**Assessment 3**  ***Water Tank Level and Light Intensity***

import time

import sys

import ibmiotf.application

import ibmiotf.device

import random

import json

#Provide your IBM Watson Device Credentials

organization ="o349yq"

deviceType = "Water"

deviceId = "1000"

authMethod = "token"

authToken = "1234567890"

# Initialize the device client.

WL=0

LI=0

def myCommandCallback(cmd):

print("Command received: %s" % cmd.data['command'])

if cmd.data['command']=='switchon':

print("SWITCH ON IS RECEIVED")

elif cmd.data['command']=='switchoff':

print("SWITCH OFF IS RECEIVED")

if cmd.command == "setInterval":

if 'interval' not in cmd.data:

print("Error - command is missing required information: 'interval'")

else:

interval = cmd.data['interval']

elif cmd.command == "print":

if 'message' not in cmd.data:

print("Error - command is missing required information: 'message'")

else:

print(cmd.data['message'])

try:

deviceOptions = {"org": organization, "type": deviceType, "id": deviceId, "auth-method": authMethod, "auth-token": authToken}

deviceCli = ibmiotf.device.Client(deviceOptions)

#..............................................

except Exception as e:

print("Caught exception connecting device: %s" % str(e))

sys.exit()

# Connect and send a datapoint "hello" with value "world" into the cloud as an event of type "greeting" 10 times

deviceCli.connect()

while True:

W=29.58

L=35.46

#Send Water-Level & Light Intensity to IBM Watson

data =jsondata={"d":{ 'waterlevel': W, 'lightintensity': L}}

print (data)

def myOnPublishCallback():

print ("Published Water Level = %s %%" % W, "Light Instensity = %s %%" % L, "to IBM Watson")

success = deviceCli.publishEvent("Data", "json", data, qos=0, on\_publish=myOnPublishCallback)

if not success:

print("Not connected to IoTF")

time.sleep(1)

deviceCli.commandCallback = myCommandCallback

# Disconnect the device and application from the cloud

deviceCli.disconnect()



