We have signed a new account - Daikibo Industrials, a global leader in the manufacturing of heavy machinery, founded and headquartered in Tokyo, Japan. They needed assistance with a variety of problems and were impressed to find out Deloitte could help in all verticals.

Daikibo is in the process of integrating IIoT (Industrial Internet-of-Things) devices to monitor, measure and analyze their manufacturing processes. Half of their infrastructure uses devices streaming telemetry data in one format, and the other half - in another. They need your help to combine the two.

In [1]:
```
import json, unittest, datetime
import pandas as pd
```

In [2]:
```
with open("data-1.json","r",  encoding="utf8") as f:
     jsonData1 = json.load(f)
```

In [3]:
```
jsonData1
```

Out[3]:
```
{'deviceID': 'dh28dslkja',
 'deviceType': 'LaserCutter',
 'timestamp': 1624445837783,
 'location': 'japan/tokyo/keiyō-industrial-zone/daikibo-factory-meiyo/section-1',
 'operationStatus': 'healthy',
 'temp': 22}
```

In [4]:
```
with open("./data-2.json","r",encoding="utf8") as f:
    jsonData2 = json.load(f)
```

In [5]:
```
jsonData2
```

Out[5]:
```
{'device': {'id': 'dh28dslkja', 'type': 'LaserCutter'},
 'timestamp': '2021-06-23T10:57:17.783Z',
 'country': 'japan',
 'city': 'tokyo',
 'area': 'keiyō-industrial-zone',
 'factory': 'daikibo-factory-meiyo',
 'section': 'section-1',
 'data': {'status': 'healthy', 'temperature': 22}}
```

In [6]:
```
with open("./data-result.json","r",encoding="utf8") as f:
    jsonExpectedResult = json.load(f)
```

In [7]:
```
jsonExpectedResult
```

Out[7]:
```
{'deviceID': 'dh28dslkja',
 'deviceType': 'LaserCutter',
 'timestamp': 1624445837783,
 'location': {'country': 'japan',
  'city': 'tokyo',
  'area': 'keiyō-industrial-zone',
  'factory': 'daikibo-factory-meiyo',
  'section': 'section-1'},
 'data': {'status': 'healthy', 'temperature': 22}}
```

In [8]:
```
def convertFromFormat1 (jsonObject):

    jsonData1["deviceID"]
    jsonData1['deviceType']
    jsonData1['timestamp']

    return NotImplemented
```

In [9]:
```
def convertFromFormat2 (jsonObject):

    # IMPLEMENT: Conversion From Type 1

    return NotImplemented
```

In [10]:
```
def main (jsonObject):

    result = {}

    if (jsonObject.get('device') == None):
        result = convertFromFormat1(jsonObject)
    else:
        result = convertFromFormat2(jsonObject)

    return result
```

In [11]:
```
class TestSolution(unittest.TestCase):

    def test_sanity(self):

        result = json.loads(json.dumps(jsonExpectedResult))
        self.assertEqual(
            result,
            jsonExpectedResult
        )

    def test_dataType1(self):

        result = main (jsonData1)
        self.assertEqual(
            result,
            jsonExpectedResult,
            'Converting from Type 1 failed'
        )

    def test_dataType2(self):

        result = main (jsonData2)
        self.assertEqual(
            result,
            jsonExpectedResult,
            'Converting from Type 2 failed'
        )

if __name__ == '__main__':
    unittest.main()
```

```
E
======================================================================
ERROR: C:\Users\Dennis\AppData\Roaming\jupyter\runtime\kernel-53456077-3b5d-446b-a0db-93a6511b6971 (unittest.loader._FailedTest)
----------------------------------------------------------------------
AttributeError: module '__main__' has no attribute 'C:\Users\Dennis\AppData\Roaming\jupyter\runtime\kernel-53456077-3b5d-446b-a0db-93a6511b6971'

----------------------------------------------------------------------
Ran 1 test in 0.002s

FAILED (errors=1)
An exception has occurred, use %tb to see the full traceback.

SystemExit: True
C:\ProgramData\Anaconda3\lib\site-packages\IPython\core\interactiveshell.py:3445: UserWarning: To exit: use 'exit', 'quit', or Ctrl-D.
  warn("To exit: use 'exit', 'quit', or Ctrl-D.", stacklevel=1)
```

In [ ]: