**Ex.No. 1 Download, install and explore the features of NumPy, SciPy, Jupyter, Statsmodels and Pandas packages.**

# How to Install Anaconda & Run Jupyter Notebook

Instructions To Install Anaconda and Run Jupyter Notebook

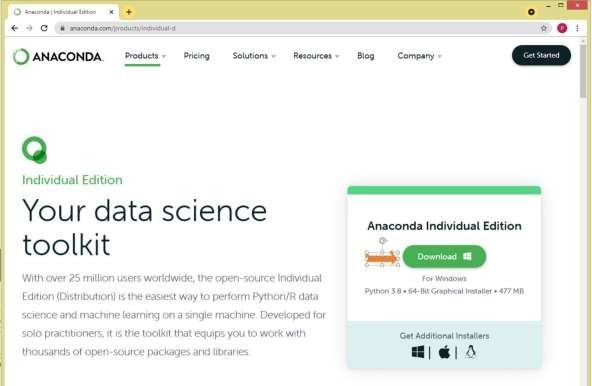
* [Download & Install Anaconda Distribution](https://sparkbyexamples.com/python/install-anaconda-jupyter-notebook/#Download-and-Install-Anaconda-Distribution)
* [Create Anaconda Environment](https://sparkbyexamples.com/python/install-anaconda-jupyter-notebook/#Create-Anaconda-Environment-from-Navigator)
* [Install and Run Jupyter Notebook](https://sparkbyexamples.com/python/install-anaconda-jupyter-notebook/#Install-and-Run-Jupyter-Notebook)

# Download & Install Anaconda Distribution

Follow the below step-by-step instructions to install Anacondadistribution.

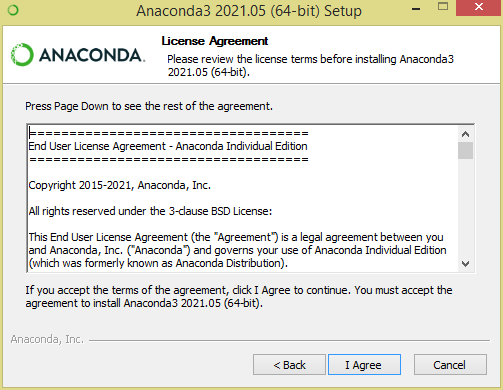
# Download Anaconda Distribution

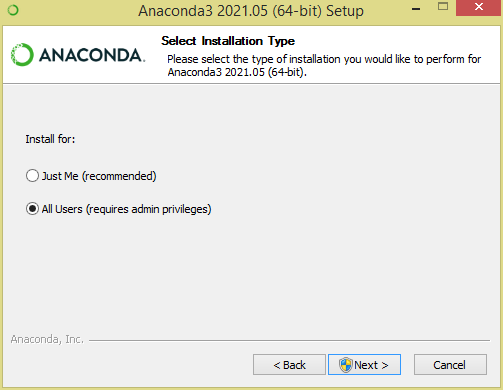
Go to [https://anaconda.com/](https://www.anaconda.com/products/individual-d) and select **Anaconda Individual Edition** todownload the latest version of Anaconda. This downloads the .exe file to the windows download folder.

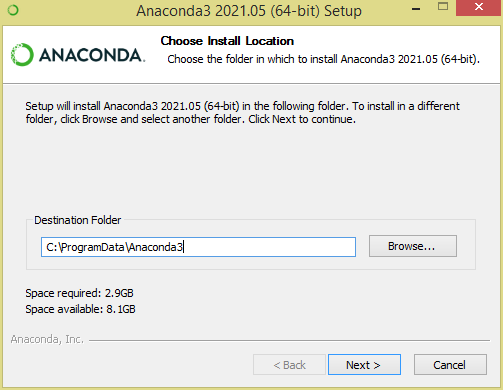


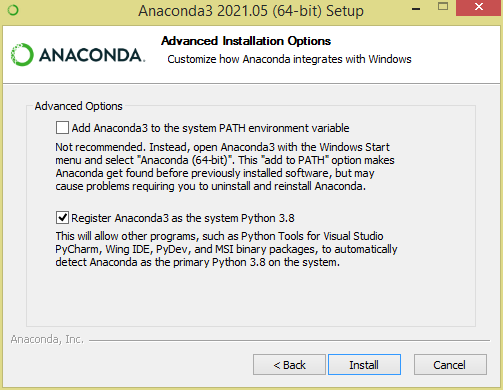
# Install Anaconda

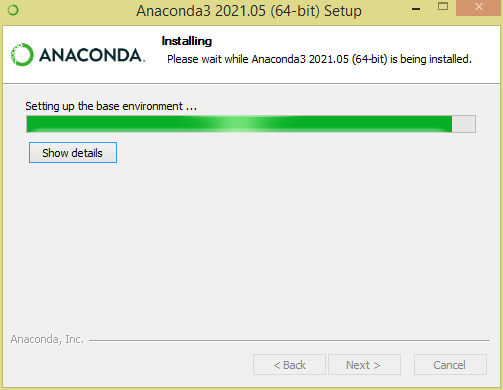
By double-clicking the .exe file starts the Anaconda installation. Follow the below screen shot’s and complete the installation

