CHRIST
(DEEMED TO BE UNIVERSITY)
DELHI-NCR, INDIA

Advance Python Programming

MCA-372

Assignment – 06

BY

HIMANSHU HEDA (24225013)

SUBMITTED TO

Dr. Manjula Shannhog

SCHOOL OF SCIENCES

2024-25

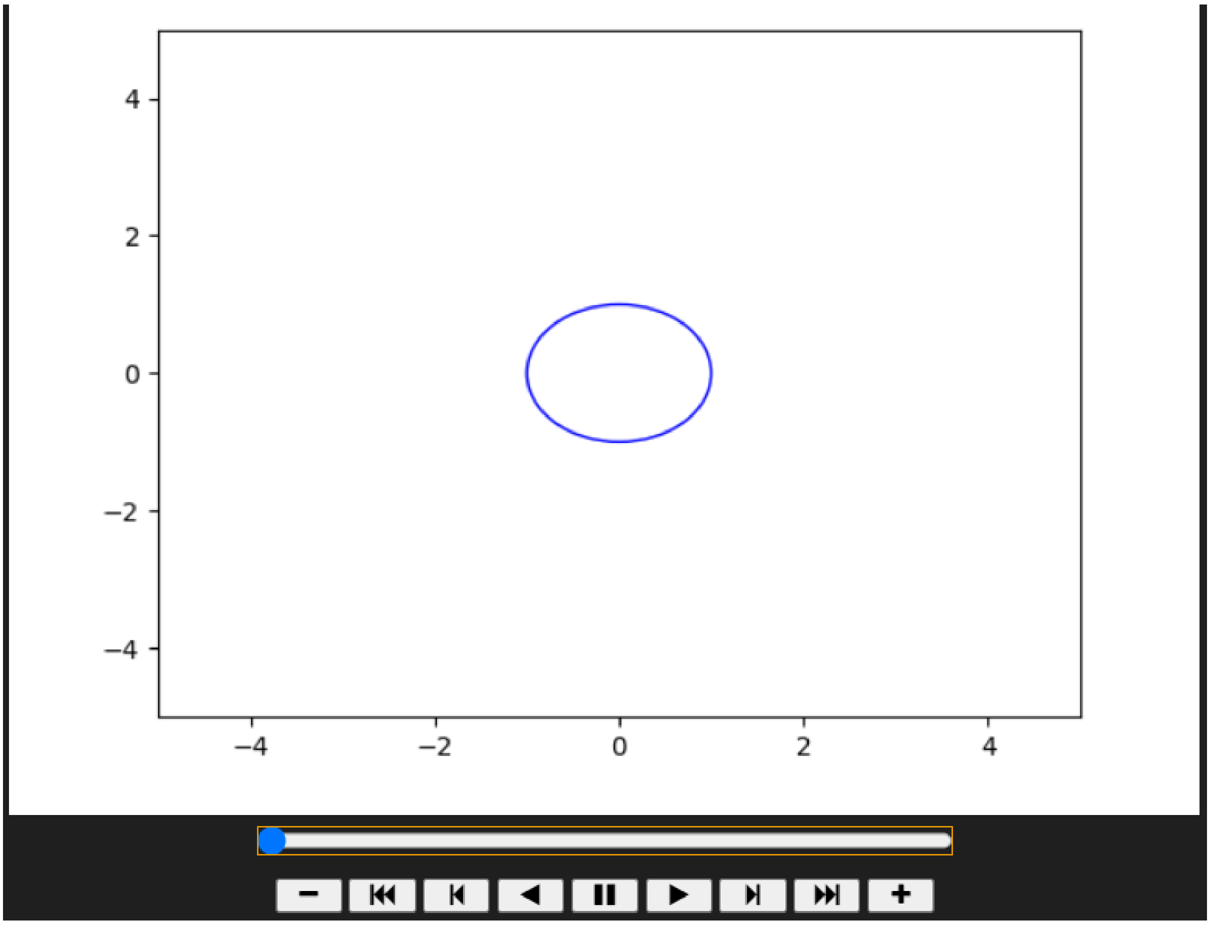
Importing The Libraries

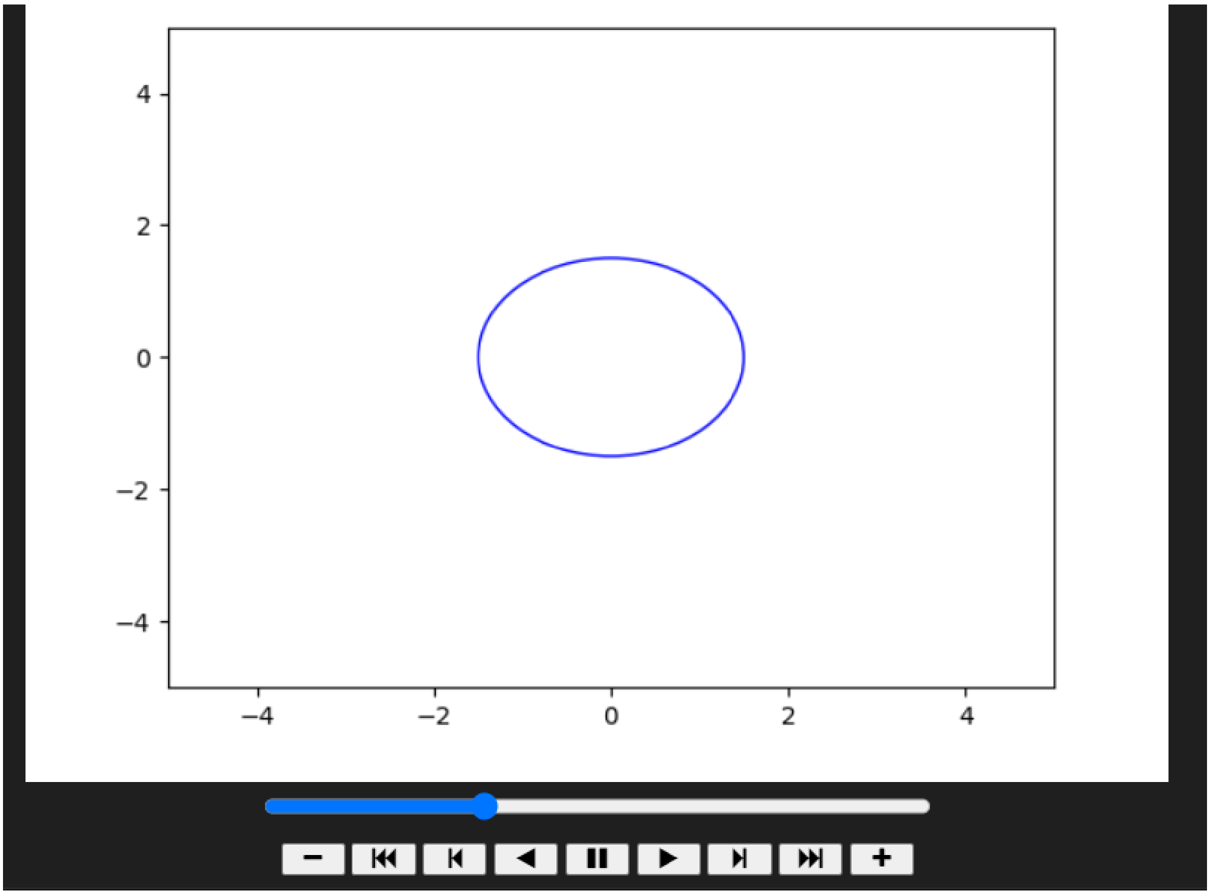
```
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.animation as animation
from IPython.display import HTML
```

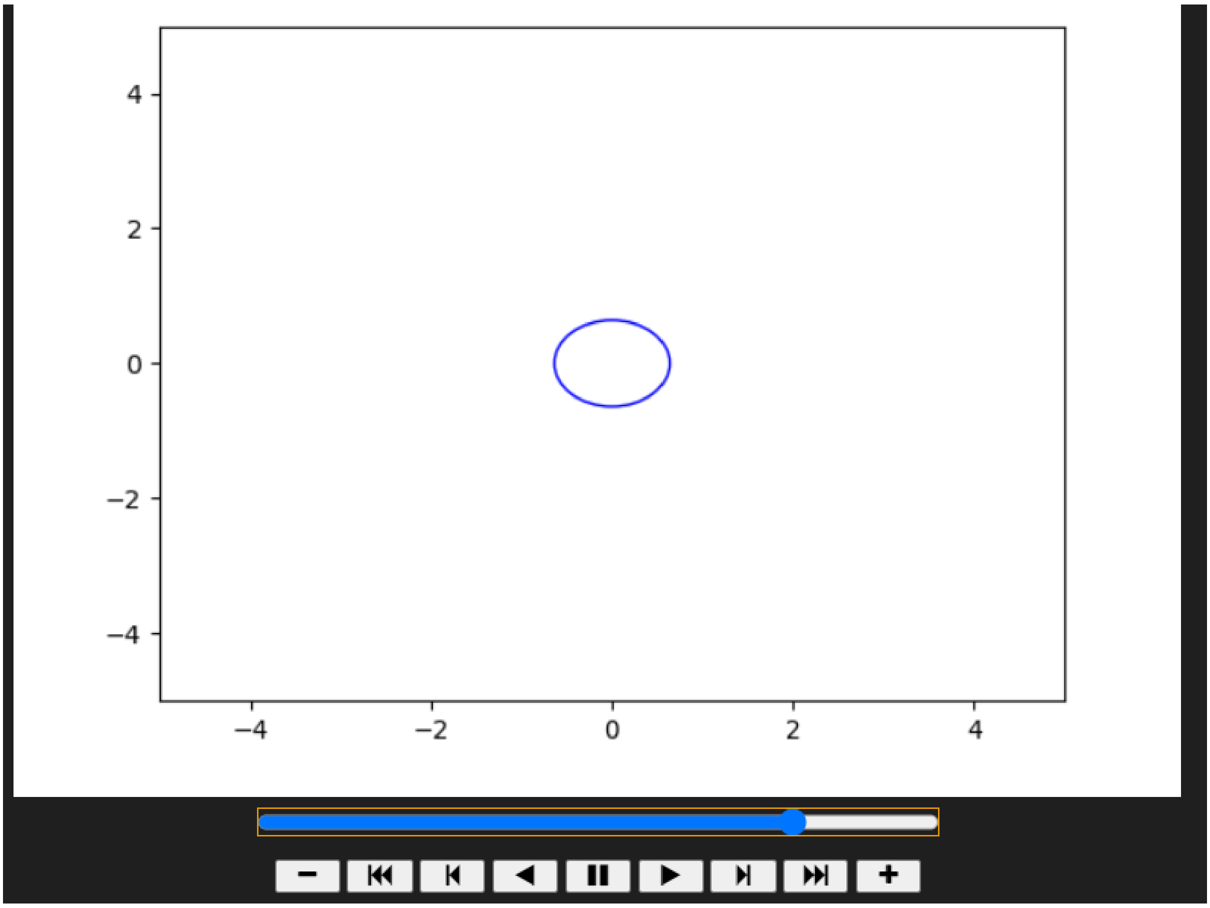
Sol 1 : -- Lets Animate the Circle With the Help of Animation Function

```
plt.ioff()

fig, ax = plt.subplots()
ax.set_xlim(-5, 5)
ax.set_ylim(-5, 5)
circle = plt.Circle((0, 0), 1, fill=False, edgecolor='blue')
ax.add_patch(circle)

def update(frame):
    circle.set_radius(1 + 0.5 * np.sin(frame / 20))
    return circle,

ani = animation.FuncAnimation(fig, update, frames=np.arange(0, 100, 1),
interval=50, blit=True)
HTML(ani.to_jshtml())
```

Output:


4
2
0
−2
−4
−4
−2
0
2
4
−
+


4
2
0
−2
−4
−4
−2
0
2
4
−
+

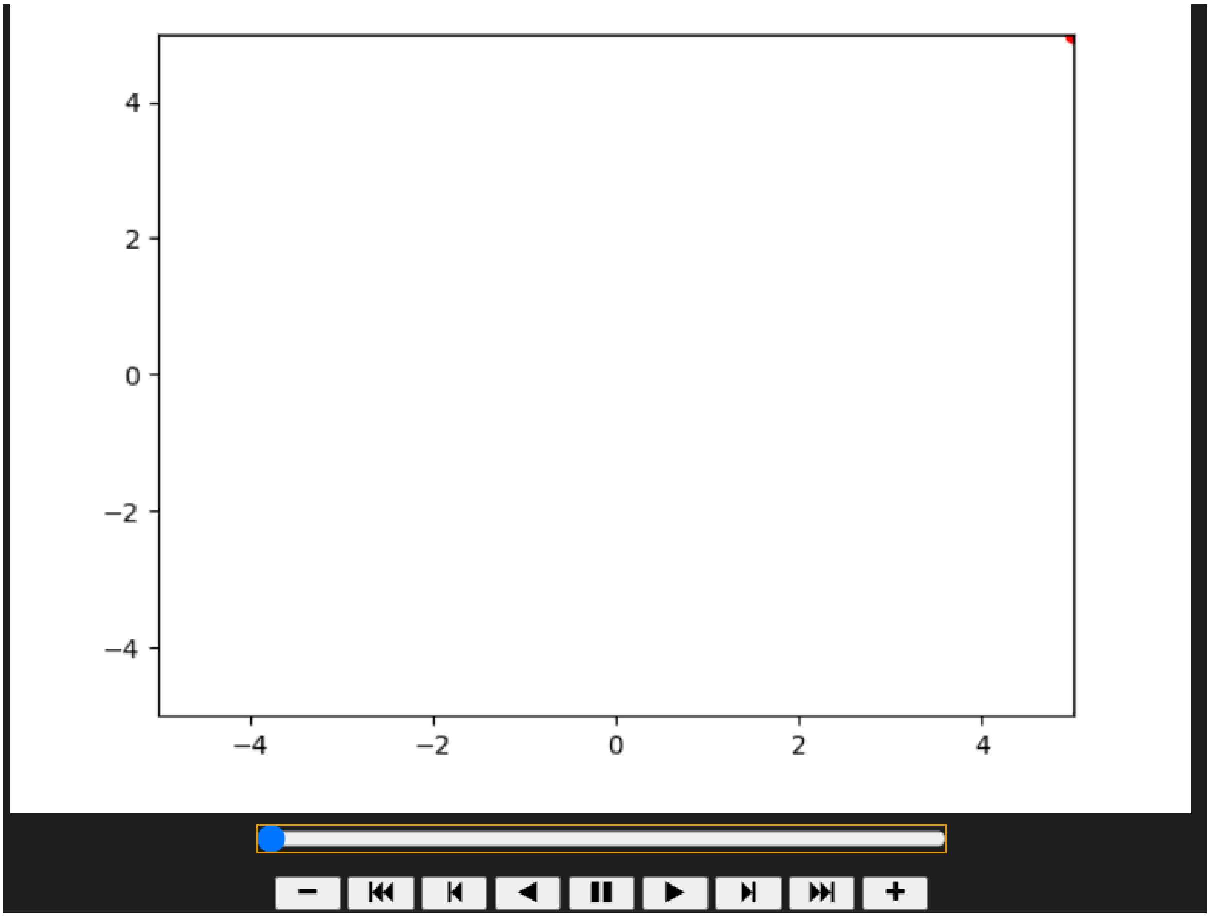


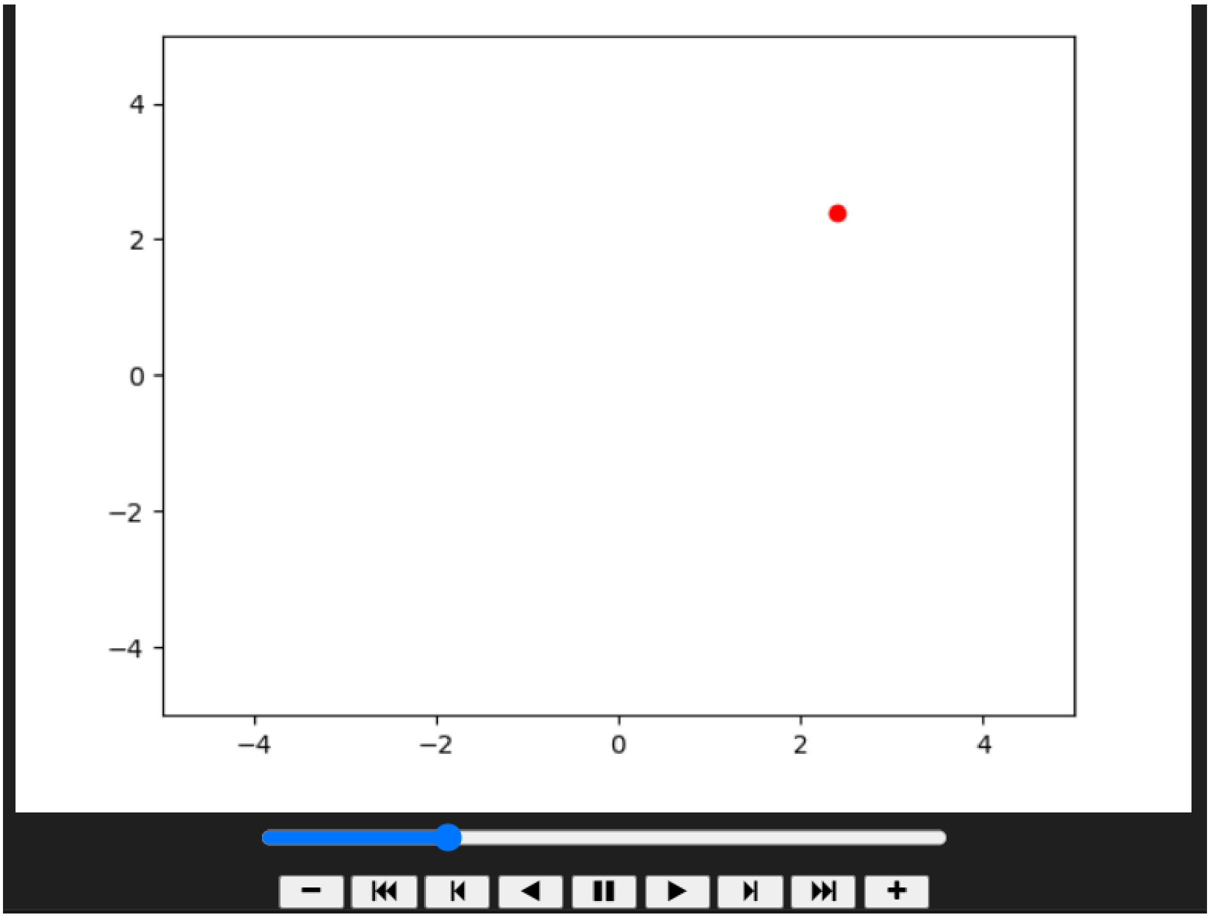
4
2
0
−2
−4
−4
−2
0
2
4
−
⏮
⏴
◀
⏸
▶
⏵
⏭
+

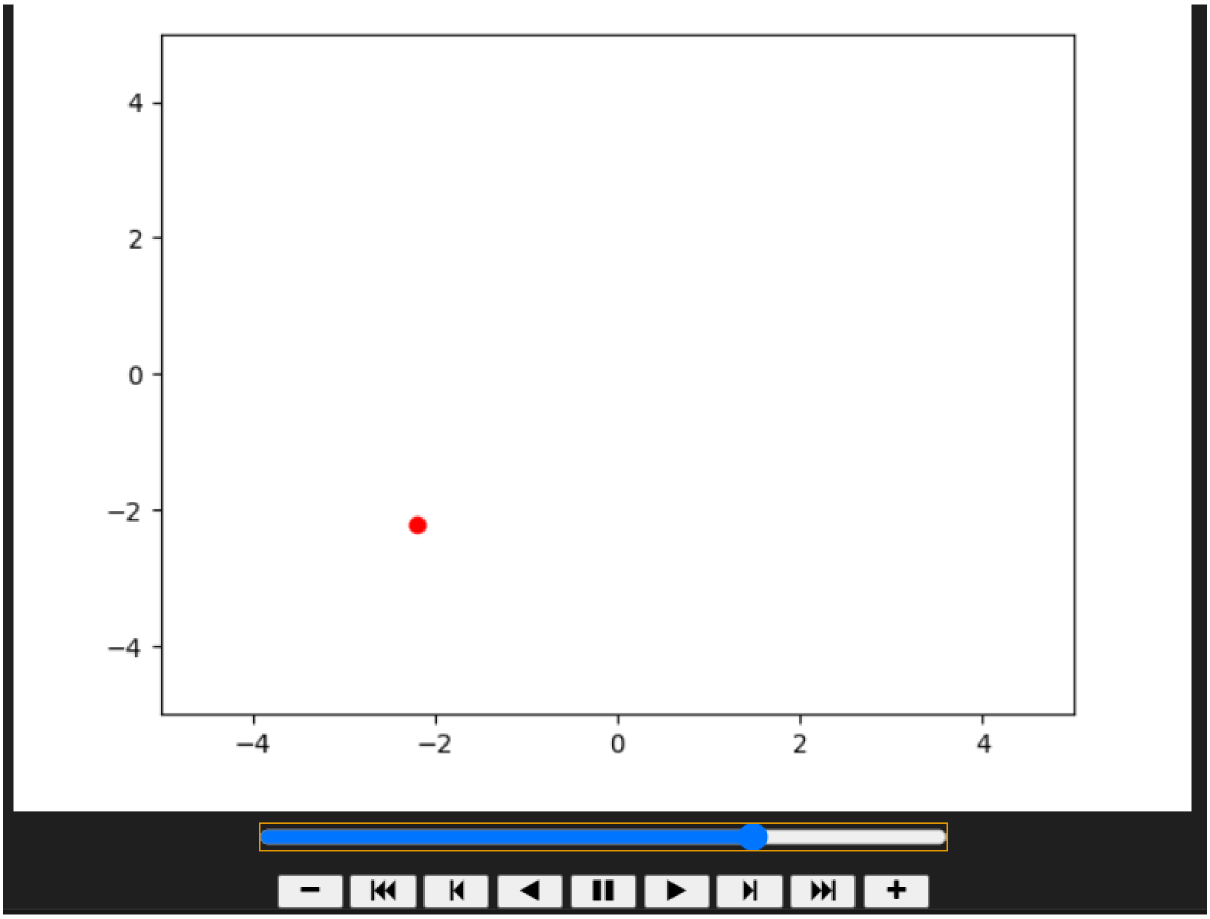
Sol 2 :-- Lets Animate the Dot animation which move from right to left Diagonal

```
plt.ioff()

fig, ax = plt.subplots()
ax.set_xlim(-5, 5)
ax.set_ylim(-5, 5)
dot, = ax.plot([], [], 'ro')

def update(frame):
    x = 5 - frame / 10
    y = 5 - frame / 10
    dot.set_data(x, y)
    return dot,

ani = animation.FuncAnimation(fig, update, frames=np.arange(0, 100, 1),
interval=50, blit=True)
HTML(ani.to_jshtml())
```

