1.Descreva as duas perspectivas de definição de um sistema de operação. Mostre claramente em que circunstâncias cada uma delas é relevante.
As duas perspectivas de definição de um sistema de operação são o *Top-down e Botton-up*. **A Perspectiva 'top-down' (ou do programador):** O sistema operativo transmite ao programador uma abstracção do sistema computacional que o liberta do conhecimento preciso dos detalhes do 'hardware' subjacente; ou seja, o sistema operativo fornece um modelo funcional do sistema computacional, designado pelo nome de *máquina virtual*, que é mais simples de compreender e programar; o interface com o 'hardware', assim criado, origina um ambiente uniforme de programação, que é operado através das *chamadas ao sistema*, e possibilita por isso a portabilidade de aplicações entre sistemas computacionais estruturalmente distintos. ***Exemplos das funcionalidades criadas pelo sistema operativo:*** estabelecimento do ambiente base de interacção com o utilizador; disponibilização de facilidades para o desenvolvimento, teste e validação de programas; fornecimento de mecanismos para a execução controlada de programas, sua comunicação e sincronização mútuas; dissociação do espaço de endereçagem do programa das limitações impostas pelo tamanho da memória principal; organização da memória de massa em sistemas de ficheiros; definição de um modelo geral de acesso aos dispositivos de entrada/saída, independente das suas especificidades próprias; detecção de situações de erro e estabelecimento de uma resposta adequada.
Perspectiva 'bottom-up' (ou do construtor): Pode definir-se *sistema computacional* como sendo um sistema formado por um conjunto de recursos (processador (es), memória principal, memória de massa e diferentes tipos de controladores de dispositivos de entrada/saída) destinados ao processamento e armazenamento de informação. Neste sentido, o sistema operativo pode ser visto como: o programa que gere o sistema computacional, fazendo a atribuição controlada e ordeira dos seus diferentes recursos aos programas que por eles competem; o objectivo é, portanto, conseguir-se a rentabilização máxima do sistema computacional, garantindo-se uma utilização tão eficiente quanto possível dos recursos existentes.

7. Quais são as semelhanças e as diferenças principais entre um sistema de operação de rede e um sistema distribuído?
R: **Sistema operativo de rede:** *Objectivo:* tirar partido das facilidades actuais de interligação de sistemas computacionais (ao nível de hardware) para estabelecer um conjunto de serviços comuns a toda uma comunidade. Serviços: transferência de ficheiros entre sistemas computacionais (ftp); partilha de sistemas de ficheiros remotos, criando a ilusão de que são locais (NFS); partilha de dispositivos remotos (impressoras, etc.); acesso a sistemas computacionais remotos (telnet, remote login); correio electrónico (E-mail); acesso à Internet. **Sistema operativo distribuído:** • *Objectivo*: tirar partido das facilidades actuais de construção de sistemas computacionais com processadores múltiplos, ou de interligação de sistemas computacionais distintos, para estabelecer um ambiente integrado muito mais eficiente de interacção com o(s) utilizador(es). • *Metodologia*: garantir uma independência tão completa quanto possível do(s) processador(es), ou sistema(s) computaciona(l-is), onde os programas são executados tendo em vista; o balanceamento estático e/ou dinâmico de carga; o aumento da velocidade de processamento por incorporação de novos processadores ou sistemas computacionais; a paralelização de aplicações; a implementação de mecanismos de tolerância a falhas.
Características comuns aos sistemas de operação actuais: • apresentam um ambiente gráfico de interacção com o utilizador; • são sistemas interactivos multiutilizadores e multitarefa: – permitem a interacção em simultâneo com múltiplos utilizadores; – permitem que cada utilizador tenha em simultâneo múltiplos programas em execução; • implementam uma organização de memória virtual; • são sistemas de operação de rede: – permitem o acesso de um modo quase indistinto a sistemas de ficheiros e a dispositivos de entrada / saída locais e remotos; – incluem aplicações que permitem, entre outras coisas, o *login* em máquinas remotas, o correio electrónico e a navegação na Internet; • têm uma colecção muito grande de *device drivers* para potenciar a interligação de tipos muito variados de dispositivos de entrada / saída; • permitem em muitos casos a ligação dinâmica de dispositivos (*plug and play*).

1. A modelação do ambiente de multiprogramação através da activação e desactivação de um conjunto de processadores virtuais, cada um deles associado a um processo particular, supõe que dois factos essenciais relativos ao comportamento dos processos sejam garantidos. Quais são eles?
R: é preciso garantir que: a execução dos processos não é afectada pelo instante, ou local no código, onde ocorre a comutação, não são impostas quaisquer restrições relativamente aos tempos de execução, totais ou parciais, dos processos.

2.Qual é a importância da tabela de controlo de processos (*PCT*) na operacionalização de um ambiente de multiprogramação? Que tipos de campos devem existir em cada entrada da tabela?
R:A implementação de um ambiente multiprogramado implica a existência de uma quantidade variada de informação acerca de cada processo. Esta informação é mantida numa tabela, designada pelo nome de *Tabela de Controlo de Processos* (*PCT*), que é usada intensivamente pelo *scheduler* para fazer a gestão do processador e de outros recursos do sistema computacional.
Os tipos de campos devem existir em cada entrada da tabela *PCT* são: *identificadores* – do processo, do pai do processo e do utilizador a que o processo pertence; *caracterização do espaço de endereçagem* – sua localização (em memória principal ou na área de 'swapping'), de acordo com o tipo de organização de memória estabelecido; *contexto do processador* - valores de todos os registos internos do processador no momento em que se deu a comutação do processo; *contexto de I/0* - informação sobre os canais de comunicação e 'buffers' associados; *informação de estado e de 'scheduling'* - estado de execução do processo (de acordo com o diagrama de estados) e outra informação associada.

