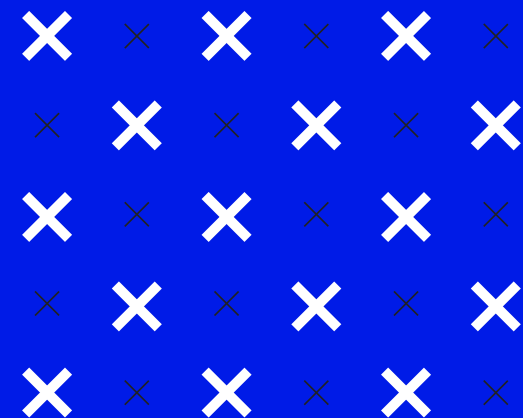


THREADS



mentorama.
@prof.felipeassuncao

Introdução a threads

- Conceitos iniciais sobre concorrência, paralelismo, multiprocessing, etc.
- Conceito de threads
- Como criar threads e trabalhar com eles
- Como usar as ferramentas comuns que o modulo threading em Python fornece

Recursos e ferramentas

- Jupyter Notebook
- Editor de código de sua preferência

Neste módulo

Aula 1 - Threads

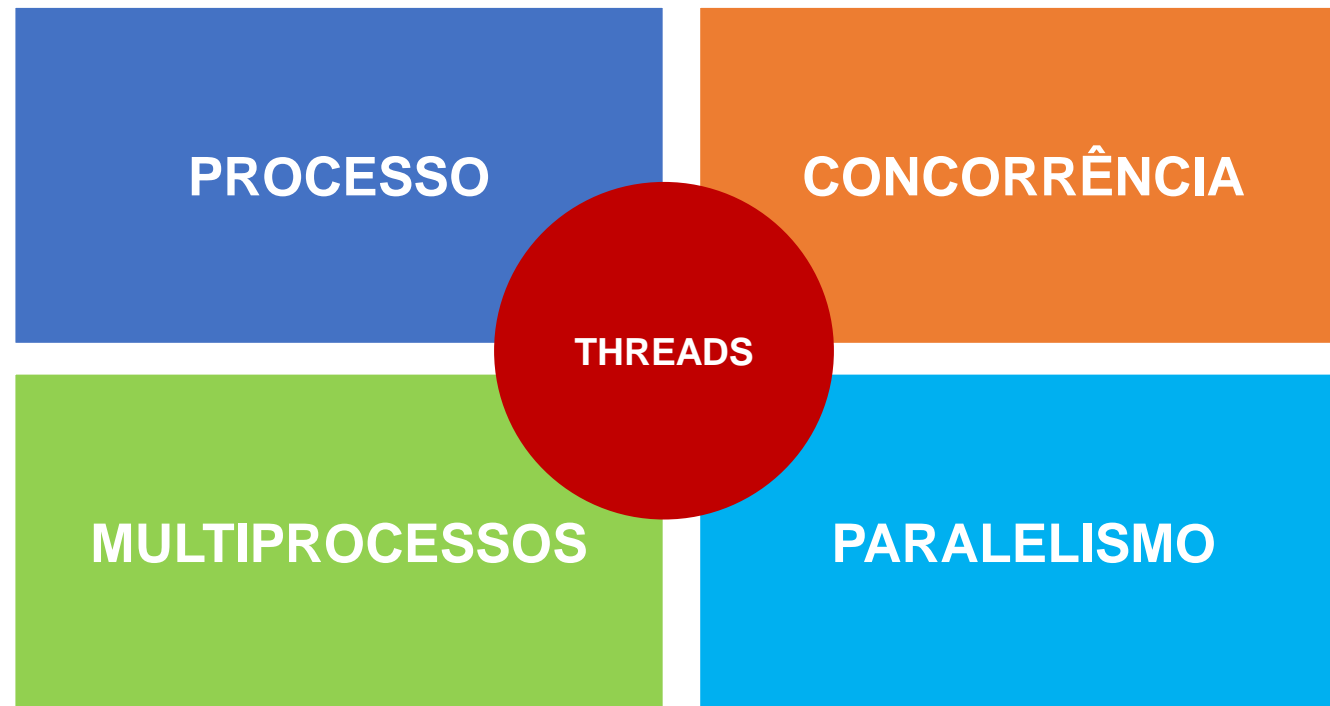
Aula 2 - Prática

Aula 3 - Exercícios

1. THREADS

mentorama.