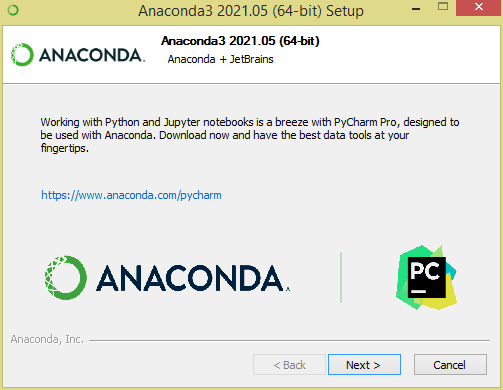


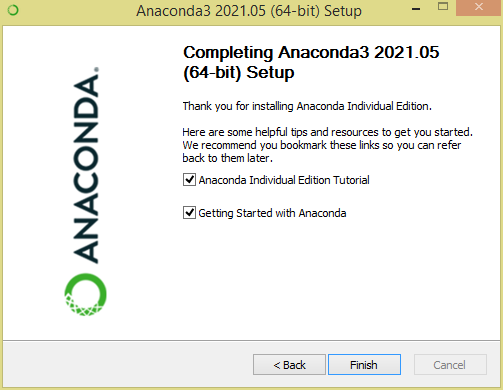












to

This finishes the installation of Anaconda distribution, now let’s see how

create an environment and install Jupyter Notebook.

# Create Anaconda Environment from Navigator

A conda environment is **a directory that contains a specific collection of conda packages that you have installed**. For example, you may have one environment with NumPy 1.7 and its dependencies, and another environment with NumPy 1.6 for legacy testing.

https://conda.io/docs/using/envs.html

# Open Anaconda Navigator

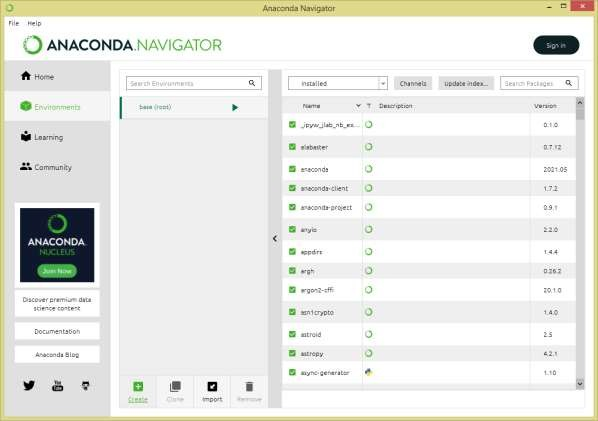
Open Anaconda Navigator from windows start or by searching it. Anaconda Navigator is a UI application where you can control theAnaconda packages, environment e.t.c



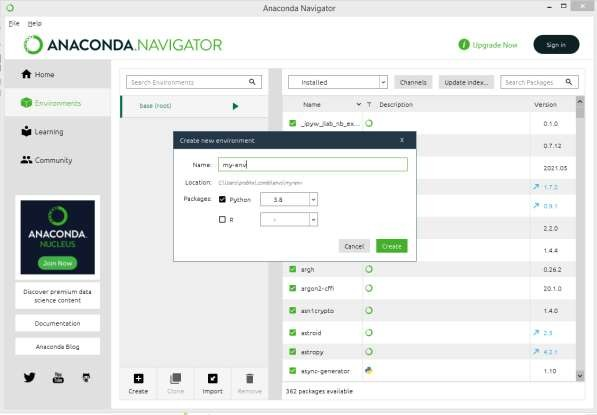
# Create an Environment to Run Jupyter Notebook

This is optional but recommended to create an environment before you proceed. This gives complete segregation of different package installs fordifferent projects you would be working on. If you already have an

environment, you can use it too.

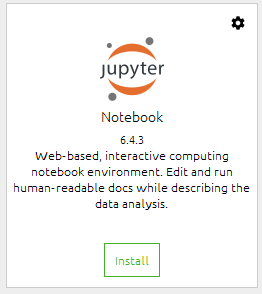


select + Create icon at the bottom of the screen to create an Anaconda environment.



# Install and Run Jupyter Notebook

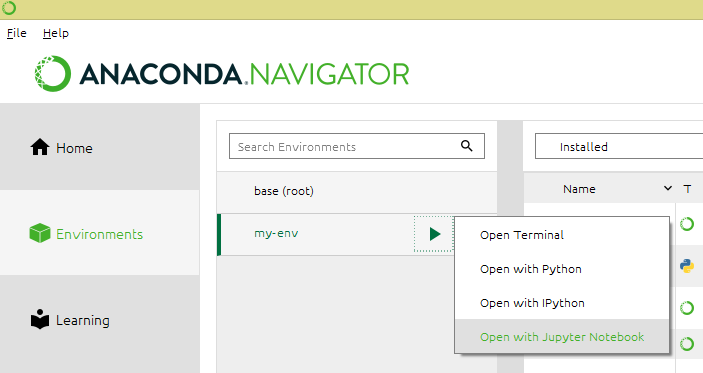
Once you create the anaconda environment, go back to the Home page on Anaconda Navigator and install Jupyter Notebook from an application on the right panel.



