Simulado – Arquitetura e Organização de Computadores
Caio Domingues da Silva Santos - 201700550

1) a) Sistemas em lote (Batch) – No inicio dos sistemas de lote, os usuários preparavam Jobs, que eram os programas, dados e informações variadas que controlam a tarefa, e o entregavam ao operador do computador. Para acelerar o processo, os operadores reuniam lotes de Jobs similares e executavam como um grupo. Nesse ambiente, a CPU fica muitas vezes ociosa, já que a velocidade dos disp. Mecânicos de Entrada e Saída são mais lentas do que as do disp. eletrônicos.

b) Sistemas Multiprogramados – Em um SMP, o SO passa para outro Job e o executa. Quando o mesmo precisa esperar, a CPU vai para o próximo Job, mantendo essa ordem de execução, até que o primeiro Job na fila termine a sua espera e tem a CPU novamente. Desde que sempre haja um Job, a CPU nunca ficará ociosa.

2) • Processamento de dados – Execução de Operações Aritméticas ou Lógicas, sobre os dados.
• **Armazenamento de dados** – Armazenamento de Dados Temporários ou Fixos, para processamento ou controle.
• **Transferência de dados** - Transferência de dados da memória p/ o processador e dele para a memória. Assim como dados entre um processador e um periférico (Teclado, Mouse e etc) pela E/S.
• **Controle** – Determina as instruções que especifiquem a sequencia de execução de instruções para que ocorra a alteração.

3) CISC - Uma instrução complexa equivale a várias microinstruções presentes no microcódigo do processador. Quando o compilador encontra um comando complexo, ele converte esse comando em uma instrução complexa do processador. Essa instrução complexa na verdade é desmembrada internamente pelo decodificador de instruções em diversas microinstruções simples, o que é feito consultando o microcódigo.
[Retirado da Apostila de Arduino / Eletrônica Digital]

RISC – Como não existem instruções complexas, o compilador deve converter comandos complexos em diversas instruções simples que o mesmo resultado da operação. Assim, enquanto que CISC um comando de alto nível é convertido pelo compilador em poucas instruções, em um processador RISC esse comando é convertido em várias instruções
[Retirado da Apostila de Arduino / Eletrônica Digital]

4) Os sistemas com múltiplos processadores mais comuns utilizam SMP, no qual cada processador executa uma cópia idêntica do sistema operacional, e essas cópias comunicam-se conforme necessário, logo, todos os processadores são iguais, sem a relação master-slave. Alguns utilizam o multiprocessamento assimétrico, sendo assim, cada processador tem atribuído a si uma tabela única. Um processador master controla o sistema por inteiro (Há a possibilidade de ter mais de um master no mesmo sistema), e, os demais procuram o master para seguir instruções.

5) a) Permite que mais de uma instrução seja executada em paralela durante cada ciclo.

b) Incluem várias unidades de execução em um único chip. Eram usadas em processadores CISC para reduzir o tempo necessário para decodificar instruções complexas. Hoje, é encontrada na maioria dos processadores porque ela permite aumenta o desempenho, utilizando o paralelismo.

6) > Processador-memória: transferência de dados da memória para o processador e vice-versa.
> Processador-E/S: transferência de dados entre o processador e um disp. periférico por E/S.
>Processamento de dados: execução de operações aritméticas ou lógicas sobre os dados.
> Controle: "X" instruções podem especificar que a sequencia de execução de instruções seja alterada.
[Retirado da Apostila de Arduino / Eletrônica Digital]

7) Programas do Sistema – gerenciam a operação do próprio computador;
Programas de Aplicação – resolvem problemas para o usuário.

8) Opção (A)

9) A cada processo é atribuída a UID (User ID) de seu usuário. Quando um processo envia um sinal para outro processo, pode ser verificado se o transmissor e o receptor possuem a mesma UID. Da mesma forma, os usuários de determinado sistema podem ser divididos em grupos de usuários, com sua própria identificação de grupo GID (Group ID).
[Retirado da Apostila de Arduino / Eletrônica Digital: Relações entre Periféricos – Arduino; Arduino – Arduino; Arduino como Periférico]

10) a) **1. Memória Principal** – É a memória RAM - tipo de memória de escrita e leitura de acesso aleatório. (Random Acess Memory)
2. Memória Secundária – É a utilização de memórias não-voláteis como CD's e PenDrive's para armazenar informações.
[Retirado da Apostila de Arduino / Eletrônica Digital]

b) RAM (Random Access Memory) - Tipo de memória de leitura e escrita de acesso aleatório. Na memória o processador irá buscar programas e armazenar dados. A memória RAM é volátil. Cortando-se sua alimentação elétrica, apagamos os dados que estavam nela armazenados. ROM (Read Only Memory) - Uma memória apenas de leitura, na qual os dados não são apagados quando desligamos a alimentação. Esse tipo de memória como o nome sugere, só pode ser lido, portanto os dados são inalteráveis, no meio eletrônico, os dados são gravados em fábrica por meio da utilização de transistores ou diodos numa rede pré-formulada.
[Retirado da Apostila de Arduino / Eletrônica Digital]

c) Memória Estendida x Memória Expandida
A diferença entre elas, diz respeito ao método de acesso de memória acima de 1 MB que o processador utilizará e será diretamente baseada em seu modo de operação. Então, a memória expandida é uma técnica para acessar a memória estendida.
[Retirado da Apostila de Arduino / Eletrônica Digital]

11)

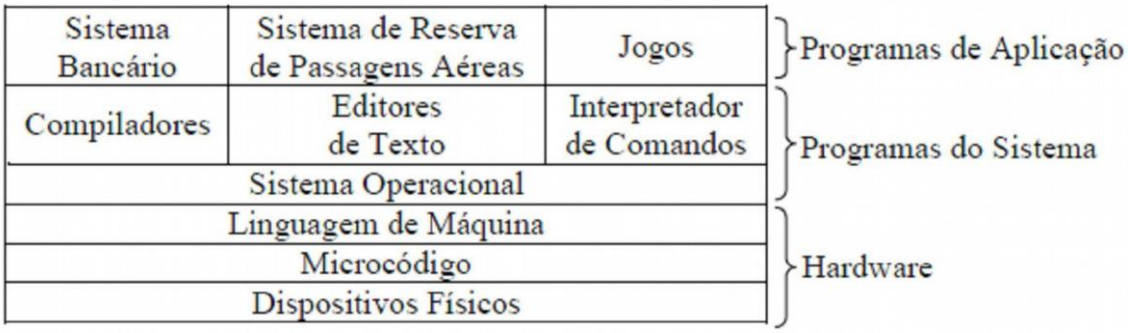

Imagem retirada da Apostila de Arquitetura e Organização de Computadores.