Conceitos iniciais



Conceitos iniciais



mentorama.

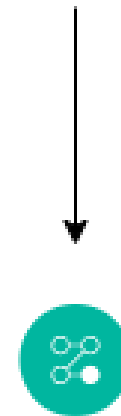
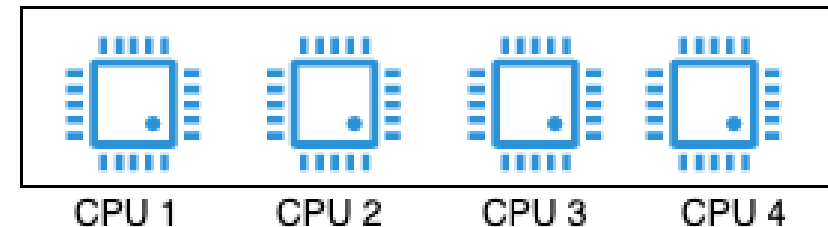
Processo

“É uma instancia de um programa sendo executado em um computador”

- Cada processo tem sua área de memória própria.

Processo Serial

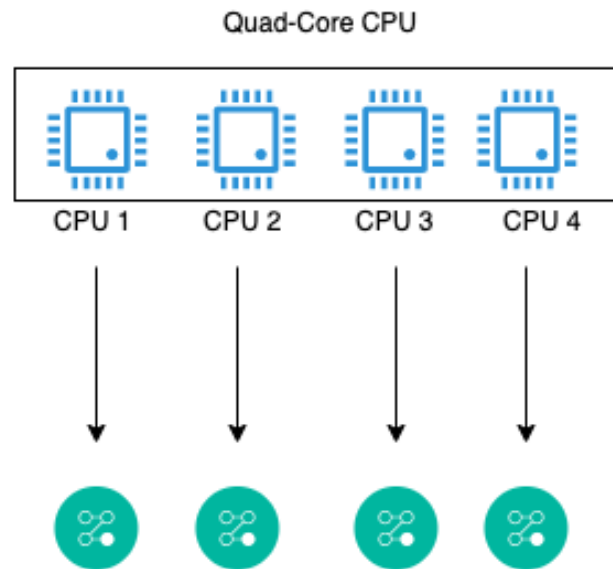
Quad-Core CPU



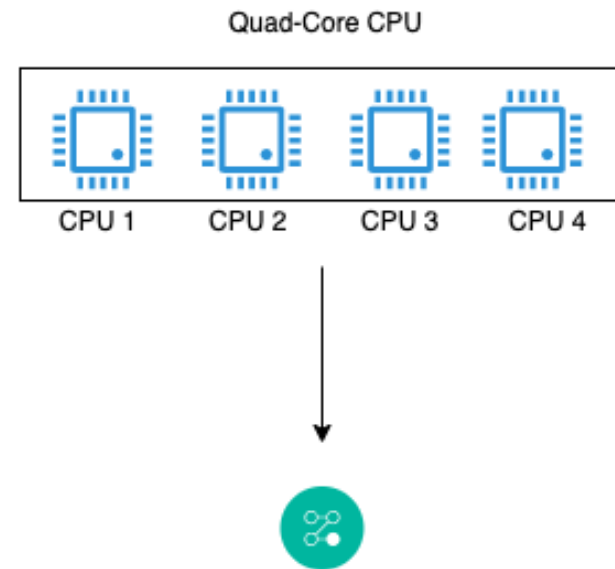
Paralelismo

"Paralelismo é sobre a execução paralela de tarefas, ou seja, mais de uma por vez de forma simultânea"

