Python - Skill Test

Subject:	Python - Skill Test
Date:	March 08, 2022
Version:	1.3

I INTRODUCTION

The objective of this task is to test the ability of the candidate to write code fragments in several areas of programming that represent some StrataVAR typical use cases.

The development can be done using Python 2.7.x or 3.x, with an IDE at the programmer's choice (preferably PyCharm free community edition)

There is no time limit to complete the tasks, although time will be measured.

II. GENERAL GUIDELINES

(1) Develop the solution as Class/Method and use code to test various scenarios.

(2) Write and comment a few scenarios. As "Unit Test" utilizing the Class/Method you developed.

(3) Each test should be document according to best-practice and at a logical level (I.E., do not state the obvious, like when the code says.

```
i = i +1; //increment I by one – this comment is not adding any information, hence bad)
```

(4) In addition, use-best practices when writing the code in terms of method comments, Python constructs use etc.

(5) Feel free to search Google. Make sure you know to explain what and how was used.

(6) The program must actually run under your choice of IDE and produce output as required.

1. EXCEL PARSING - TEST#1

(1) The objective is to read, and parse a given Excel file, using the "xlrd" module. **Note**:- If you are using Python3 then use xlrd==1.2.0
(2) The result should be nicely printed as a Dict (JSON)
(3) The input Excel has two parts: The header which is built like a form with labels and values, followed by rows with several columns.
(4) Each label should be a key in the output Dict, with the corresponding value.
(5) The rows should be a list element in the DIct, where each item should have the column header as the key.
(6) There is a list of fields that should be extracted from the header part, and a list of fields from the columns. Fields not on the list should be ignored.

(7) Date values (label has the string somewhere in it) should be converted to the forma “yyyy-mm-dd”

(8) Header list to be extracted:
   (a) The code should look for form field labels called “Quote Number”, “Date”, “Ship To”, “Ship From”, “Name”.
   (b) The value is usually to the right of the label.
   (c) There is one exception, and that is “Name”, that both label and value are stored in the same Excel cell with a colon as a separator (like “Name: StrataVAR”).

(9) Item columns to be extracted: LineNumber, PartNumber, Description, Price. Other columns should be ignored.

(10) Incase an expected field is not found, print a detailed error message to the console, continue the process nevertheless.

The items have a heading row that always start with ‘LineNumber” field. The list of items ends when either the Excel file is exhausted or a separator that has at least 10 dashes (‘----------') is found in first column

Sample output:

```
{
     'Quote': '1234',
     'Date': '2019-03-31',
     'Items':
          [
               {
                    'LineNumber': 1,
                    'PartNumer': 'ABC',
                    'Description': 'very good',
                    'Price': 123.45
               },
               {
                    'LineNumber': 2,
                    'PartNumer': 'XYZ',
                    'Description': 'Not very good',
                    'Price': 678.9
               }
          ]
}
```

2. DISTI QUOTE MERGE SIMULATION– TEST #2

Please write a Python class and a Test Class that resolves the following problem.
The sample data should be used in the Test Class. The method should be generic and receive the data as input and return the result. The test class should use python's print function to print the Unified List table. You can Copy/Paste the result from the log.
Create hard coded test data, and show as output the "Unified List"
No need for fancy printout. A simple print will suffice.

The Input:

1. 2 lists of key: value pairs called "BoM" and "Disti"
2. Each list element has "Part Number" as a key and "Quantity" as a number.
3. Part Number may appear multiple times in the BoM with an arbitrary Quantity (including zero)
4. Part Number will appear only once in Disti, in other words, Quantities will be aggregated.
5. Part Numbers may exist only in BoM or only in Disti (there is no quantity zero)

The purpose is to "intelligently" compare the BoM and Disti, after fixing the Disti Quantity if needed, and showing items missing on each side.

The Expected Result.

1. Create a 3rd list, with 5 fields: bom_pn, bom_qty, Dsti_pn, Dsti_qty, Error Flag.
2. All items from the BoM will appear as is and based on the same order of the input BoM.
3. Against each bom_pn there should be a Dsti_pn (if one exists). If Disti PartNumber is missing a Part Number in Disti, the Disti side should be left blank and Error Flag should be checked
4. It is possible that Disti will be missing PN or BOM will be missing some Part Numbers
5. If Disti Quantity is bigger than BoM Quantity (for any given Part Number), Disti line should be **split** to match the Quantity of the BoM, and the remainder should be aligned with another BoM line, if exists.
6. If there is another record with same part number, the above should be repeated.
7. The joint list should be as similar as possible in term of quantities.
8. You may end up with some lines that do not have a Disti values, and vice-a-versa.
9. No matter what, the overall quantities per Part Number on BoM side in the Unified List output table should be the same as the original BoM. Same for Disti.

10. The Disti Side, as a whole, should include ALL items from original Disti, even if lines were split.

Test Data

Below is the test data you **must** use (build hard coded list of and the expected result of the algorithm. The code must actually run and produce results using print function. Key:Value pairs)

This will be tested against some hidden test cases at our end.

Develop the class/methods and a unit test with the sample data given below.

*The Test Data below **must** be used in the test class.*

BoM						
Part Number	**Quantity**					
ABC	2					
XYZ	1					
IJK	1					
ABC	1					
IJK	1					
XYZ	2					
DEF	2					
Disti						
Part Number	**Quantity**					
XYZ	2					
GEF	2					
ABC	4					
IJK	2					
Unified List						
BoM PN	**BoM Qty**		**Disti PN**	**Disti Qty**		**Error Flag**
ABC	2		ABC	2		
XYZ	1		XYZ	1		
IJK	1		IJK	1		
ABC	1		ABC	1		
IJK	1		IJK	1		
XYZ	2		XYZ	1		X
DEF	2					X
			GEF	2		X
			ABC	1		X

GOOD LUCK!