DIFERENÇA ENTRE MULTIPROGRAMAÇÃO, MULTITAREFA, MULTITHREADING E MULTIPROCESSAMENTO

1. **Multiprogramação -** Um computador executando mais de um programa ao mesmo tempo (como o Excel e o Firefox simultaneamente).
2. **Multiprocessamento -** Um computador que usa mais de uma CPU por vez.
3. **Multitarefa -** Tarefas que compartilham um recurso comum (como 1 CPU).
4. **Multithreading** é uma extensão da multitarefa.

**1. Multi programação -**

Em um sistema de computação moderno, geralmente há vários processos de aplicativos simultâneos que devem ser executados. Agora é responsabilidade do Sistema Operacional gerenciar todos os processos de forma eficaz e eficiente.  
Um dos aspectos mais importantes de um sistema operacional é a multiprogramação.  
Em um sistema de computador, existem vários processos aguardando para serem executados, ou seja, eles estão aguardando quando a CPU será alocada para eles e começarem sua execução. Esses processos também são conhecidos como jobs. Agora, a memória principal é muito pequena para acomodar todos esses processos ou trabalhos nela. Assim, esses processos são inicialmente mantidos em uma área chamada pool de jobs. Este pool de trabalhos consiste em todos os processos que aguardam a alocação da memória principal e da CPU.  
A CPU seleciona um trabalho de todos esses trabalhos em espera, o traz do pool de trabalhos para a memória principal e começa a executá-lo. O processador executa um trabalho até ser interrompido por algum fator externo ou ir para uma tarefa de E / S.

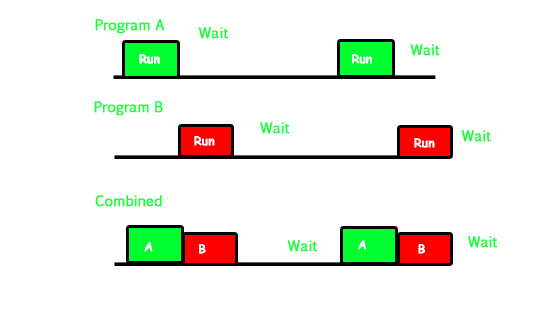
**Funcionamento do sistema não multiprogramado -**

* Em um sistema não multiprogramado, assim que um trabalho sai da CPU e vai para outra tarefa (digamos, E / S), a CPU fica ociosa. A CPU continua esperando e aguardando até que este trabalho (que estava em execução anteriormente) volte e reinicie sua execução com a CPU. Portanto, a CPU permanece livre por todo esse tempo.
* Agora, tem a desvantagem de que a CPU permanece ociosa por um longo período de tempo. Além disso, outras tarefas que estão aguardando para serem executadas podem não ter a chance de serem executadas porque a CPU ainda está alocada para a tarefa anterior.  
  Isso representa um problema muito sério que, embora outros trabalhos estejam prontos para serem executados, a CPU não está alocada para eles, pois a CPU está alocada para um trabalho que nem mesmo a está utilizando (pois está ocupada em tarefas de E / S).
* Não pode acontecer que um trabalho esteja usando a CPU por, digamos, 1 hora, enquanto os outros estão esperando na fila por 5 horas. Para evitar situações como essa e chegar a uma utilização eficiente da CPU, surgiu o conceito de multiprogramação.

A ideia principal da multiprogramação é maximizar o tempo da CPU.  
**Funcionando do sistema multi programado -**

* Em um sistema multiprogramado, assim que um trabalho vai para uma tarefa de E / S, o Sistema Operacional interrompe esse trabalho, escolhe outro trabalho do pool de trabalhos (fila de espera), dá CPU para este novo trabalho e inicia sua execução. O trabalho anterior continua realizando sua operação de E / S enquanto este novo trabalho realiza tarefas associadas à CPU. Agora, digamos que o segundo trabalho também vá para uma tarefa de E / S, a CPU escolhe um terceiro trabalho e começa a executá-lo. Assim que um trabalho conclui sua operação de E / S e volta para as tarefas da CPU, a CPU é alocada para ele.
* Dessa forma, o sistema não perde tempo de CPU aguardando a conclusão da tarefa de E / S.  
  Portanto, o objetivo final da multiprogramação é manter a CPU ocupada enquanto houver processos prontos para serem executados. Dessa forma, vários programas podem ser executados em um único processador, executando uma parte de um programa de uma vez, uma parte de outro programa depois disso, então uma parte de outro programa e assim por diante, executando, portanto, vários programas. Portanto, a CPU nunca fica ociosa.

Na imagem abaixo, o programa A é executado por algum tempo e depois entra no estado de espera. Nesse ínterim, o programa B começa sua execução. Portanto, a CPU não desperdiça seus recursos e dá ao programa B a oportunidade de ser executado.