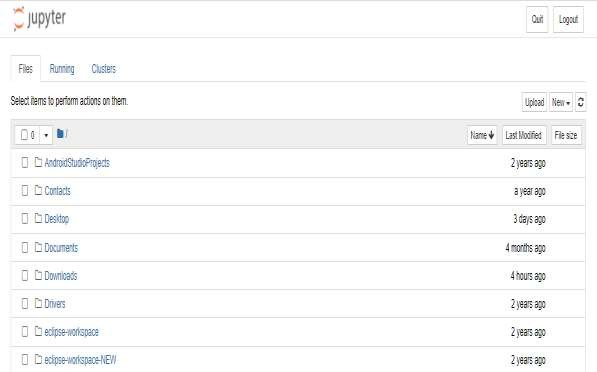
| It will take a few seconds to install Jupyter to your environment, once the | | install |
| --- | --- | --- |
| completes, you can open Jupyter from the same screen or by | accessing **Anaconda** | |

# Navigator -> Environments -> your

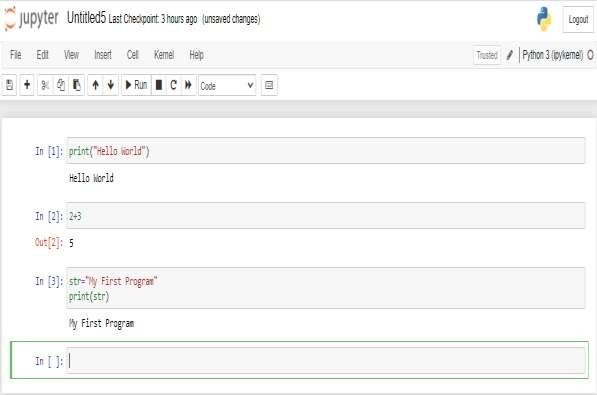
**environment** (mine pandas-tutorial) -> select **Open With Jupyter Notebook**.



This opens up Jupyter Notebook in the default browser.



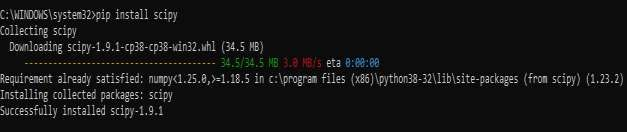
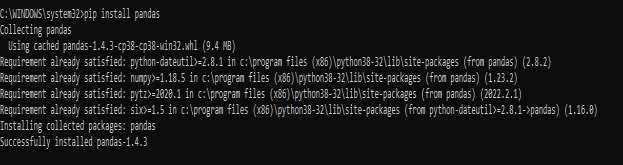
Now select **New** -> **PythonX** and enter the below lines and select **Run**. On Jupyter, each cell is a statement, so you can run each cell independently when there are no dependencies on previous cells.



This completes installing Anaconda and running Jupyter Notebook.







# RESULT:

Thus Jupyter Notebook environment has been successfully installed with all the necessary packages using Anaconda distribution.

# Ex. No 2 Working with Numpy arrays

**Aim**

To implement array object using Numpy module in Python programming

# Algorithm

**Step 1:** Start the program

**Step 2:** Import the required packages

**Step 3:** Read the elements through list/tuple/dictionary

**Step 4:** Convert List/tuple/dictionary into array using built-in methodsStep Step 5: Check the number of dimensions in an array

**Step 6:** Compute the shape of an array or if it’s required reshape an array

**Step 7:** Do the required operations like slicing, iterating, searching, concatenatingand splitting an array element.

**Step 8:** Stop the program

# Create a NumPy ND array Object Program

import numpy as np

arr = np.array([1, 2, 3, 4, 5]) print(arr) print(type(arr))

# Output

[1 2 3 4 5]

<class 'numpy.ndarray'>

# Dimensions in Arrays 0-D Arrays Program

import numpy as np arr = np.array(42) print(arr)

# Output 42

**1-D Arrays Program**

import numpy as np

arr = np.array([1, 2, 3, 4, 5]) print(arr)

# Output [1 2 3 4 5]

**2-D Arrays Program**

import numpy as np

arr = np.array([[1, 2, 3], [4, 5, 6]]) print(arr)

# Output [[1 2 3]

**[4 5 6]]**

# 3-D arrays Program

import numpy as np

arr = np.array([[[1, 2, 3], [4, 5, 6]], [[1, 2, 3], [4, 5, 6]]])

print(arr)

# Output [[[1 2 3]

**[4 5 6]]**

# [[1 2 3]

**[4 5 6]]]**

# Check Number of Dimensions?

**Program**

import numpy as np a = np.array(42) b = np.array([1, 2, 3, 4, 5])

c = np.array([[1, 2, 3], [4, 5, 6]])

d = np.array([[[1, 2, 3], [4, 5, 6]], [[1, 2, 3], [4, 5, 6]]])

rint(a.ndim) print(b.ndim) print(c.ndim) print(d.ndim)

# Output 0

**1**

# 2

**3**

# Access Array Elements Program

import numpy as np

arr = np.array([1, 2, 3, 4]) print(arr[0])

# Output

1

# Program

import numpy as np

arr = np.array([1, 2, 3, 4]) print(arr[2] + arr[3])

# Output

7

# Slicing arrays Program

import numpy as np

arr = np.array([1, 2, 3, 4, 5, 6, 7]) print(arr[1:5])

# Output

[2 3 4 5]

# NumPy Array Shape Program

import numpy as np

arr = np.array([[1, 2, 3, 4], [5, 6, 7, 8]])

print(arr.shape)

# Output (2, 4)

1. **Reshaping arrays Program**

import numpy as np

arr = np.array([1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12])

newarr = arr.reshape(4, 3) print(newarr)

# Output

[[ 1 2 3]

[ 4 5 6]

[ 7 8 9]

