Projeto Classificatório - ROCKY Full Digital Performance

Processo seletivo – TI - Candidato: Diego da Silva Lourenço

1. Motivo da escolha da linguagem

A linguagem Python foi escolhida por eu já ter feito um curso introdutório sobre os conceitos básicos da linguagem, portanto teria mais facilidade para executar o projeto e aplicar as soluções.

2. Tratamento com relação a bugs

Atenção na hora de importar e exportar os arquivos JSON para evitar problemas com os caracteres especiais presentes no arquivo. Utilização de arredondamento de duas casas decimais ao realizar operações para obter os valores totais de estoque. Além disso, a inserção do atributo 'quantity' para os itens corrompidos ocorreu na mesma posição que os demais para facilitar futuras manipulações do arquivo de saída.

3. Funções implementadas:

3.1 ler_arquivo(nome_arquivo):

A função recebe o nome do arquivo que deverá ser manipulado em formato JSON e carrega esse arquivo em formato de um objeto do Python, uma lista de dicionários. Aqui foi passado como argumento que a codificação era UTF-8 para evitar problemas no carregamento dos dados.

3.2 corrigir_nomes(dados):

A função recebe os dados, percorre a lista e cada um dos caracteres de todos os nomes é verificado para os casos possíveis de erros e, caso confirmado algum erro, os caracteres são modificados para os originais. Os casos de erros são as substituições de "a" por "æ", "c" por "¢", "o" por "ø", "b" por "ß".

3.3 corrigir_precos(dados):

A função recebe um conjunto de dados e percorre os itens da lista em sequência. Caso seja verificado que o valor do preço está no formato texto, ele será formatado para o formato *float*, para contemplar os valores decimais.

3.4 corrigir quantidade(dados):

A função recebe como parâmetro os dados e verifica a existência do atributo ‘quantity’ para cada item da lista. Caso ele não seja encontrado, é definido um atributo ‘quantity’ com valor igual a zero.

3.5 exportar_arquivo(nome_arquivo):

A função recebe o nome do arquivo que será gerado em formato JSON e converte o objeto do Python em um arquivo. Aqui também é passado como argumento a codificação UTF-8. Definiu-se “ensure_ascii” como “false” devido a existência de caracteres especiais que não pertencem ao ASCII, como “ç” e caracteres com acento. Além disso foi definida a identação igual a 4 para facilitar a leitura do arquivo exportado.

3.6 imprime_lista(nome_arquivo):

A função recebe o nome do arquivo a ser lido em formato JSON, lê o arquivo corrigido através da função ler_arquivo() e ordenada a lista por categoria e id, respectivamente. Os nomes dos produtos dessa lista ordenada são impressos.

3.7 valor_total(nome_arquivo):

A função recebe o nome do arquivo a ser lido em formato JSON e lê o arquivo corrigido através da função ler_arquivo(). A lista carregada é percorrida em sequência. Para cada item da lista é verificado se o valor do atributo ‘category’ já existe como um atributo num novo dicionário criado. Em caso negativo um novo atributo é definido. Ao mesmo tempo, o resultado da multiplicação dos valores dos atributos ‘quantity’ por ‘price’ é adicionado como valor de seu respectivo atributo. Ao final, a função retorna o valor total do estoque por categoria na forma de um dicionário.

4. Referências

Documentação do Python (docs.python.org)

Livro de Charles R. Severance (Python for Everybory)

Stack Overflow (stackoverflow.com)