**2. Multiprocessamento -**

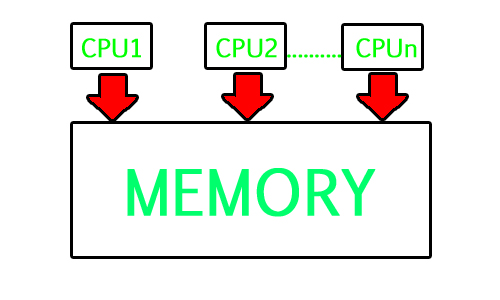
Em um sistema uni-processador, apenas um processo é executado por vez.  
Multiprocessamento é o uso de duas ou mais CPUs (processadores) em um único sistema de computador. O termo também se refere à capacidade de um sistema de suportar mais de um processador em um único sistema de computador. Agora, uma vez que existem vários processadores disponíveis, vários processos podem ser executados ao mesmo tempo. Esses multiprocessadores compartilham o barramento do computador, às vezes o relógio, a memória e os dispositivos periféricos também.

**Funcionando do sistema de multiprocessamento -**

* Com a ajuda do multiprocessamento, muitos processos podem ser executados simultaneamente. Digamos que os processos P1, P2, P3 e P4 estejam aguardando execução. Agora, em um sistema de processador único, primeiro um processo será executado, depois o outro, depois o outro e assim por diante.
* Mas com o multiprocessamento, cada processo pode ser atribuído a um processador diferente para sua execução. Se for um processador dual-core (2 processadores), dois processos podem ser executados simultaneamente e, portanto, serão duas vezes mais rápidos, da mesma forma um processador quad core será quatro vezes mais rápido que um único processador.

**Por que usar multiprocessamento -**

* A principal vantagem do sistema multiprocessador é fazer mais trabalho em um período de tempo mais curto. Esses tipos de sistemas são usados ​​quando uma velocidade muito alta é necessária para processar um grande volume de dados. Os sistemas de multiprocessamento podem economizar dinheiro em comparação aos sistemas de processador único porque os processadores podem compartilhar periféricos e fontes de alimentação.
* Ele também fornece maior confiabilidade no sentido de que, se um processador falhar, o trabalho não para, apenas fica lento. por exemplo, se tivermos 10 processadores e 1 falhar, o trabalho não será interrompido; em vez disso, os 9 processadores restantes podem compartilhar o trabalho do 10º processador. Portanto, todo o sistema funciona apenas 10% mais devagar, em vez de falhar completamente.



Multiprocessamento se refere ao hardware (ou seja, as unidades de CPU) em vez do software (ou seja, processos em execução). Se o hardware subjacente fornece mais de um processador, isso é multiprocessamento. É a capacidade do sistema de aproveitar o poder de computação de vários processadores.

**Diferença entre multiprogramação e multiprocessamento -**

* Um sistema pode ser multiprogramado por ter vários programas em execução ao mesmo tempo e multiprocessado por ter mais de um processador físico. A diferença entre multiprocessamento e multiprogramação é que o multiprocessamento basicamente executa vários processos ao mesmo tempo em vários processadores, enquanto que a multiprogramação mantém vários programas na memória principal e os executa simultaneamente usando uma única CPU.
* O multiprocessamento ocorre por meio de processamento paralelo, enquanto a multiprogramação ocorre pela troca de um processo para outro (fenômeno denominado como troca de contexto).

**3. Multitarefa -**

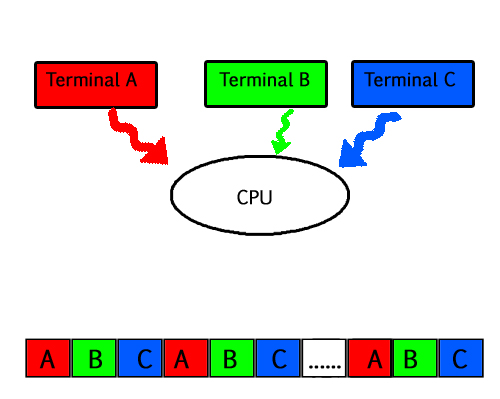
Como o próprio nome sugere, multitarefa se refere à execução de várias tarefas (digamos, processos, programas, threads, etc.) ao mesmo tempo. Nos sistemas operacionais modernos, podemos tocar música MP3, editar documentos no Microsoft Word, navegar no Google Chrome tudo simultaneamente, isso é feito por meio de multitarefa.

Multitarefa é uma extensão lógica da multiprogramação. A principal maneira pela qual a multitarefa difere da multiprogramação é que a multiprogramação funciona unicamente no conceito de troca de contexto, enquanto a multitarefa se baseia no compartilhamento de tempo juntamente com o conceito de troca de contexto.