[10 11 12]]

# Iterating Arrays Program

import numpy as np arr = np.array([1, 2, 3]) for x in arr: print(x)

# Output

1

2

3

1. **Joining NumPy Arrays Program** import numpy as np arr1 = np.array([1, 2, 3]) arr2 = np.array([4, 5, 6])

arr = np.concatenate((arr1, arr2)) print(arr)

# Output

[1 2 3 4 5 6]

,

# Splitting NumPy Arrays Program

import numpy as np

arr = np.array([1, 2, 3, 4, 5, 6]) newarr = np.array\_split(arr, 3) print(newarr)

# Output

[array([1, 2]), array([3, 4]), array([5, 6])]

# Searching Arrays Program

import numpy as np

arr = np.array([1, 2, 3, 4, 5, 4, 4]) x = np.where(arr == 4) print(x)

# Output (array([3, 5, 6]),)

1. **Sorting Arrays Program**

import numpy as np

arr = np.array([3, 2, 0, 1]) print(np.sort(arr))

# Output

[0 1 2 3]

# RESULT:

Thus Array object has been explored using Numpy module in Python programming successfully.

# Exp. No. 3. Working with Pandas data frames

**Aim:**

To work with DataFrame object using Pandas module in Python Programming

# Algorithm:

**Step 1:** Start the program

**Step 2:** Import the required packages

**Step 3:** Create a DataFrame using built in method.

**Step 4:** Load data into a DataFrame object otherwise Load Files(excel/csv) into a DataFrame

**Step 5:** Display the rows and describe the data set using built in method.

**Step 6:** Display the last 5 rows of the DataFrame. **Step 7:** Check the number of maximum returned rows **Step 8:** Stop the program

# Create a simple Pandas DataFrame:

**Program**

import pandas as pd data = {

"calories": [420, 380, 390],

"duration": [50, 40, 45]

}

#load data into a DataFrame object: df = pd.DataFrame(data) print(df)

# Output

calories duration 0 420 50

1 380 40

2 390 45

1. **Locate Row Program** print(df.loc[

0])

# Output

calories 420

duration 50

Name: 0, dtype: int64

Note: This example returns a Pandas Series.

# (iv )use a list of indexes:

**Program**

print(df.loc[[0, 1]])

# Output

calories duration 0 420 50

1 380 40

Note: When using [], the result is a Pandas DataFrame.

# Named Indexes Program

import pandas as pd data = {

"calories": [420, 380, 390],

"duration": [50, 40, 45]

}

df = pd.DataFrame(data, index = ["day1", "day2", "day3"]) print(df)

# Output

calories duration day1 420 50

day2 380 40

day3 390 45

# Locate Named Indexes

print(df.loc["day2"])

# Output

calories 380

duration 40

Name: 0, dtype: int64

1. **Load Files Into a DataFrame Program** import pandas as pd

df = pd.read\_csv('data.csv') print(df)

# Output

Duration Pulse Maxpulse Calories 0 60 110 130 409.1

1 60 117 145 479.0

2 60 103 135 340.0

3 45 109 175 282.4

4 45 117 148 406.0

.. ... ... ... ...

164 60 105 140 290.8

165 60 110 145 300.4

166 60 115 145 310.2

167 75 120 150 320.4

168 75 125 150 330.4

[169 rows x 4 columns]

# Check the number of maximum returned rows: Program

import pandas as pd print(pd.options.display.max\_ro ws)

In my system the number is 60, which means that if the DataFrame contains more than 60 rows, the print(df) statement will return only the headers and the first and last 5 rows.

import pandas as pd pd.options.display.max\_rows = 9999 df = pd.read\_csv('data.csv') print(df)

# Viewing

**the Data Program** import pandas as pd df = pd.read\_csv('data.csv') print(df.head(4))

# Output

Duration Pulse Maxpulse Calories 0 60 110 130 409.1

1 60 117 145 479.0

2 60 103 135 340.0

3 45 109 175 282.4

4 45 117 148 406.0

# Print the last 5 rows of the DataFrame:

print(df.tail())

print(df.info())

# Output

<class 'pandas.core.frame.DataFrame'> RangeIndex: 169 entries, 0 to 168

Data columns (total 4 columns): # Column Non-Null Count Dtype 0 Duration 169 non-null int64

1. Maxpulse 169 non-null int64
2. Calories 164 non-null float64 dtypes: float64(1), int64(3) memory usage: 5.4 KB

None

# RESULT:

Thus Data Frame object using Pandas module in Python Programming has been successfully explored

# Exp. No. 4. Reading data from text files, Excel and the web and exploring various commands for doing descriptive analytics on the Iris data set.