3. O que é o *scheduling* do processador? Que critérios devem ser satisfeitos pelos algoritmos que o põem em prática? Quais são os mais importantes num sistema multiutilizador de uso geral, num sistema de tipo *batch* e num sistema de tempo real?
R: O *scheduling* do processador é o sistema que permite a organização da comutação entre processos, isto é, permite calendarizar de forma eficaz e racional a atribuição dos recursos físicos aos diversos processos num ambiente de multiprogramação.
Desta forma a atribuição desses recursos deverá seguir determinados objectivos principais a serem satisfeitos numa política de 'scheduling' do processador que são os seguintes: Justiça – direito de cada processo obter uma parcela razoável de tempo de processador; Throughput – maximizar o número de processos terminados por unidade de tempo; Tempo de resposta - minimizar o tempo de resposta de utilizadores interactivos; Tempo de turn around – minimizar o tempo gasto pelo processador em tarefas relacionadas com a implementação da política escolhida; Eficiência – minimização do tempo associado com a selecção do próximo processo a que vai ser atribuído o processador e com a comutação propriamente dita.
O tempo de resposta é naturalmente o objectivo mais crítico a ter em conta em sistemas interactivos. Adicionalmente, são ainda de considerar o througput e a eficiência.
Em sistemas do tipo 'batch', deve-se particularmente ter em conta o tempo de turn-around, o 'throuput' e a eficiência

6. Indique quais são as funções principais desempenhadas pelo *kernel* de um sistema de operação. Neste sentido, explique porque é que a sua operação pode ser considerada como um *serviço de excepções*.
R: Uma das funções principais desempenhadas pelo *kernel* de um sistema de operação é o atendimento de excepções, ora podemos considerar que o kernel funciona como um ambiente de processamento uniforme em que as requisições externos por parte dele (excepções) ele responde com rotinas próprias que permitem o tratamento de erros e ocorrências. Desta forma toda a operação do kernel pode ser, visto como, atendimento a excepção desde uma linha de comando a uma comutação de processamento.
Assim, para que o sistema operativo, ao nível do ***kernel***, funcione no modo privilegiado, com total acesso a toda a funcionalidade do processador, as *chamadas ao sistema* associadas, quando não despoletadas pelo próprio 'hardware', são implementadas a partir de instruções ***trap***. Cria-se, portanto, um ambiente operacional uniforme, em que todo o processamento pode ser encarado como o *serviço de excepções*. Nesta perspectiva, a comutação de processos pode ser visualizada globalmente como uma vulgar rotina de serviço à excepção, apresentando, porém, uma característica peculiar que a distingue de todas as outras: *normalmente, a instrução que vai ser executada, após o serviço da excepção, é diferente daquela cujo endereço foi salvaguardado ao dar-se início ao processamento da excepção.*

7. O que é uma *comutação de contexto*? Descreva detalhadamente as operações mais importantes que são realizadas quando há uma comutação de contexto.
R: Uma *comutação de contexto*, consiste em alterar tudo o que tem a ver com o processo em causa (contexto do CPU e I/O) e alterar por outra previamente agendada. As operações mais importantes que são realizadas quando há uma comutação de contexto são as seguintes: - Salvaguarda da caracterização do espaço de endereçamento actual e dos contextos do processador e de I/O na entrada da PCT referente ao processo; - Actualização do estado do processo e de outra informação associada na entrada da PCT referente ao processo; - Selecção do próximo processo para execução da fila de espera READY-TO-RUN, usando um algoritmo específico de *scheduling*; - Colocação no estado RUN do processo agendado para execução e modificação de informação associada por actualização da entrada da PCT referente ao processo; - Restauro da caracterização do espaço de endereçamento actual e dos contextos do processador e de I/O por transferência da entrada da PCT referente ao processo.

9. Distinga disciplinas de prioridade estática das de prioridade dinâmica. Dê exemplos de cada uma delas.
R: As prioridades podem ser de dois tipos: • ***prioridades estáticas*** – quando o método de definição é determinístico independe da forma de como o sistema evolui ; • ***prioridades dinâmicas*** – quando o método de definição depende da história passada de execução do processo;
Do ponto de vista da, prioridade estática: Os processos são agrupados em classes de prioridade fixa, de acordo com a sua importância relativa. A comutação pode ocorrer sempre que seja necessário executar um processo de prioridade mais elevada. Trata-se da disciplina de atribuição mais *injusta*, existindo um risco claro de ocorrência de *adiamento indefinido* na calendarização para execução dos processos de prioridade mais baixa. Disciplina característica dos sistemas de tempo real. Aquando da sua criação, é atribuído a cada processo um dado nível de prioridade. Depois, à medida que o processo é calendarizado para execução, a sua prioridade é decrementada de uma unidade, sendo reposta no valor inicial quando atinge o valor mínimo. Deste modo, é colmatado o carácter *injusto* do método anterior. Disciplina característica de sistemas multi-utilizador. O Unix usa uma disciplina de 'scheduling' deste tipo. Enquanto que na *prioridade dinâmica:* Um método alternativo de privilegiar os processos interactivos consiste na definição de classes de prioridade de carácter funcional, entre as quais os processos transitam de acordo com a ocupação da última ou últimas janelas de execução