# INTERNATIONAL SPACE STATION ASTRONAUTS INFORMATION AND ISS LOCATION

**(USING PYTHON)**

**PROJECT REPORT**
**Submitted for Google Developers' Group 2-Credit course**


**Submitted by:**
**Krishna Kumar Mahto**
**Registration no.- 16BIT0453**


## 1. Abstract:

The project is based on finding the current location (latitude and longitude points) of ISS overhead earth's map, and also finding the people who currently are in the ISS.

## 2. Introduction:

**Goal:** Implementing a software system that can give the names of people currently staying in the International Space Station, and also finding the current location of the ISS.

Space exploration was a dream once, now it has become a phenomenon, that it almost appears a trivial event. Technology has taken us so far that humans have started to live in space. Over the years, a number of space stations have been launched into orbit. Currently there are 3 stations orbiting the earth, one of which is the International Space Station.

Although it is easy to google to find information about ISS (and just about anything), it would be really cool to see something being done just by running an application. This is what is intended to be done through this project.

## 3. Methodology:

**3.1. Basic Concept:** We cannot directly access ISS data (for obvious reasons), however, they have been generous enough to have made available APIs to get some data from their database. We need two informations in order to achieve the goal of the project- names of astronauts currently in the ISS, and the current position of the ISS overhead earth's map, both of which have been made available as JSON API's. These APIs can be easily be used using Python.

API for astronaut details- http://api.open-notify.org/astros.json

API for ISS location details- http://api.open-notify.org/iss-now.json

The location of ISS is to be plotted on the map of the world, this will be achieved by using a Python library, called Turtle. The turtle module provides turtle graphics primitives, in both object-oriented and procedure-oriented ways. Because it uses

**Tkinter** for the underlying graphics, it needs a version of Python installed with Tk support.
The image of the world map that I have used has been taken from the official website of NASA which has been mapped with the latitude and longitude values to correct scale.

### 3.2. Code implemented and execution steps:

#### 3.2.1. The complete code

```
#!/bin/python3

import json
import urllib.request
import turtle

############### Printing people information ################
url = 'http://api.open-notify.org/astros.json'
response = urllib.request.urlopen(url)
result = json.loads(response.read())
# print(result)

print('People in Space: ', result['number'])

people = result['people']
# print(people)

for p in people:
  print(p['name'], ' in ', p['craft'])

############### Printing ISS location ################
url = 'http://api.open-notify.org/iss-now.json'
response = urllib.request.urlopen(url)
result = json.loads(response.read())

location = result['iss_position']
lat = location['latitude']
lon = location['longitude']
print('Latitude: ', lat)
print('Longitude: ', lon)

screen = turtle.Screen()
```

```
screen.setup(720, 360)
screen.setworldcoordinates(-180, -90, 180, 90)
screen.register_shape('iss.gif')

# image source:
# map.jpg: http://visibleearth.nasa.gov/view.php?id=57752 Credit: NASA
screen.bgpic('map.gif')

iss = turtle.Turtle()
iss.shape('iss.gif')
iss.setheading(90)
iss.penup()
iss.goto(float(lon), float(lat))
turtle.done()
```

### 3.2.2. Execution steps

#### 3.2.2.1. Importing libraries:

```
1. import json
2. import urllib.request
3. import turtle
```

#### 3.2.2.2. Interfacing with API for astronaut information:

```
1. url = 'http://api.open-notify.org/astros.json'
2. response = urllib.request.urlopen(url)
3. result = json.loads(response.read())
```

Line 2, `urllib.request.urlopen(url)` returns a valid JSON object which is stored in response variable.
Line 3, `json.loads(response.read())` returns a python dictionary which is obtained after decoding the JSON object (`response.read()` does that) and converting it into python dictionary (`json.loads(response.read())` does that.

#### 3.2.2.3. Printing astronauts' information:

```
print('People in Space: ', result['number'])
people = result['people']
for p in people:
  print(p['name'], ' in ', p['craft'])
```

On opening the API url, we see something like:

```
{
  "message": "success",
  "number": 3,
  "people": [
    {
      "name": "Anton Shkaplerov",
      "craft": "ISS"
    },
    {
      "name": "Scott Tingle",
      "craft": "ISS"
    },
    {
      "name": "Norishige Kanai,
      "craft": "ISS"
"
    }
  ]
}
```

The corresponding python dictionary structure obtained in result:

```
print(result) :

{
      'number': 3,
```

```
    'message': 'success',
    'people':
        [
        {
            'name': 'Anton Shkaplerov',
            'craft': 'ISS'
        },
        {
            'name': 'Scott Tingle',
            'craft': 'ISS'
        },
        {
            'name': 'Norishige Kanai',
            'craft': 'ISS'
        }
        ]
}
```

In Line 1, `result['number']`, therefore, returns the value corresponding to key 'number' in the Python dictionary 'result'.

In Line 2, `result['people']` similarly returns the corresponding list of dictionary with key-value pairs representing each astronaut's name and craft. This dictionary is saved in `people` variable.

The `for` loop through line 3 and line 4, simply iterate through the `people` variable, printing the contents through `print(p['name'], ' in ', p['craft']).`

### 3.2.2.4. Printing latitude and longitude of ISS over the earth:

```
1. url = 'http://api.open-notify.org/iss-now.json'
2. response = urllib.request.urlopen(url)
3. result = json.loads(response.read())
4. location = result['iss_position']
5. lat = location['latitude']
6. lon = location['longitude']
7. print('Latitude: ', lat)
8. print('Longitude: ', lon)
```

On opening the API url, we see something like:


```
{
      "timestamp": 1521396584,
      "message": "success",
      "iss_position":
            {
                  "longitude": "-173.4266",
                  "latitude": "17.2913"
            }
}
```


The corresponding python dictionary:


```
{
      'timestamp': 1521396773,
```

```
    'message': 'success',
    'iss_position':
    {
        'longitude': '-166.2024',
        'latitude': '7.8235'
    }
}
```

Following the normal conversion of JSON object to Python dictionary, lines 5 and 6 are used to store the latitude and longitude values in `lat` and `lon` variables, which are printed in lines 7 and 8.

### 3.2.2.5. Plotting the latitudes and longitudes on the map

```
1. screen = turtle.Screen()
2. screen.setup(720, 360)
3. screen.setworldcoordinates(-180, -90, 180, 90)
4. screen.register_shape('iss.gif')
5. screen.bgpic('map.gif')
6. iss = turtle.Turtle()
7. iss.shape('iss.gif')
8. iss.setheading(90)
9. iss.penup()
10. iss.goto(float(lon), float(lat))
11. turtle.done()
```

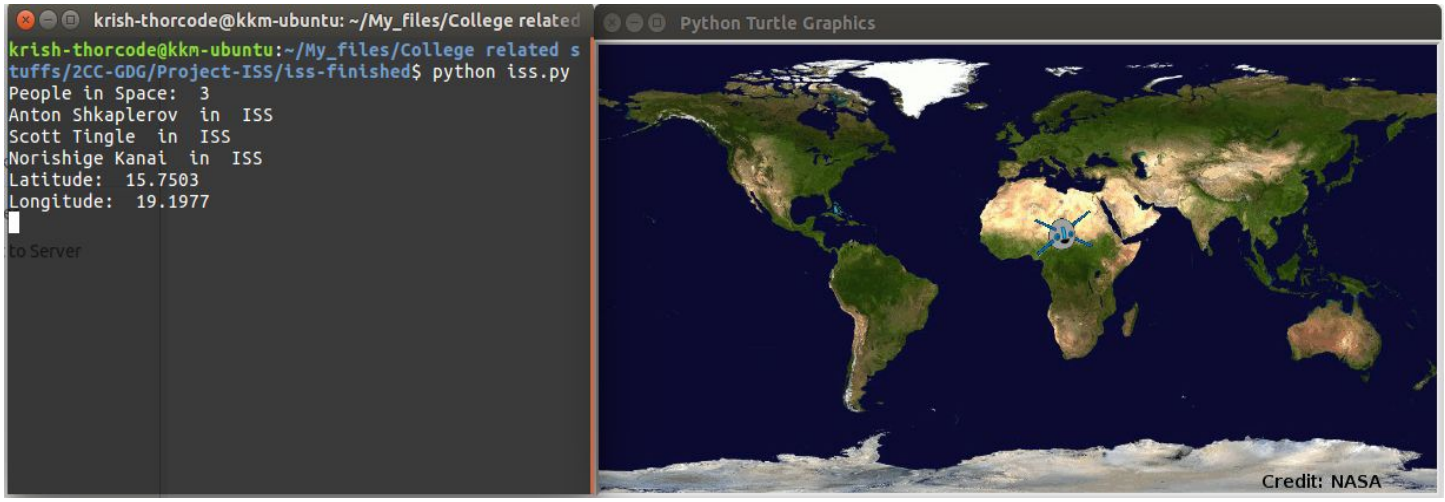
Line 1 draws a TurtleScreen object the turtle is supposed to draw on. This object is stored in `screen` variable. Line 2 sets up the window size of the `screen` object. Line 3 sets up the user defined coordinates which are convenient for the current problem. Line 4 registers a shape that can be laid on the turtle screen. Line 5 is self-explanatory, as it sets background image of the window.

Line 6 defines the Turtle object that'll draw the shapes on the screen object.
Line 7 sets up the shape of the turtle.
Line 8 sets the direction in which turtle will be facing its head.
In line 9, the `iss.penup()` command makes sure that following movements done by the turtle won't draw anything on the screen.
Next, line 10 simply takes the turtle to the latitudinal and longitudinal location of the International Space Station, which therefore, makes out a plot on the map of the world of where ISS is present now.
The last line calls the Tkinter mainloop() function that prevents the window screen close after the execution gets over.

## 4. Result

### 4.1. The output

When the code given in *3.2.1* is executed, the output that I got is as expected, the snapshot of the same is as following:



### 4.2. Difficulties faced:

- Limited knowledge of Python:
  I am not as fluent in Python, never really use it. I had ideas of how easy it makes life of the coders, but still never really used Python for anything.

- Lack of well-explained documentation of the python turtle library and/or community support.

- Used a real JSON serialised data in a real app for the first time.

- Never used API's before. API's were just a theoretical concept, so had to learn how to work with them in the code.

## 5. CONCLUSION:

The information of astronauts currently staying in the ISS and the current location of the ISS were successfully obtained by using respective APIs and decoding them using python, and plotting them using python library (turtle).

## 6. REFERENCES:


[1]
https://docs.python.org/3.3/library/turtle.html?highlight=turtle#turtle.setheading
(turtle documentation)

[2]
https://docs.python.org/3/library/json.html
(python3 json encoding/decoding)

[3]
https://docs.python.org/3/library/urllib.html
(python3 urllib)