**Aim:**

To perform descriptive analytics on Iris dataset using Python programming

# Algorithm

**Step 1:** Start the program

**Step 2:** Import the required packages

**Step 3:** Load Files(excel/csv/ text) into a DataFrame from Iris data set **Step 4:** Display the rows and describe the data set using built in methods **Step 5:** Compare Petal Length and Petal Width

**Step 6:** Visualize the data set using histogram with distplot, heatmapsbox plots methods

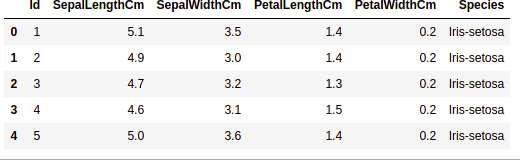
**Step 7:** Check Missing Values, Duplicates and remove outliers

**Step 8:** Stop the program

# Program

import pandas as pd # Reading the CSV file df = pd.read\_csv("Iris.csv") # Printing top 5 rows **df.head()**

# Output:



**Getting Information about the Dataset**

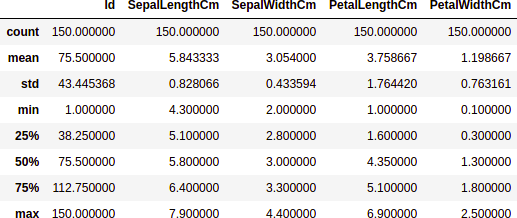
df.shape **Output:** (150, 6)

**df.info()**

**Output**

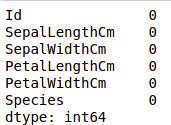


df.describe()



# Checking Missing Values

df.isnull().sum()

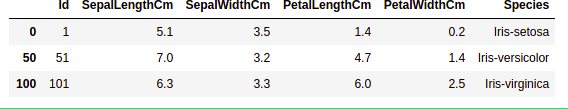


# Checking Duplicates

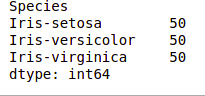
data = df.drop\_duplicates(subset

**="Species",) data**

**Output**



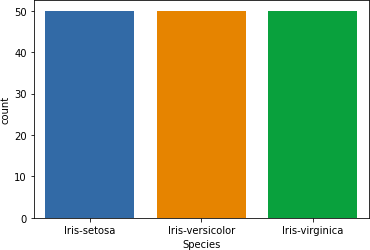
df.value\_counts("Species")



# Data Visualization

# importing packages import seaborn as sns

import matplotlib.pyplot as plt sns.countplot(x='Species', data=df, ) plt.show()

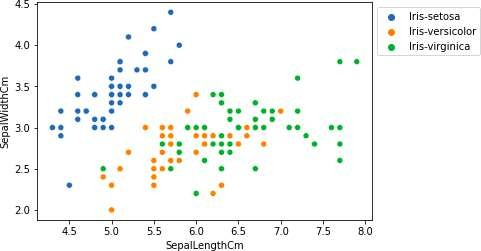


# Comparing Sepal Length and Sepal Width

# importing packages import seaborn as sns

import matplotlib.pyplot as plt sns.scatterplot(x='SepalLengthCm', y='SepalWidthCm',

hue='Species',

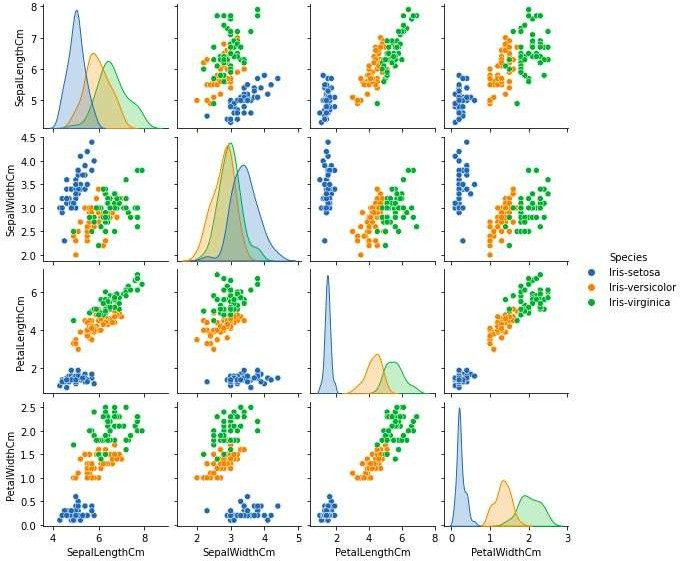
data=df, ) # Placing Legend outside the Figure plt.legend(bbox\_to\_anchor=(1, 1), loc=2) plt.show()

# importing packages

**import** seaborn as sns

**import** matplotlib.pyplot as plt sns.pairplot(df.drop(['Id'], axis **=**1), hue**=**'Species', height**=**2)

# Output:



**Histograms Program**

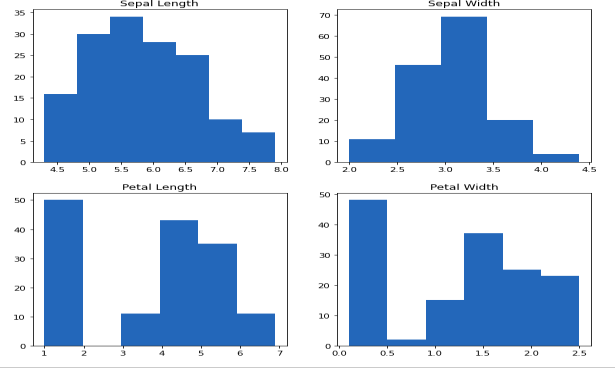
# importing packages

import seaborn as sns

import matplotlib.pyplot as plt

fig, axes = plt.subplots(2, 2, figsize=(10,10)) axes[0,0].set\_title("Sepal Length") axes[0,0].hist(df['SepalLengthCm'], bins=7) axes[0,1].set\_title("Sepal Width") axes[0,1].hist(df['SepalWidthCm'], bins=5) axes[1,0].set\_title("Petal Length") axes[1,0].hist(df['PetalLengthCm'], bins=6) axes[1,1].set\_title("Petal Width") axes[1,1].hist(df['PetalWidthCm'], bins=6)

# Output:



**Histograms with Distplot Plot Program**

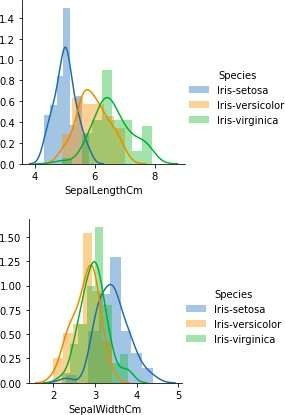
# importing packages

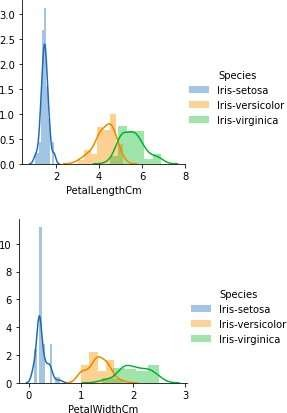
import seaborn as sns

import matplotlib.pyplot as plt

plot = sns.FacetGrid(df, hue="Species") plot.map(sns.distplot, "SepalLengthCm").add\_legend() plot = sns.FacetGrid(df, hue="Species") plot.map(sns.distplot, "SepalWidthCm").add\_legend() plot = sns.FacetGrid(df, hue="Species") plot.map(sns.distplot, "PetalLengthCm").add\_legend() plot = sns.FacetGrid(df, hue="Species") plot.map(sns.distplot, "PetalWidthCm").add\_legend() plt.show()

# Output:

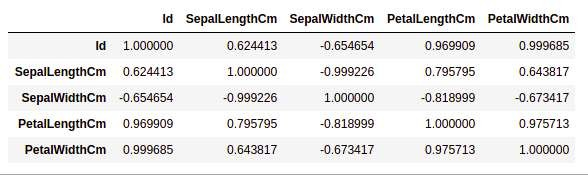




**Handling Correlation**

data.corr(method**=**'pearson')

# Output:



**Heatmaps Program**

# importing packages

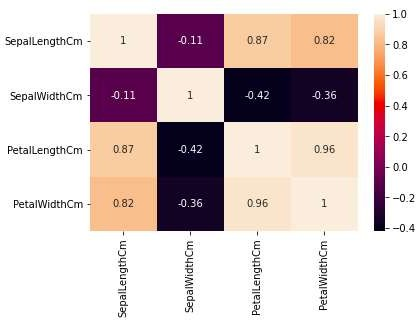
import seaborn as sns

import matplotlib.pyplot as plt sns.heatmap(df.corr(method='pearson').drop( ['Id'], axis=1).drop(['Id'], axis=0),

annot =

True); plt.show()

# Output:



**Box Plots**

**Program**

# importing packages

**import** seaborn as sns

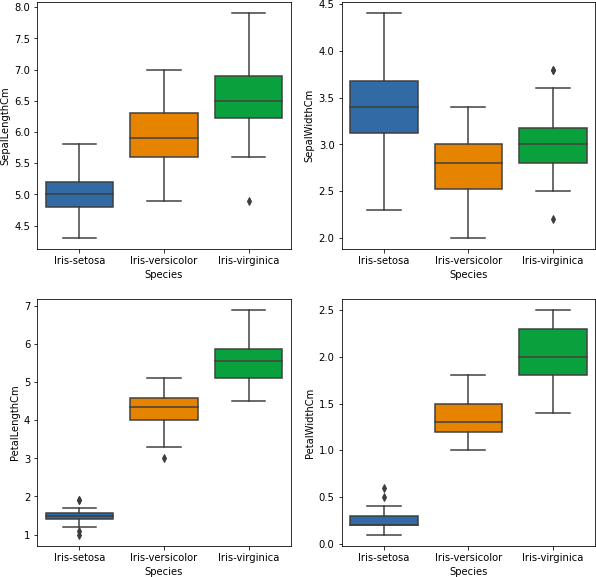
**import** matplotlib.pyplot as plt

**def** graph(y): sns.boxplot(x**=**"Species", y**=**y, data**=**df) plt.figure(figsize**=**(10,10))

# Adding the subplot at the specified # grid position plt.subplot(221) graph('SepalLengthCm') plt.subplot(222) graph('SepalWidthCm')

plt.subplot(223) graph('PetalLengthCm') plt.subplot(224) graph('PetalWidthCm') **plt.show()**

# Output:

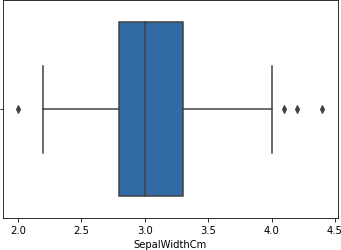


**Program**

# importing packages import seaborn as sns import matplotlib.pyplot as plt # Load the dataset

df = pd.read\_csv('Iris.csv') sns.boxplot(x='SepalWidthCm', data=df)

# Output:



**Removing Outliers Program**

# Importing

import sklearn

from sklearn.datasets import load\_boston

import pandas as pd import seaborn as sns # Load the dataset

df = pd.read\_csv('Iris.csv') # IQR

Q1 = np.percentile(df['SepalWidthCm'], 25, interpolation = 'midpoint')

Q3 = np.percentile(df['SepalWidthCm'], 75, interpolation = 'midpoint')

IQR = Q3 - Q1

print("Old Shape: ",

df.shape) # Upper bound

upper = np.where(df['SepalWidthCm'] >= (Q3+1.5\*IQR)) # Lower bound

lower = np.where(df['SepalWidthCm'] <= (Q1-1.5\*IQR))

# Removing the Outliers df.drop(upper[0], inplace = True) df.drop(lower[0], inplace = True) print("New Shape: ", df.shape) sns.boxplot(x='SepalWidthCm', data=df)

# Output:

**RESULT:**

Thus Iris dataset has been explored and descriptively analysed using Python programming

# Exp. No. 5. Use the diabetes data set from UCI and Pima Indians Diabetes data set for performing the following:

**Aim:**

To perform various exploratory data analysis on Pima Indians Diabetes dataset using Python Programming

# Univariate analysis: Frequency, Mean, Median, Mode, Variance,Standard Deviation, Skewness and Kurtosis.

1. **Bivariate analysis: Linear and logistic regression modeling**

# Multiple Regression analysis

1. **Also compare the results of the above analysis for the two data sets.**

# Algorithm

**Step 1:** Start the program

**Step 2:** Import the required packages

**Step 3:** Load Files (excel/csv/ text) into a Data Frame from UCI and Pima IndiansDiabetes data set

**Step 4**: Display the rows and describe the data set using built in methods

**Step 5:** Compute Frequency, Mean, Median, Mode, Variance, Standard Deviation,Skewness and Kurtosis

**Step 6:** Visualize the data set using histogram with distplot, heatmapsbox plots methods **Step 7:** Check Missing Values, Duplicates and remove outliers using built in methodStep 8: Stop the program

# Program

import pandas as pd import seaborn as sns

import matplotlib.pyplot as plt

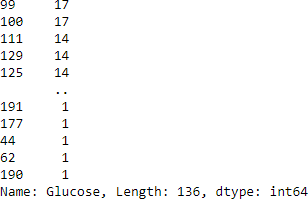
%matplotlib inline

from sklearn.linear\_model import LogisticRegression from sklearn.externals import joblib

df = pd.read\_csv('C:/Users/praveen/Downloads/diabetes.csv')

51

count = df['Glucose'].value\_counts()



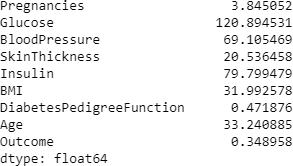
display(count) df.head()



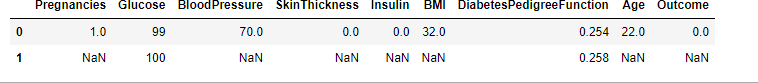
df.describe()



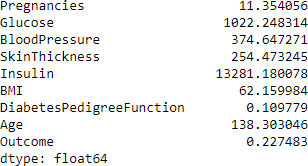
df.mean()



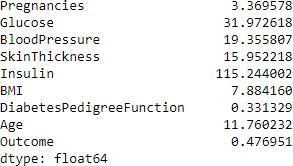
df.mode()



df.var()



df.std()



df.skew() Pregnancies 0.9016

74 Glucose

0.1737

54 BloodPressure -

1.843608 SkinThickness

0.1093

72 Insulin

2.2722

51 BMI -

0.428982

DiabetesPedigreeFunction 1.919911 Age

1.129597

Outcome 0.

635017 dtype: float64

df.kurtosis() Pregnancies 0.159220

Glucose 0.640780

BloodPressure 5.180157

SkinThickness -0.520072

Insulin 7.214260

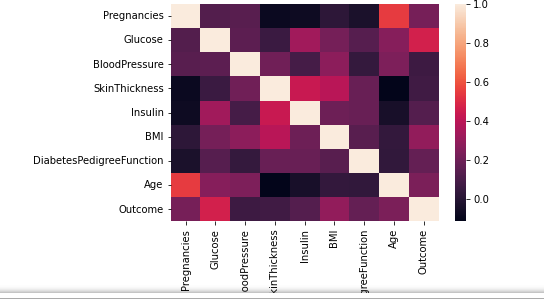
BMI 3.290443

DiabetesPedigreeFunction 5.594954

Age 0.643159

Outcome –1.600930 dtype: float64

corr = df.corr() sns.heatmap(corr, xticklabels=corr.columns, yticklabels=corr.columns)



sns.countplot('Outcome', data=df) plt.show()



# Computing the %age of diabetic and non-diabetic in the sample Out0=len([df.Outcome==1])

Out1=len([df.Outcome==0]) Total=Out0+Out1

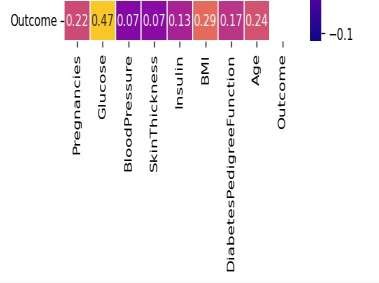
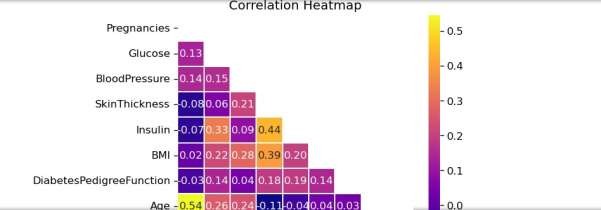
PC\_of\_1 = Out1\*100/Total PC\_of\_0 =

Out0\*100/Total PC\_of\_1, PC\_of\_0 (50.0, 50.0)

plt.figure(dpi = 120,figsize= (5,4))

mask = np.triu(np.ones\_like(df.corr(),dtype = bool)) sns.heatmap(df.corr(),mask = mask, fmt = ".2f",annot=True,lw=1,cmap = 'plasma') plt.yticks(rotation = 0)

plt.xticks(rotation = 90) plt.title('Correlation Heatmap') plt.show()



# RESULT:

Thus various exploratory data analysis has been performed on Pima Indians Diabetes dataset using Python Programming successfully.

