At KPI Sense we ingest data from multiple sources to be used throughout the platform. Today you will be building a command line script that will read an excel file and parse it into a standard format.

Task One:
Parse the KPI Dashboard sheet inside of the Demo_Assessment_Model_08.18.20.xlsx file and extract all the source data information and return a JSON document matching our standard format for the source object laid out below.

Background:

When importing data we have the concept of **sources** and **categories** this allows us to ingest data from a source object and have it immediately available to use on the front end without writing any additional front end logic or integration.

These models can get very complicated but for this scenario let's say the desired schema for category looks like this.

```
// Category schema
{
 "name": string,
 "fields": [] -> array of string,

 "subsets": [] -> array of string,
 "start_date": datetime,
 "end_date": datetime
}
```

Below is an demo example of a full category object

```
{
      "name": "Summary Financial Metrics",
      "fields": [
        "Subscription Revenue",
        "Services Revenue",
        "Total Revenue",
        "Cost of Goods Sold - COGS",
      ],
      "subsets": [ "all"],
      "start_date": "2018-01-01T00:00:00.000+00:00",
```

```
      "end_date": "2020-05-31T00:00:00.000+00:00"
    }
```

For this scenario we can say the **source** object schema should look like the following

```
{
   "source": string,
   "categories": [] -> an array of non subset category objects
}
```

Notes:

1. For this the source can just be KPI Dashboard it is okay to hardcode this
2. Row labels will always be in the “c” column if the rest of the column values for that row are date labels then that row can be considered a category and all rows below that row until you reach the next category are considered fields inside of the category.

For example in the demo excel file row 5 is the first category with a label of “Summary Financial Metrics” and everything below that until you get to the next category “Customer Metrics” is considered a field so the parsed JSON for this category would be

```
{
 "name": "Summary Financial Metrics",
 "fields": [
   "Subscription Revenue",
   "Services Revenue",
   "Total Revenue",
   "Cost of Goods Sold - COGS",
   "Gross Margin $",
   "Gross Margin %",
   "Sales & Marketing",
   "Research & Development",
   "General & Administrative",
   "Total Operating Expenses",
   "EBITDA",
   "Check",
   "Cash Balance at End of Month",
   "Change in Cash Balance"
 ],
 "subsets": [
   "all"
 ],
```

```
 "start_date": "2018-01-01T00:00:00.000+00:00",
 "end_date": "2020-05-31T00:00:00.000+00:00"
}
```

By default all categories should have a subset of “all” which can be hardcoded in as well for now which will represent a category and values that don't have a subset label mentioned below.

A category is considered a subset if the row is determined to be a category row **and** in column “b” there is a label. If this is the case then column “b” is considered to be the subset name and the label in column “c” is considered to be the parent category (This is especially important for part 2)

For example: In the demo file on row 68 there is a category row with label “Customer Metrics” this has a subset name of “Higher Ed” This means inside of the “Customer Metrics” category object inside of the subsets array we will add a “Higher Ed” entry. You can assume all subcategories will have the same exact fields as the parent category so nothing will need to be added to the fields array.

Your extracted data should match the contents of part_1_expected_extracted.json file. Don’t hard code any row number values assume you have no idea which rows are considered categories.

Task Two:
After you finish task one now extend the category model to include a **data** field which will be an array of value objects for the fields in that category

Here is an example of what the value object schema should look like.

```
{
 "date": datestring,
 "values": [
   {
     "name" fieldName,
     "subset": subset value belongs to,
     "value": raw value of cell
   },
   etc….
 ]
}
```

Here is an example of what the parsed data entry should look like for jan-01 entry for Category Customer Metrics with the Field Opening Customers


```
{
 "date": "2018-01-31T00:00:00+00:00",
 "values": [
   {
     "name": "Opening Customers",
     "subset": "all",
     "value": 32
   },
   {
     "name": "Opening Customers",
     "subset": "Higher Ed",
     "value": 27
   },
   {
     "name": "Opening Customers",
     "subset": "OPM",
     "value": 5
   }
 ]
}
```


Notice there is one value for each subset the first value of 32 is coming from row # 25
The second value of 27 is coming from the subset row # 69
The third value of 5 is coming from the subset row # 105

Submission:

Write code using Python3 OOP best practices as well as unit tests. Push code to github repository with instructions on how to run the program as well as how to run the unit test and see test coverage.

Squash all commits so submission only has 1 commit hash.

Once finished email a link to final code repo for review to mhemmingsen@kpisense.com