# **Program to compare two JSONs**

# Objectives:

1. Do not use any 3rd party tools/libraries for this program, and use your own logic
2. Use Java/Python for development
3. The program should take two JSON files as inputs
4. The JSONs can have nested structures (as per standard JSON format)
5. Program should iterate over every {key, value} pair of first file and compare the same with the second file
6. Print the differences as
   1. Missing keys
   2. Different values in keys
   3. New keys
7. The program should have a provision to exclude certain keys from comparison
   1. This is useful when there is a timestamp or index field that can often change when a JSON is created with same fields
   2. What keys to be excluded can be inputted via a config file or at command line
   3. Example config file:

“exclude\_keys “ : [ “request.uuid” ]

“mask\_keys” : {

“request. originatorType” : “testxxxx”

}

1. There should be a way to mask certain part of the value of a given key
   1. Example, “95xxx” should match 95034

# Sample JSON inputs:

Input1.json

{

"workspace": "abc",

"request": {

"header": {

"uuid": "00000000-0000-0008-6cfc-3b60d1904348",

"workspace": "salesdata",

"ts": null,

"originatorType": "user",

}

}

}

Input2.json

{

"workspace": "a123",

"request": {

"header": {

"uuid": "00000000-0000-0008-6cfc-3b60d1900000",

"workspace": "salesdata",

"ts": null,

"originatorType": "testUser",

}

}

}

Example Input 2 (modify some fields for testing)

{

         "tokenPhrase": "TIME\_PERIOD",

         "level": 4,

         "wordPos": 1,

         "tokenValues": {

            "12": [

               {

                  "tokenType": "PAST",

                  "value": "last",

                  "word": "last",

                  "wordPos": 1,

                  "opt": false,

                  "matchedNode": 12,

                  "terminal": false,

                  "valueKey": null,

                  "next": null,

                  "valueTypeList": false

               }

            ],

            "14": [

               {

                  "tokenType": "TIME",

                  "value": "YEAR",

                  "word": "year",

                  "wordPos": 2,

                  "opt": false,

                  "matchedNode": 14,

                  "terminal": true,

                  "valueKey": "unit",

                  "next": null,

                  "valueTypeList": false

               }

            ]

         },

         "valueTokenType": "NUMBER",

         "nodeId": 14,

         "values": {

            "unit": "YEAR",

            "number": 1

         },

         "wordPosOfValues": [

            2,

            2

         ]

      }