Esta ficha foi elaborada para acompanhar o estudo da disciplina de Sistemas de Operação (a4s1) ou para complementar a preparação para os momentos de avaliação finais da mesma. Num segundo ficheiro poderás encontrar um conjunto de propostas de solução aos exercícios que estão nesta ficha. É conveniente relembrar que algum conteúdo destes documentos pode conter erros, aos quais se pede que sejam notificados pelas vias indicadas na página web, e que serão prontamente corrigidos, com indicações de novas versões.

1. Qual é o papel de um sistema de operação?

Um sistema de operação é tal que fornece um ambiente de execução de programas sobre hardware computacional de uma forma conveniente e eficiente. Para o fazer deverá alocar os devidos recursos à medida da necessidade, de acordo com detalhes de justiça e eficiência. Sendo este, por si, também um programa (embora de controlo), ele supervisiona a execução de programas dos utilizadores de forma a previnir erros e usos indevidos do computador e gere a operação de módulos de I/O.

2. Qual é o propósito das chamadas ao sistema (syscalls)?

As chamadas ao sistema permitem que processos de alto nível peçam serviços de baixo-nível, cujo risco de operação indevida poderá danificar o hardware computacional (daí a divisão, havendo componentes que asseguram a validade das operações que são feitas a baixo nível).

3. A escrita, por um utilizador, através de um teclado é feita através de uma leitura constante do módulo de I/O. Como é que tal se processa a partir do nível de utilizador, se o módulo de I/O é controlado pelo kernel?

De facto, quando premimos uma tecla de um teclado podemos estar em modo de utilizador, mas ao fazê-lo criamos uma interrupção, esta que, por polling, é atendida pelo modo de supervisão (kernel), o que vai ler o valor do módulo de I/O e tratá-lo da forma devida, dependendo do caso de aplicação.

4. Qual é a diferença entre as duas execuções abaixo e quais são os respetivos outputs no terminal?

```
# execução 1:
$ echo “Hello World!”

# execução 2:
$ echo “Hello World!” &
```

Na primeira execução tentamos correr o comando echo num terminal, onde esperamos que nele seja escrito "Hello World!". Esta execução será feita de forma síncrona com o terminal, isto é, o terminal só poderá prosseguir uma execução depois desta terminar. Na segunda execução temos uma assincronia devida ao ampersand '&' no fim do comando, o que fará com que o terminal se encontre preparado antes da execução do echo. Considerando que o nosso prompt é '$', então teríamos os seguintes outputs:

```
# output da execução 1:
Hello World!

# output da execução 2:
$Hello World!
```

5. Num sistema UNIX é permitido o redirecionamento da entrada e saída. Como é que é possível?

O redirecionamento da entrada e saída é possível nos sistemas operativos da família UNIX porque ele é constituído por um conjunto de ficheiros, sendo que tanto a entrada como a saída também são vistos como ficheiros.

6. O que é que significa o comando 'cat ficheiro.txt | grep "a"' e que técnica é que estamos a usar?

O comando 'cat ficheiro.txt | grep "a"' usa uma técnica designada de pipe (através do operador pipe '|') e serve para procurar instâncias da palavra "a" no ficheiro "ficheiro.txt" que será imprimido no terminal. O que acontece é que a impressão do ficheiro "ficheiro.txt" servirá de entrada para o comando "grep "a"".