Output :


4
2
0
−2
−4
−4
−2
0
2
4
− ⏮ ⏴ ◀ ⏸ ▶ ⏵ ⏭ +

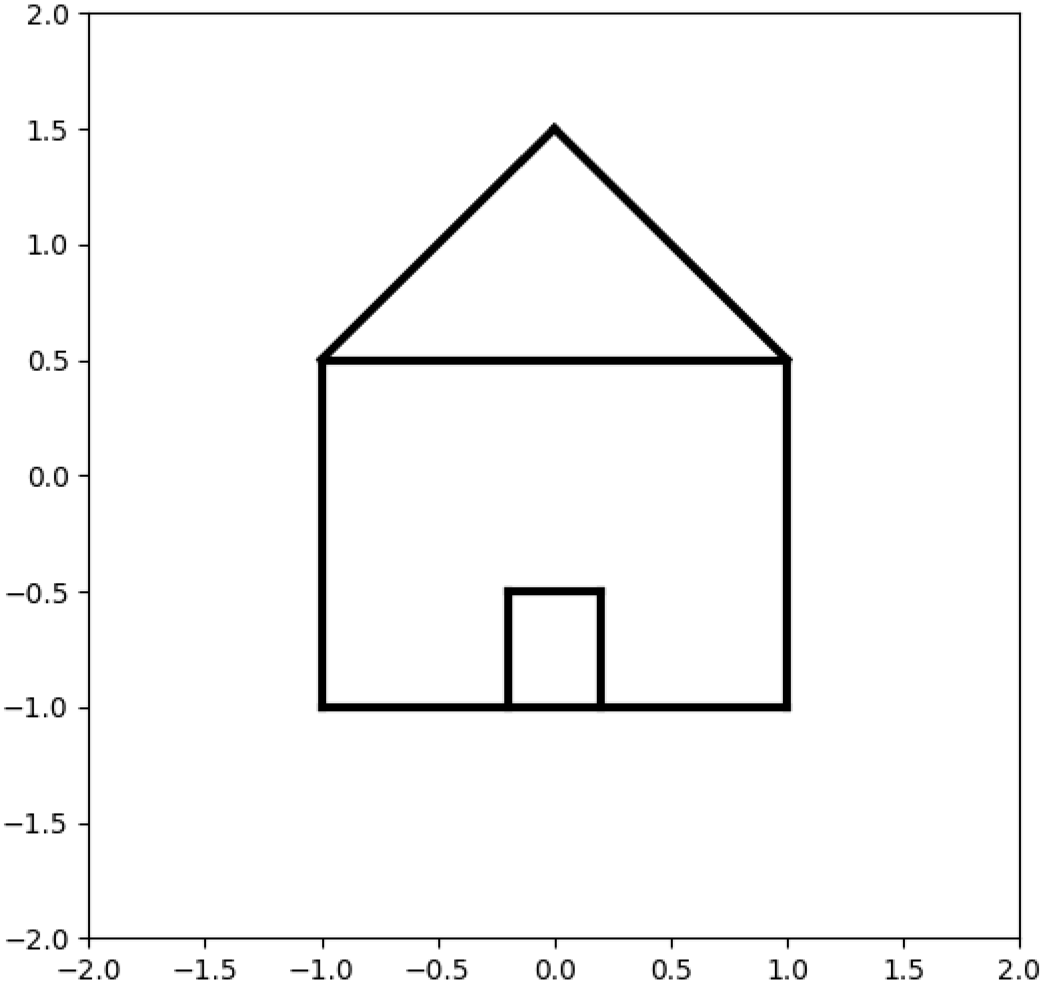

4
2
0
−2
−4
−4
−2
0
2
4
− ⏮ ⏴ ◀ ⏸ ▶ ⏵ ⏭ +


4
2
0
−2
−4
−4
−2
0
2
4
−
+

Sol 3:-- Lets Draw the House

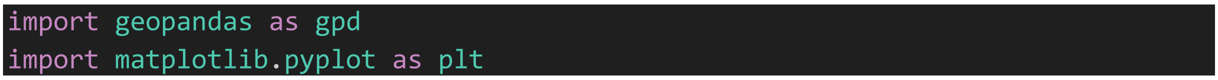
```
import matplotlib.pyplot as plt

plt.ioff()

fig, ax = plt.subplots(figsize=(6, 6))
ax.set_xlim(-2, 2)
ax.set_ylim(-2, 2)

# Draw the house
plt.plot([-1, 1], [-1, -1], 'k-', linewidth=3)  # Bottom line
plt.plot([-1, -1], [-1, 0.5], 'k-', linewidth=3)  # Left wall
plt.plot([1, 1], [-1, 0.5], 'k-', linewidth=3)  # Right wall
plt.plot([-1, 1], [0.5, 0.5], 'k-', linewidth=3)  # Top of house
plt.plot([-1, 0], [0.5, 1.5], 'k-', linewidth=3)  # Left roof
plt.plot([1, 0], [0.5, 1.5], 'k-', linewidth=3)  # Right roof

# Draw the door
plt.plot([-0.2, -0.2], [-1, -0.5], 'k-', linewidth=3)  # Left side of door
plt.plot([0.2, 0.2], [-1, -0.5], 'k-', linewidth=3)  # Right side of door
plt.plot([-0.2, 0.2], [-0.5, -0.5], 'k-', linewidth=3)  # Top of door

plt.show()
```

Output


2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
−2.0
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0

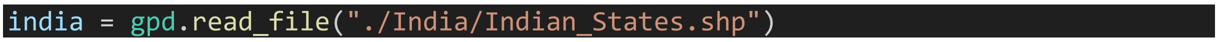
Sol 4 :-- Plot all the states Except Rajasthan and Gujarat

Importing the Libraries



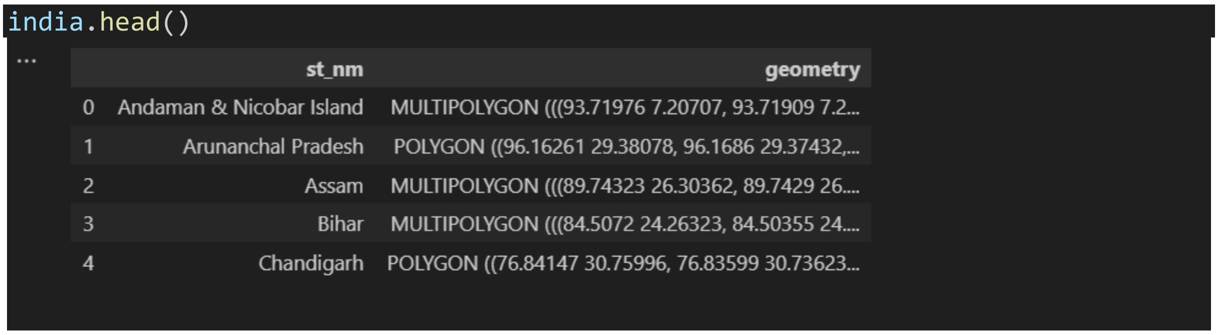
```
import geopandas as gpd
import matplotlib.pyplot as plt
```

Lets Load the Shape File to Fetch the Data



```
india = gpd.read_file("./India/Indian_States.shp")
```

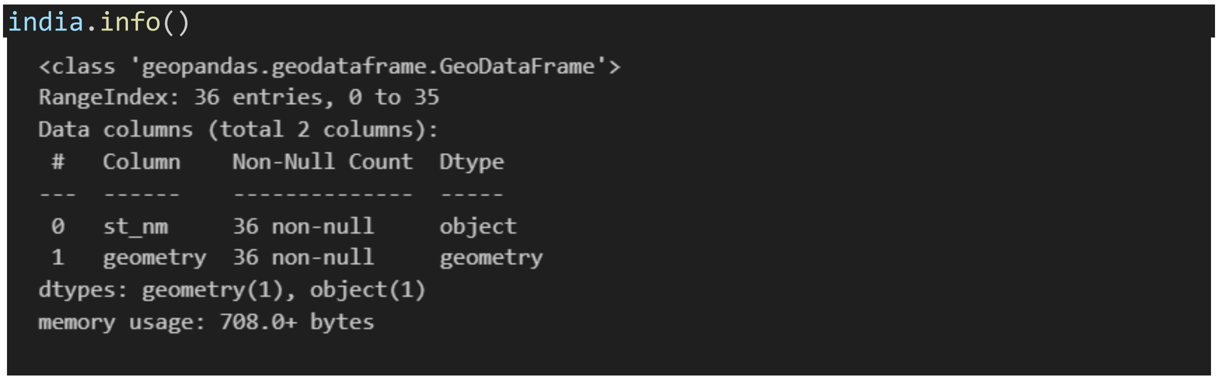
Lets Check the Data Which is Present



```
india.head()
```

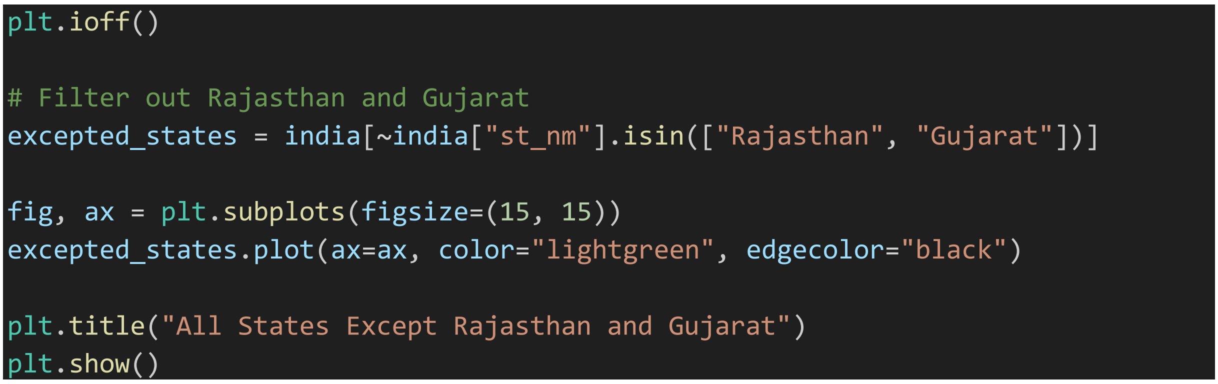
...

	st_nm	geometry
0	Andaman & Nicobar Island	MULTIPOLYGON (((93.71976 7.20707, 93.71909 7.2...
1	Arunanchal Pradesh	POLYGON ((96.16261 29.38078, 96.1686 29.37432,...
2	Assam	MULTIPOLYGON (((89.74323 26.30362, 89.7429 26....
3	Bihar	MULTIPOLYGON (((84.5072 24.26323, 84.50355 24....
4	Chandigarh	POLYGON ((76.84147 30.75996, 76.83599 30.73623...



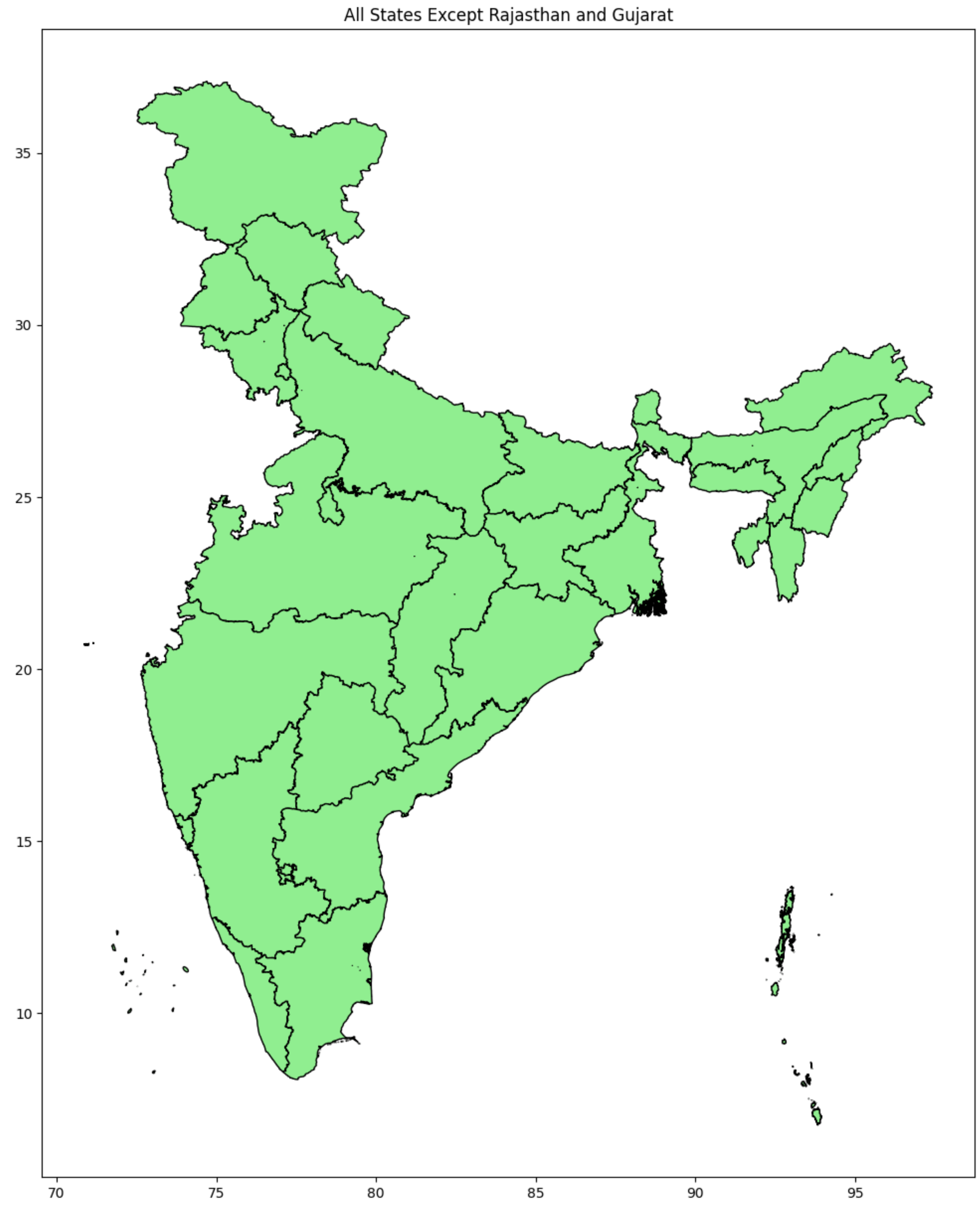
```
india.info()
```

```
<class 'geopandas.geodataframe.GeoDataFrame'>
RangeIndex: 36 entries, 0 to 35
Data columns (total 2 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   st_nm     36 non-null     object
 1   geometry  36 non-null     geometry
dtypes: geometry(1), object(1)
memory usage: 708.0+ bytes
```



```
plt.ioff()

# Filter out Rajasthan and Gujarat
excepted_states = india[~india["st_nm"].isin(["Rajasthan", "Gujarat"])]

fig, ax = plt.subplots(figsize=(15, 15))
excepted_states.plot(ax=ax, color="lightgreen", edgecolor="black")

plt.title("All States Except Rajasthan and Gujarat")
plt.show()
```

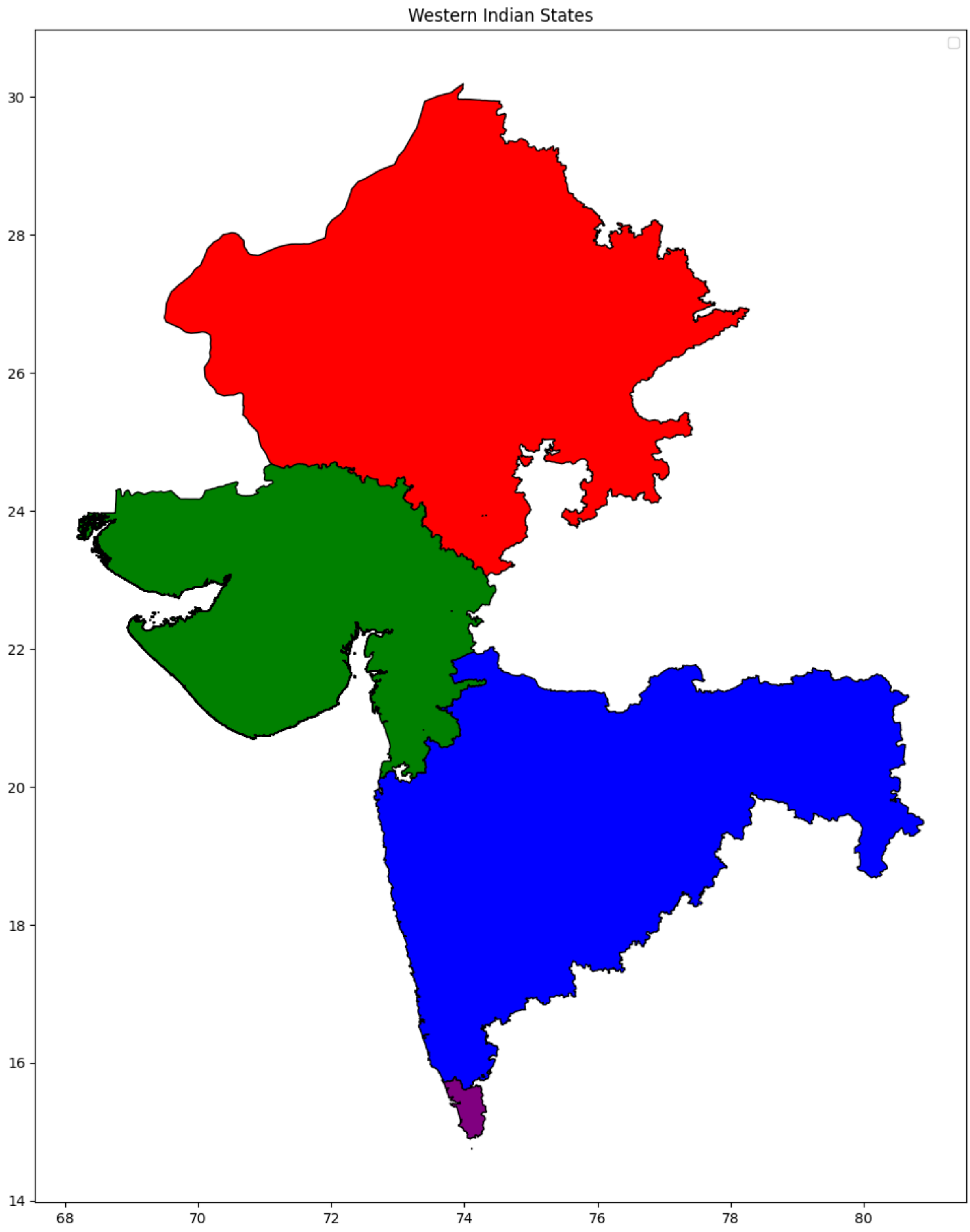
Output


All States Except Rajasthan and Gujarat
35
30
25
20
15
10
70
75
80
85
90
95

Sol 5 : --Lets Draw only Western inidan states in a map

```
plt.ioff()

# Select Western Indian states
western_states = india[india["st_nm"].isin(["Rajasthan", "Gujarat",
"Maharashtra", "Goa"])]

# Define colors for each state
colors = {
    "Rajasthan": "red",
    "Gujarat": "green",
    "Maharashtra": "blue",
    "Goa": "purple"
}

fig, ax = plt.subplots(figsize=(15, 15))
for state_name, color in colors.items():
    state = western_states[western_states["st_nm"] == state_name]
    state.plot(ax=ax, color=color, edgecolor="black", label=state_name)

plt.title("Western Indian States")
plt.legend()
plt.show()
```

Output


Western Indian States
30
28
26
24
22
20
18
16
14
68
70
72
74
76
78
80