PROJETO CLASSIFICATÓRIO

Processo seletivo – Rocky TI

Autor: Guilherme Milani de Oliveira

Problema: Um arquivo contendo um banco de dados NoSQL foi corrompido, a tarefa do algoritmo desenvolvido é restaurar os dados perdidos, reparando o nome, preço e quantidade dos produtos. Também foi realizada uma verificação da funcionalidade do algoritmo com uma listagem ordenada e contagem de algumas variáveis.

Escolha da linguagem: Foi utilizada a linguagem Python para a realização do projeto devido a experiência e familiaridade com a língua. Além de, ao ler a proposta do projeto, já ter uma breve noção de como construir o código em Python.

Funcionalidade: A seguir uma breve explicação sobre as funções elaboradas no projeto.

- json2py: Recebe o caminho do arquivo JSON que será aberto e o converte em uma lista de dicionários no Python.

- py2json: Recebe os dados a serem transformados em JSON e realiza a conversão com uma indentação similar ao “broken-database.json”.

- fix_name: Recebe a variável com os dados do “broken-database” e, a cada objeto da lista, verifica cada carácter no campo “name”, para identificar onde houveram carácteres indevidos e substituí-los pelo seu respectivo valor correto.

- fix_price: Recebe os dados e a cada dicionário na lista verifica o tipo do valor associado a key “price” e converte para float se necessário.

- fix_quantity: Recebe os dados e a cada dicionário na lista verifica se há a key “quantity”, se não houver, inicializa o mesmo, porém com o valor 0.

- change_char_at: Função utilizada internamente para substituir um carácter específico numa string passada como argumento por meio de uma manipulação simples utilizando lista e o método join.

- sort: Ordena o banco de dados primeiro por categoria em ordem alfabética e por id em ordem crescente, ambos utilizando a função sort com uma função lambda representando a key que deverá ser utilizada para ordenar em cada situação. Após isso imprime os dados ordenados.

- sum_by_category: Retorna um dicionário com a soma acumulada dos produtos em estoque para cada categoria, criando uma key para cada categoria diferente que identificar ao iterar sobre o banco de dados, somando tudo em seguida.

Tratamento de exceções: Realizamos um tratamento caso o arquivo inserido como argumento não seja encontrado. Mas ainda existem exceções que podem ser tratadas, como , por exemplo, caso o banco de dados inserido não corresponda aos moldes esperados, o que geraria diversos problemas diferentes.

Observações: O código foi montado como uma pequena API, em que um usuário qualquer poderia utilizá-la para tratar um banco de dados de sua escolha sem saber detalhes sobre a implementação, porém, um trabalho que vai além do escopo desse projeto seria necessário para realizar essa tarefa plenamente, e é importante salientar essa informação. Também é válido destacar que uma GUI seria bem simples de implementar, mas, como não foi algo requisitado, a melhor escolha foi deixar a utilização do código mais simples.
Para testar o projeto, além de ter o Python 3 devidamente instalado e configurado em sua máquina, é necessário apenas utilizar o comando ‘python3 resolucao.py’ e ter o arquivo “broken-database.json” na mesma pasta de “resolucao.py”.