Processo Paralelo



Processo Serial



Concorrência

"É a capacidade de se executar duas ou mais tarefas em um mesmo período de tempo"

Paralelismo vs. concorrência

*“Concorrência é sobre **lidar** com várias coisas ao mesmo tempo e paralelismo é sobre **fazer** várias coisas ao mesmo tempo.”*

Threads

- Representa uma atividade que será executada em um fluxo separado
- Para iniciar uma thread chamamos a função 'start()'

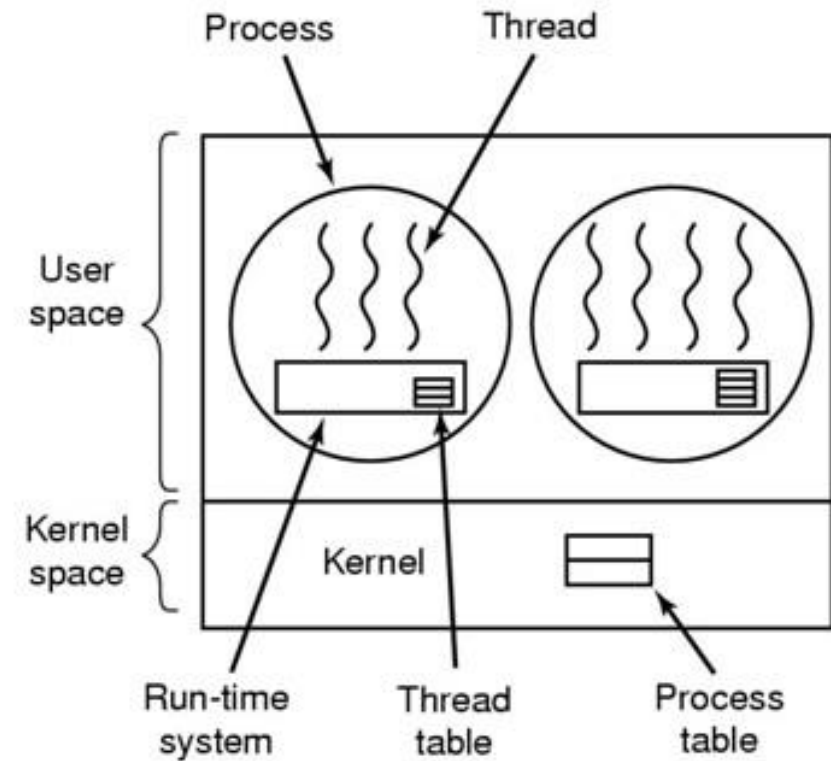
```
>>> import threading
```

```
>>> t1 = Threads(target=obj.Function_Name).start()
```

