Assignment – 3(part-B) Python

End Date: **25 July 2021 11.59 PM (NO SUBMISSIONS WILL BE CONSIDERED BEYOND THIS)**

- This assignment is an individual submission.
- This assignment is continuation to previous Python Part A assignment.
- Total 100 Marks with duration of 1 week.
- All submissions should be done as per instructions.
- All other conditions are open to your interpretations.
- NO SUBMISSIONS will be considered for evaluation past deadline

New Requirements:

1. Additional attribute **"chronic"** is added to the rule derivation in ***rules.json***. So "chronic" attribute is now part of the rule, which will derive the result tag.
2. New **"Status"** tag added. Rule applies only when **"status"** is Active.
3. For any given rule, evaluator can change status from **"Active"** to **"Inactive"**. Code should be able to dynamically read status and update Result's tag in **results.csv**. Evaluator will not change the alphabet case. It will be Active or Inactive.
4. If the status is **"Inactive"** or any rule combination not found in ***rules.json***, then result tag in results.csv must be updated as **"No Access"**
5. Evaluator can add new combination of rules (with same attributes) and mark the status as **"Active"**. Code must update results tag with right value of **"results"** attribute without any code modification.

Given a file ***rules.json*** with rules written in the following format:

```
[
  {
    "fields": {
      "profession": "Healthcare",
      "travel": "No",
      "symptomatic": "No",
      "chronic": "Yes"
    },
    "status": "Active",
    "results": "High Risk"
  },
  {
    "fields": {
      "profession": "Healthcare",
      "travel": "No",
      "symptomatic": "Yes",
       "chronic": "Yes"
```

```
  },
  "status": "Active",
  "results": "High Risk"
},
{
  "fields": {
    "profession": "Healthcare",
    "travel": "Yes",
    "symptomatic": "Yes",
     "chronic": "Yes"
  },
  "status": "Active",
  "results": "High Risk"
},
{
  "fields": {
    "profession": "IT",
    "travel": "No",
    "symptomatic": "No",
    "chronic": "Yes"
  },
  "status": "Active",
  "results": "Low Risk"
},
{
  "fields": {
    "profession": "IT",
    "travel": "Yes",
    "symptomatic": "No",
    "chronic": "Yes"
  },
  "status": "Active",
  "results": "Moderate Risk"
},
{
  "fields": {
    "profession": "IT",
    "travel": "Yes",
    "symptomatic": "Yes",
    "chronic": "Yes"
  },
  "status": "Active",
  "results": "High Risk"
},
{
  "fields": {
    "profession": "Police",
```

```
      "travel": "No",
      "symptomatic": "No",
      "chronic": "Yes"
    },
    "status": "In Active",
    "results": "Moderate Risk"
  },
  {
    "fields": {
      "profession": "Police",
      "travel": "No",
      "symptomatic": "Yes",
      "chronic": "Yes"
    },
    "status": "Active",
    "results": "High Risk",
  },
  {
    "fields": {
      "profession": "Police",
      "travel": "Yes",
      "symptomatic": "No",
      "chronic": "Yes"
    },
    "status": "Active",
    "results": "High Risk",
  },
  {
    "fields": {
      "profession": "Police",
      "travel": "Yes",
      "symptomatic": "Yes",
      "chronic": "Yes"
    },
    "status": "Active",
    "results": "High Risk"
  },
  {
    "fields": {
      "profession": "Healthcare",
      "travel": "No",
      "symptomatic": "No",
      "chronic": "No"
    },
    "status": "Active",
    "results": "High Risk"
  },
  {
```

```
    "fields": {
      "profession": "Healthcare",
      "travel": "No",
      "symptomatic": "Yes",
      "chronic": "Yes"
    },
    "status": "Active",
    "results": "High Risk"
  },
  {
    "fields": {
      "profession": "Healthcare",
      "travel": "Yes",
      "symptomatic": "Yes",
      "chronic": "No"
    },
    "status": "Active",
    "results": "High Risk"
  },
  {
    "fields": {
      "profession": "Healthcare",
      "travel": "Yes",
      "symptomatic": "No",
      "chronic": "No"
    },
    "status": "In Active",
    "results": "High Risk"
  },
   {
    "fields": {
      "profession": "IT",
      "travel": "Yes",
      "symptomatic": "Yes",
      "chronic": "Yes"
    },
    "status": "In Active",
    "results": "High Risk"
  },
  {
    "fields": {
      "profession": "IT",
      "travel": "No",
      "symptomatic": "No",
      "chronic": "No"
    },
    "status": "In Active",
    "results": "Low Risk"
```

```
  },
   {
    "fields": {
      "profession": "IT",
      "travel": "Yes",
      "symptomatic": "No",
      "chronic": "No"
    },
    "status": "Active",
    "results": "Moderate Risk"
  },
]
```

And a file ***data.csv*** that contains multiple records in the following format (Email, Profession, Travel, Symptomatic, Chronic):

Email	Profession	Travel	Symptomatic	Chronic	Status
a@b.com	Healthcare	No	No	No	Active
x@y.com	IT	Yes	No	No	Active
xyz@z.com	IT	Yes	Yes	Yes	In Active

Based on the rules and the incoming data from data file, create a new file ***results.csv*** in the following schema:

Email	Profession	Travel	Symptomatic	Chronic	Results
a@b.com	Healthcare	No	No	No	High Risk
x@y.com	IT	Yes	No	No	Moderate Risk
xyz@z.com	IT	Yes	Yes	Yes	No Access

Result is derived based on the values of the fields mentioned in the rules.

Instructions

1. *rules.json* is a static JSON file. That means for the submitted code, file content will not change. Rule list is exhaustive. Incoming data in *data.csv* will always have combination found in Rule JSON File
2. File format for Data file and New file with Result tag will be Comma Separated.
3. Ensure you lookup the rules file, match the incoming record data with right tag(as per the rules) and then insert the data into *results.csv* .

For example
data.csv has three columns (*Profession, Travel, Symptomatic*). Rules Tag is derived from these three keys from *rules.json* file. Once you get the tag, insert the Data record along with new Result column in *results.csv* file.

4. HINTS:
    a. You can create separate logic for deriving rules from JSON file and do look up on the rules when inserting data.
    b. You are free to generate any intermediary files.
    c. You are free to use any python libraries. (supplied with standard python installation)

Grading Criteria:

a. Able to **read** from RULES JSON file and **write** appropriate tag into Results file. (**25**)
b. Submit a Flow chart for code logic (**10**)
c. Accuracy of Result tag is 100%. Evaluators will test the code on a different Data file to evaluate the accuracy of Result's column. Students are required to create their own data file, however schema of data file should be unchanged (**25**)
d. Code is properly commented. Comments are inputted **#.** (**5**)
e. Rule logic to derive Tags from RULES JSON (**30**).
   Use correct conditional logic for deriving rule (**Strictly use If else**) – **15 marks**
   OR If you use Dictionary along with correct conditions – **30 marks**
   Else 0
f. Moodle submission format (**5**):

   <roll-number>.zip

   ------- lookup.py

   ------- rules.py ***(Code to derive Rules from JSON File)***

   ------- results.csv ***(Code to lookup on the rules and insert into Result's file)***

   ------- flow_chart (png/jpeg/jpg)

Note:

It is recommended that the code is flexible to accommodate change in specs in the future assignments. This is however not a criterion for grading.