

Otimização de memória

O Python utiliza um *buffer*, uma região temporária na memória, como área de transferência para os dados que estão sendo lidos ou escritos em um arquivo.

Quando lemos um arquivo, por exemplo, o Python pega os dados do arquivo, passa para o *buffer*, e, então, o conteúdo do arquivo pode ser utilizado pela aplicação.

Quando iteramos sobre um arquivo, o Python vai colocando os dados no *buffer* à medida que o *loop* é executado, sendo uma boa forma de lidar com a memória consumida pelo sistema.

Por que é interessante não carregar todo o conteúdo do arquivo na memória de uma só vez?

Selecione uma alternativa

- A** Na verdade, o ideal é carregar todo o arquivo na memória. O Python vai ler todo o arquivo e colocá-lo em memória quando o método `open` for invocado.
- B** Quando não carregamos todo o arquivo na memória, o sistema fica mais lento pois é necessário uma ação do usuário para que a leitura, ou a escrita, seja realizada.
- C** Caso o arquivo seja muito grande, pode ocorrer de não existir espaço suficiente na memória para alocar todos os dados, ocasionando um erro de memória.