- Podemos finalizar um thread quando o processo python é finalizado, ou não

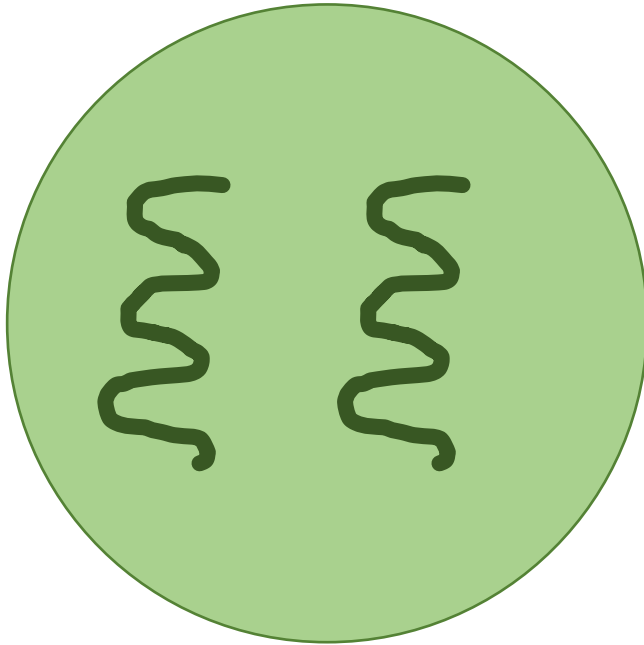
Threads vs. processo

- Um thread é uma linha de execução que está dentro de um processo
- Cada um desses threads é um pacote independente de execução
- Um processo pode conter mais de um thread

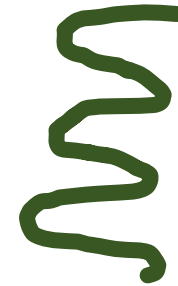


Threads vs. processo

PROCESSO



THREADS

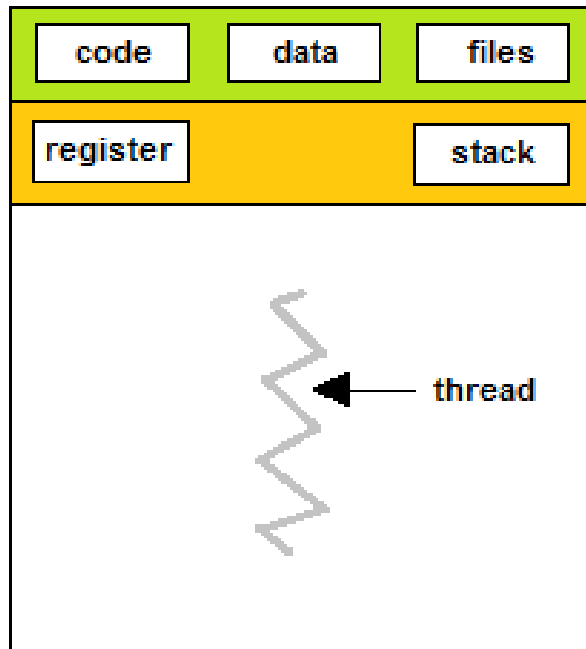


Multithreads

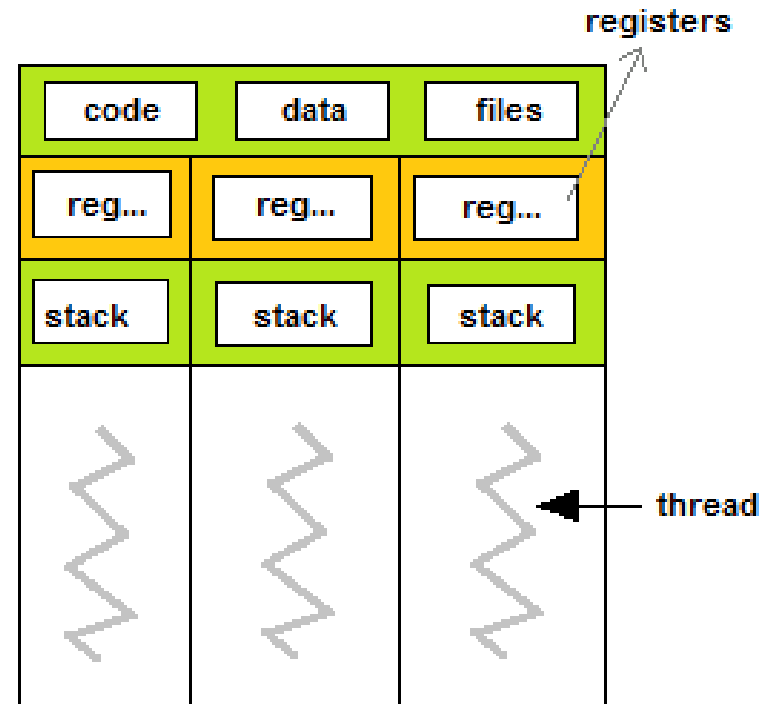
- Multithreading é uma técnica em que vários threads são gerados por um processo para fazer diferentes tarefas, quase ao mesmo tempo, apenas uma após a outra.
- Isso dá a ilusão de que os encadeamentos estão sendo executados em paralelo, mas na verdade são executados de maneira simultânea.
- Em Python, o Global Interpreter Lock (GIL) impede que os threads sejam executados simultaneamente.

mentorama.

Monothread vs. multithread



single-threaded process



multithreaded process

Multiprocessamento

- Multiprocessamento é uma técnica em que o paralelismo em sua forma mais verdadeira é alcançado.
- Vários processos são executados em vários núcleos de CPU, que não compartilham os recursos entre eles.
- Cada processo pode ter muitos threads em execução em seu próprio espaço de memória.
- Ele roda em Unix e Windows.


```
>>> import multiprocessing
```

mentorama.

Tipos de threads

- Modo usuário
- Modo de núcleo
- Modo híbrido

Threads no modo usuário

- Implementado totalmente no espaço do usuário **ENDEREÇAMENTO NO ESPAÇO DO USUÁRIO**
- Por meio de uma biblioteca (criação, exclusão, execução etc)  **NÃO NECESSARIAMENTE GERENCIAMENTO**
- Criação e escalonamento são realizados sem o conhecimento do kernel
- Para o kernel é como se rodasse um programa monothread, gerenciadas como processos no kernel

Vantagens de uso das threads

- Criar uma thread é mais rápida que criar um novo processo
- Tende a aumentar o desempenho com a exploração do paralelismo real, com melhor capacidade de resposta
- Melhor organização do programa
- Obrigação de “programar bem”

Desvantagens de uso das threads

- O processo de depuração é mais difícil
- A exclusão das variáveis em um tarefa que depende da outra, pode resultar em estados não definidos, prejudicando o uso posterior desta variável pela próxima tarefa
- Pode levar maior tempo de execução quando demandam uso intensivo da CPU

Comandos comuns para threads

threading.Thread

COMANDOS	DESCRIÇÃO
start()	Inicia as ações da thread
run()	Ações de thread a serem executadas
join()	Bloqueia até a thread encerrar
is_alive()	Retorna se a thread está viva
daemon	Indicador se a thread é do tipo daemon

Objetos em thread

OBJETO	DESCRIÇÃO
Thread	Objeto que representa um único thread de execução.
Lock	Objeto de bloqueio primitivo.
RLock	O objeto de bloqueio RLock ou Reentrante fornece capacidade para um único thread (re) adquirir um bloqueio já mantido (bloqueio recursivo).
Condition	Objeto de variável de condição faz com que um thread espere até que certa "condição" seja satisfeita por outro thread (como mudança de estado ou algum valor de dados)
Event	É uma versão mais geral das variáveis de condição, em que várias threads podem ser feitas para aguardar a ocorrência de algum evento e todas as threads em espera só serão ativadas quando o evento acontecer.
Semaphore	Fornece um "contador" de recursos finitos compartilhados entre blocos de threads quando nenhum está disponível.
BoundedSemaphore	Semelhante a um semáforo, mas garante que nunca exceda seu valor inicial.
Timer	Semelhante ao Thread, exceto que ele espera por um período de tempo especificado antes de ser executado.
Barrier	Cria uma "barreira" na qual um número especificado de threads deve chegar antes que todos tenham permissão para continuar.

Resumo

- Processos
- Paralelismo
- Threads
- Multithreads
- Multiprocessos
- Objetos em threads



2. PRÁTICA

mentorama.



Vamos praticar?

- Nesta prática iremos explorar a utilização de Threads



Resumo

- Modulo threading e time
- Threads com funções
- Threads com classes



EXERCICIOS

mentorama.

