamento de todos os processos que está a gerir, o sistema operativo pode optar por utilizar apenas uma única política (p.ex. *Round-Robin*, *FCFS*), ou então, pode combinar e utilizar várias destas políticas para diferentes grupos de processos.

Indique qual a vantagem de suportar múltiplas políticas de escalonamento, para diferentes grupos de processos, e de que forma o sistema operativo tipicamente fornece esta possibilidade. Justifique a sua resposta.

2 Ao observar o comportamento do seu sistema operativo nota que, ao contrário do esperado, à medida que o mesmo executa mais programas intensivos em termos de CPU a utilização do processador decresce. Nota também que o número de *page-faults* e acessos a disco aumenta consideravelmente.

Com base nesta informação, que fenómeno pode estar a acontecer no seu sistema operativo? Ainda, na sua opinião, que acções deviam ser tomadas pelo sistema operativo perante esta situação? Justifique a sua resposta.

II

Considere uma empresa que utiliza um único ficheiro em formato **binário** para armazenar todos os registos individuais dos seus funcionários. Cada registo contém o nome, cargo e salário na empresa de um funcionário, de acordo com a seguinte estrutura:

```
struct RegistoF {  
    char nome[20];  
    char cargo[20];  
    int salário;  
};
```

1 Escreva a função `void aumentaSalarios(char* ficheiro, char* cargo, int valor, long N, int P)` que atualiza o ficheiro com N registos de forma aumentar em valor o salário dos funcionários com um dado cargo. **Valorização:** A função deve desencadear uma atualização concorrente com P processos.

2 Escreva a função `int validaSalarios(char* ficheiro, char* cargo)` que valida o salário de todos os funcionários com um dado cargo. A função deve retornar 1 caso algum funcionário receba menos do que o salário mínimo. Caso contrário, deve retornar 0. Para resolver este exercício, deve tirar proveito dos seguintes ficheiros executáveis:

- `./filtraCargo <ficheiro> <cargo>` - procura no ficheiro de registos os funcionários de um dado cargo e escreve para o `stdout` os respectivos registos.
- `./validaMin` - lê registos do `stdin`, no mesmo formato produzido pelo ficheiro executável `filtraCargo`, e termina com código de saída 1, caso algum funcionário receba menos que o salário mínimo, ou 0 caso contrário.

Algumas chamadas ao sistema relevantes

Processos

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

Sistema de Ficheiros

- `int open(const char *pathname, int flags, 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);`
- `long lseek(int fd, long offset, int whence);`
- `int access(const char *pathname, int amode);`
- `int pipe(int filedes[2]);`
- `int dup(int oldfd);`
- `int dup2(int oldfd, int newfd);`