**Funcionando do sistema multitarefa -**

* Em um sistema de compartilhamento de tempo, cada processo recebe algum quantum específico de tempo para o qual um processo deve ser executado. Digamos que haja 4 processos P1, P2, P3, P4 prontos para execução. Assim, a cada um deles é atribuído algum quantum de tempo para o qual serão executados, por exemplo, quantum de tempo de 5 nanossegundos (5 ns). Quando um processo começa a ser executado (digamos P2), ele é executado por aquele período de tempo (5 ns). Após 5 ns, a CPU inicia a execução do outro processo (digamos P3) pelo período de tempo especificado.
* Assim, a CPU faz com que os processos compartilhem fatias de tempo entre eles e os executem de acordo. Assim que o quantum de tempo de um processo expira, outro processo começa sua execução.
* Aqui também está ocorrendo basicamente uma troca de contexto, mas tão rápido que o usuário é capaz de interagir com cada programa separadamente enquanto ele está sendo executado. Dessa forma, o usuário tem a ilusão de que vários processos / tarefas estão sendo executados simultaneamente. Mas, na verdade, apenas um processo / tarefa está sendo executado em um determinado instante de tempo. Em multitarefa, o compartilhamento de tempo se manifesta melhor porque cada processo em execução consome apenas uma boa parte do tempo da CPU.

Em um sentido mais geral, multitarefa se refere a ter vários programas, processos, tarefas e threads em execução ao mesmo tempo. Este termo é usado em sistemas operacionais modernos quando várias tarefas compartilham um recurso de processamento comum (por exemplo, CPU e memória).



* Conforme ilustrado na imagem acima, A qualquer momento a CPU está executando apenas uma tarefa enquanto outras tarefas aguardam sua vez. A ilusão de paralelismo é alcançada quando a CPU é reatribuída a outra tarefa. ou seja, todas as três tarefas A, B e C parecem ocorrer simultaneamente devido ao compartilhamento de tempo.
* Portanto, para que a multitarefa ocorra, em primeiro lugar deve haver multiprogramação, ou seja, a presença de vários programas prontos para execução. E em segundo lugar, o conceito de compartilhamento de tempo.

**4. Multi threading -**

Um thread é uma unidade básica de utilização da CPU. Multi threading é um modelo de execução que permite que um único processo tenha vários segmentos de código (ou seja, threads) em execução simultaneamente dentro do “contexto” desse processo.  
por exemplo, VLC media player, onde um thread é usado para abrir o VLC media player, um thread para reproduzir uma música específica e outro thread para adicionar novas músicas à lista de reprodução.

Multi threading é a capacidade de um processo de gerenciar seu uso por mais de um usuário ao mesmo tempo e de gerenciar várias requests do mesmo usuário sem ter que ter várias cópias do programa.

**Funcionamento do sistema multi threading -**

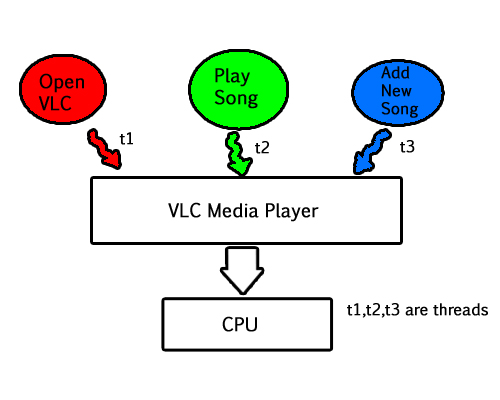
**Exemplo 1 -**

* Digamos que haja um servidor web que processa requests de clientes. Agora, se ele for executado como um único processo encadeado, não será capaz de processar várias requests ao mesmo tempo. Primeiramente um cliente fará sua solicitação e finalizará sua execução e só então o servidor poderá processar outra solicitação do cliente. Esta é uma tarefa muito cara, demorada e cansativa. Para evitar isso, o multi threading pode ser usado.
* Agora, sempre que uma nova solicitação do cliente chega, o servidor da web simplesmente cria um novo encadeamento para processar essa solicitação e retoma sua execução para ouvir mais requests do cliente. Portanto, o servidor web tem a tarefa de ouvir as novas requests do cliente e criar threads para cada solicitação individual. Cada thread recém-criado processa uma solicitação do cliente, reduzindo assim a carga no servidor da web.

**Exemplo 2 -**

* Podemos pensar em threads como processos filhos que compartilham os recursos do processo pai, mas são executados de forma independente. Agora veja o caso de uma GUI. Digamos que estejamos realizando um cálculo na GUI (que está demorando muito para ser concluído). Agora não podemos interagir com o resto da GUI até que este comando termine sua execução. Para poder interagir com o resto da GUI, este comando de cálculo deve ser atribuído a um thread separado. Portanto, neste momento, 2 threads estarão em execução, ou seja, um para cálculo e um para o resto da GUI. Portanto, aqui em um único processo, usamos vários threads para várias funcionalidades.

A imagem abaixo descreve completamente o exemplo do player VLC:



**Vantagens do Multi threading -**

* Os benefícios do Multi threading incluem maior capacidade de resposta. Como há vários threads em um programa, se um thread estiver demorando muito para ser executado ou se for bloqueado, o restante dos threads continuará executando sem nenhum problema. Assim, todo o programa permanece responsivo ao usuário por meio de threads restantes.
* Outra vantagem do multithreading é que ele é mais barato. Criar novos processos e alocar recursos é uma tarefa demorada, mas como os threads compartilham recursos do processo pai, criar threads e alternar entre eles é comparativamente fácil. Conseqüentemente, o multithreading é a necessidade de sistemas operacionais modernos.