Respostas SO

1. A diferença é que é o programa mais importante é a parte que fica entre o hardware e os programe aplicativos, então o sistema operacional que carrega os programas e aplicativos.
2. Para permitir o compartilhamento dos recursos de hardware por diversas aplicações além de gerenciar os recursos do sistema.
3. 0
4. Processo é um trabalho em execução em um sistema computacional, que aloca recursos do sistema para a execução
5. Não, o processador executa partes dos programas onde o nome disso é chaveamento cujo o processador executa por determinado tempo um programa e chaveia e com isso muda para outro programa, o tempo desse chaveamento é muito rápido e da a impressão que o usuário tem é que está sendo executando sendo que na verdade não está.
6. O sistema operacional salva todo o estado do processo que esta em execução e carrega o estado do processo é salvo em uma estrutura chamada PCB ( Process Control Block).
7. Threads são fluxos de execução que rodam dentro de um processo (aplicação). Normalmente os threads compartilham regiões de memória, mas não necessariamente. Lembre-se de encapsulamento. Processos, os avós dos threads permitem que o seu sistema operacional execute mais de uma aplicação ao mesmo tempo enquanto que threads permitem que sua aplicação execute mais de um método ao mesmo tempo.

Todos os programas que fizemos até agora só tinham um único caminho, fio, fluxo, de execução. Nenhum deles executava duas coisas (dois pedaços de código) simultaneamente. Grande parte do software de qualidade escrito hoje faz uso de mais de uma linha de execução, mais de um thread. São os chamados programas multithreaded.

1. Preemptivo:

* não depende somente do processo;
* S.O. pode tomar o processador mesmo que ainda tenha processos a executar;

Não-Preemptivo:

* depende somente do processo;
* processador não pode ser tomado;

9- First-Come,First-Served (FCFS) : os processos são executados na ordem em que chegam ao escalonador. Pode apresentar diferentes tempos médios de espera de acordo com a ordem de chegada dos processos.

-Shortest JobFirst (SJF):

* Versão não-preemptiva: uma vez que o processo recebe tempo de CPU, não pode ser interrompido até que utilize o tempo de CPU alocado.
* Versão preemptiva: se um novo processo chega e possui um tempo de CPU menor que o tempo restante do processo em execucão, este será interrompido. Esta forma é conhecida como Shortest-Remaining-TimeFirst (SRTF).

10-   Cooperação: threads trabalham em cooperação para atingir um objetivo.

Ex:

Semáforo Mecanismos que garantem que somente N processos podem acessar um   certo recurso em um dado momento.

       Competição: threads concorrentes disputam pelo uso exclusivo de recursos.

Ex:

Combinação de transferência de dados com exclusão mútua e sincronização. Um processo gera as mensagens (produtor), colocando-as em uma fila de mensagens. O outro processo (consumidor) retira as mensagens da fila. Caso a fila fique totalmente cheia, o processo produtor espera até que seja liberado espaço na fila. Caso a fila fique vazia, o processo consumidor espera até que um novo elemento seja inserido na fila.

Resposta GCM

1. o gerenciamento de configuração é o processo para estabelecer e manter a consistência dos atributos físicos, funcionais e de performance de um produto durante sua vida, de acordo com seus requisitos, desenho e informações operacionais.
2. Manual do usuário, documentação de manutenção, descrição do banco de dados, programa executável, requisitos e especificação de projetos.
3. Modelagem de negócios, requisitos e implantação.
4. **Comerciais**:

Microsoft Source Control

Microsoft Visual SourceSafe

IBM Rational Clear Case

Borland StarTeam

**Gratuitas**:

CVS (Concurrent Version System)

Subversion

Git

1. Existem dois tipos de arquitetura uma centralizada e um distribuída (descentralizada)

**Centralizada:**

Os arquivos ficam em um servidor e as modificações são feita upando elas.

**Descentralizada:**

Existe um servidor mas todos os computadores tem uma versão do máster local, onde são feitas as alterações e depois são upadas no servidor máster. O Git por exemplo é uma ferramenta de controle de versão que utiliza uma arquitetura distribuída.

1. Uma vantagem é o desenvolvimento em paralelo que todos podem produzir ao mesmo tempo com uma versão do máster que deixa o servidor livre para outras funções, demais operações feitas são feitas em uma copia então não irar afetar o principal tendo também uma vantagem de trabalhar estando off-line.
2. Não o repositório está criado no git entretanto pode ser criado um repositório em seu desktop.
3. Criar um branch, criar o repositório local, realizar os commits e pushes no branch criado, criar um pull request e realizar o merge no branch master do repositório local.
4. Ter que ser feito uma alteração no arquivo manual mente o conflito vai dar # file quadrado.py

&lt;&lt;&lt;&lt;&lt;&lt;&lt; branch2

n1= int(input(&#39;n1:&#39;))

quadrado = n1\*n1

print(quadrado)

print(&#39;fim&#39;)

=======

print(&#39;início&#39;)

n1= int(input(&#39;n1:&#39;))

# cálculo do quadrado

print(&#39;FIM&#39;)

&gt;&gt;&gt;&gt;&gt;&gt;&gt; máster

O resultado será:

# file quadrado.py

print(&#39;início&#39;)

n1= int(input(&#39;n1:&#39;))

# cálculo do quadrado

quadrado = n1\*n1

print(quadrado)

print(&#39;FIM&#39;)

1. nao vai dar nenhum problema e o ficar

# file dobro.py

n = int(input(&#39;n:&#39;))

dobro = 2\*n

print(dobro)

print(&#39;fim&#39;)