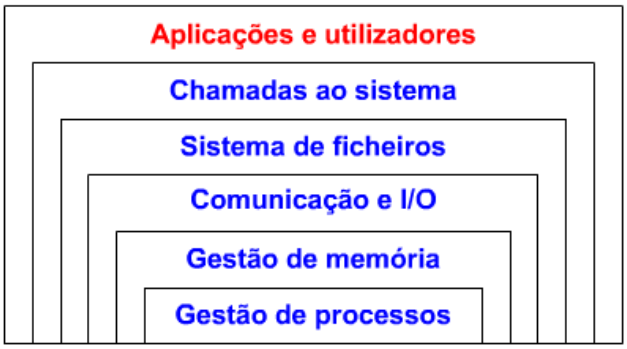
1. Explique quais devem ser as duas principais funções dos Sistemas Operativos e em que módulos funcionais se pode dividir o SO dos computadores.
   Uma possível resposta é o facto de o SO gerir recursos, isto é, ele é responsável por gerir a memória virtual, CPU, também processos (acho) e qualquer tipo de dispositivos. Outra das vertentes é estender a máquina, isto é, fazer com que se possa comunicar com a máquina com mais facilidade, isto em termos de programação, pois existem inúmeras coisas que teríamos que saber e controlar, por exemplo para simplesmente ligar o cabo da impressora ao PC.



2. Explique o que é entendido por Multiprogramming e por pseudo-paralelismo entre processos dum sistema operativo e qual o papel do escalonador.
   Num sistema de multiprogramação, o CPU também alterna os programas em execução. Na realidade, em qualquer instante de tempo o CPU apenas executa um único programa, no entanto, no decorrer de um segundo ele pode estar a trabalhar alternadamente com vários programas, dando ao utilizador a sensação que estão a ser executados vários processos em paralelo. Utiliza-se por vezes o termo pseudoparalelismo para designar esta troca rápida entre programas feita pelo CPU, como contraste face ao verdadeiro paralelismo que consiste em ter o CPU a executar enquanto um ou mais dispositivos de E/S estão também em execução.
   Escalonador decide que o processo actual já se encontra há demasiado tempo em execução e concede algum tempo de CPU a outro processo

3. Explique qual o propósito de guardar a informação sobre os vários ponteiros para os segmentos de dados, texto e stack na tabela de processos.

Guardar a informação é necessário pois, no caso do texto, se um texto está a ser escrito e perde a vez para um outro processo, é necessário guardar a informação de onde parou, para continuar a escrita no ponto correcto. Quando o processo é trocado do estado “Em execução” para o estado “Pronto a executar”, para que, quando retoma a execução, o processo possa continuar como se nada tivesse acontecido.

4. Explique o funcionamento interno do SO quando ocorrem a sequência dos seguintes eventos:

    a. É criado o processo PID#52;
    b. O processo PID#31 fica bloqueado à espera do Input;
    c. O processo PID#43 termina normalmente a sua execução;

    O processo 52 é criado e fica em execução em modo utilizador e passa para modo nuclear, de seguida fica pronto a executar e de seguida fica bloqueado.

    O processo 31 verifica se existe input, caso exista input passa de bloqueado para execução, caso não exista input continua bloqueado.

    O processo 43 passa de bloqueado para pronto a executar, de seguida para execução nuclear e passa para extinção (zombie), que é o estado intermédio entre a execução e a destruição do processo.

5. Explique qual o papel desempenhado pelos vários semáforos no código da fig.1.

```
typedef int semaphore;
semaphore mutex = 1;
semaphore db = 1;
int rc = 0;

void reader(void)
{
        while (TRUE) {
                down(&mutex);
                rc = rc + 1;
                if (rc == 1) down(&db);
                up(&mutex);
                read_data_base();
                down(&mutex);
                rc = rc – 1;
                if (rc == 0) up(&db);
                up(&mutex);
                use_data_read();
        }
}

void writer(void)
{
        while (TRUE) {
                think_up_data();
                down(&db);
                write_data_base();
                up(&db);
        }
}
```

Existem 2 semáforos, o db serve para controlar se existe leitores para não deixar o escritor escrever enquanto existem leitores. O mutex serve para controlar o acesso à região critica.
O leitor tenta aceder inicialmente à região critica, assim que entra incrementa o numero de leitores, assim que existir um leitor bloqueia o acesso à base de dados. Ele sai da região critica, lê os dados e volta a entrar na região critica. E decrementa o numero de leitores. Caso já não exista leitores dá acesso à base de dados e liberta o acesso à região critica.
No caso do escritor ele verifica se pode aceder a base de dados, caso possa escreve na base de dados e volta a pô-la disponível.