1. **Explique o que entende por Processos em Sistemas Operativo.**

Em Sistemas Operativos, um processo é uma instância de um programa em execução.

Um processo inclui não só o código do programa, mas também o seu estado atual, como o contador de instruções, os registos do processador, as variáveis, a pilha de chamadas (stack), a área de dados e outros recursos necessários para a sua execução.

O sistema operativo é responsável pela gestão e controlo destes processos, garantindo que cada um recebe tempo de CPU, memória e acesso aos dispositivos de forma organizada e eficiente.

2. **Considere as duas implementações de mecanismos de comunicação entre processos: memória partilhada, objeto de comunicação do sistema.**
   a) **Compare-as tendo em conta a complexidade da programação da sincronização. Justifique sucintamente.**

Na memória partilhada, a sincronização é mais complexa, pois o programador tem de controlar o acesso concorrente (ex.: com semáforos).

Nos objetos de comunicação do sistema, a sincronização é mais simples, porque é gerida pelo sistema operativo.

   b) **Compare-as em termos de eficiência. Justifique sucintamente.**

A memória partilhada é mais eficiente, pois permite troca de dados direta entre processos, sem intervenção constante do sistema operativo.

Os objetos de comunicação do sistema são menos eficientes, devido à sobrecarga das chamadas ao sistema e cópia de dados.

3. **O que entende por Escalonamento?**

O escalonamento é uma funcionalidade do sistema operativo responsável por decidir a ordem de execução dos processos e quanto tempo de CPU cada processo irá receber.

Esta tarefa é gerida por um componente chamado ***scheduler***.

4. O que significa optimizar a utilização do processador e restantes componentes do sistema?

Significa maximizar o aproveitamento dos recursos do sistema, como o processador, memória e dispositivos de entrada/saída, reduzindo tempos de espera, evitando ociosidade e garantindo que os recursos são usados de forma eficiente e equilibrada.

O objetivo é melhorar o desempenho geral do sistema.

5. No Linux existem mecanismos de comunicação entre processos que são identificados pelos programadores por nomes/endereços pertencentes a diferentes espaços de nomes. Dê exemplos.

No Linux, os mecanismos de comunicação entre processos (IPC) podem usar nomes ou endereços em diferentes espaços de nomes. Exemplos:

- **Sockets Unix** – usam caminhos no sistema de ficheiros, como /tmp/socket_exemplo.
- **Pipes** nomeados (**FIFOs**) – identificados por um nome no sistema de ficheiros, como /tmp/minha_fifo.
- **Shared memory** (memória partilhada **POSIX**) – identificada por nomes como /shm_exemplo.

6. Considere o seguinte comando shell do Unix resultante do encadeamento de comandos elementares:

ls -l | sort > saída

a) Descreva o resultado do comando.

O comando lista os ficheiros do diretório atual em formato detalhado (**ls -l**), ordena essa listagem (**sort**) e redireciona o resultado ordenado para um ficheiro chamado saída. O conteúdo não aparece no ecrã.

b) Explique que mecanismos possibilitam este encadeamento de comandos.

O encadeamento é possível graças aos **pipes** e ao **redirecionamento** de ficheiros.

- O **pipe** (**|**) usa um mecanismo de comunicação entre processos que liga a saída de um comando à entrada de outro.
- O **redirecionamento** (**>**) envia a saída final para um ficheiro, usando o descritor de ficheiro associado à saída padrão (**stdout**).

c) Descreva sucintamente os principais passos do código que a shell executa para efetuar as operações deste comando encadeado.

I. A shell cria um pipe para ligar **ls -l** ao **sort**.
II. Cria um processo filho para executar **ls -l** com a saída ligada à escrita do **pipe**.
III. Cria outro processo filho para executar **sort** com a entrada ligada à leitura do **pipe**.
IV. A saída do **sort** é redirecionada para o ficheiro saída.
V. A shell espera que ambos os processos terminem.