# Exp. No. 6 Apply and explore various plotting functions on UCI data sets.

**Aim:**

To apply various plotting functions on UCI data set using Python Programming

# Normal curves

1. **Density and contour plots**

# Correlation and scatter plots

1. **Histograms**

# Three dimensional plotting

**Algorithm**

**Step 1:** Start the program

**Step 2:** Import the required packages

**Step 3:** Load Files (excel/csv/ text) into a Data Frame from UCI data set

**Step 4:** Describe the data set using built in method

**Step 5:** Compute Frequency, Mean, Median, Mode, Variance, Standard Deviation,

**Step 6:** Visualize the data set using Explore various plotting functions on UCI datasets for the following

1. Normal curves
2. Density and contour plots
3. Correlation and scatter plots
4. Histograms
5. Three-dimensional plotting

**Step 7:** Analyze the sample data and do the required operations

**Step 8:** Stop the program

# Normal curves Program

import pandas as pd

import matplotlib.pyplot as plt import numpy as np

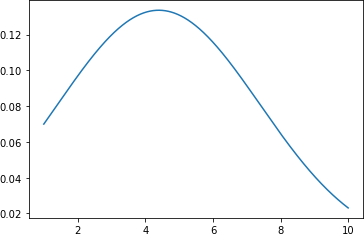
df=pd.read\_csv("C:/Users/praveen/Downloads/dataset\_diabetes/diabetic\_dat a.cs v")

df.head() mean

=df['time\_in\_hospital'].mean() std =df['time\_in\_hospital'].std() x\_axis = np.arange(1, 10, 0.01)

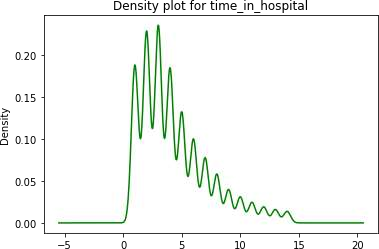
plt.plot(x\_axis, norm.pdf(x\_axis, mean, std)) plt.show()

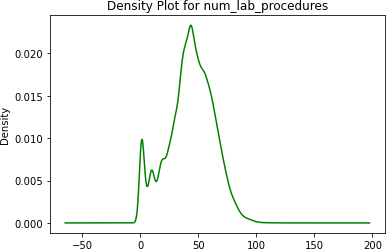
# Output



1. **Density and contour plotsProgram** df.time\_in\_hospital.plot.density(color='gre en') plt.title('Density plot for

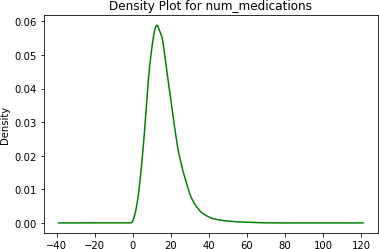
time\_in\_hospital') plt.show()



df.num\_lab\_procedures.plot.density(color='gre en') plt.title('Density Plot for num\_lab\_procedures') plt.show()

df.num\_medications.plot.density(color='gree n') plt.title('Density Plot for num\_medications') plt.show()

# Output



**Program** # for 'tip' attribute # using plot.kde()

df.number\_emergency.plot.kde(color='green') plt.title('KDE-Density plot for number\_emergency') plt.show()

# Output

**Program**

def func(x, y):

return np.sin(x) \*\* 2 +

np.cos(y) \*\*2 # generate 50 values b/w 0 a5 mean

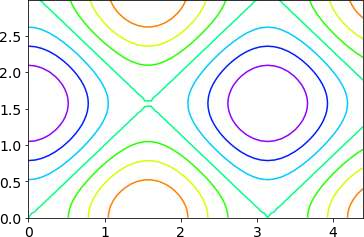
=df['time\_in\_hospital'].mean() std =df['time\_in\_hospital'].std() x = np.linspace(0,mean)

y =np.linspace(0, std)

# Generate combination of grids X, Y = np.meshgrid(x,y) Z = func(X, Y)

# Draw rectangular contour plot plt.contour(X, Y, Z, cmap='gist\_rainbow\_r');

# Output

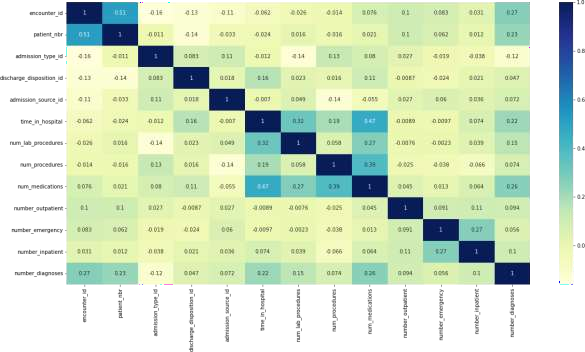


1. **Correlation and scatter plots Program**

mp.figure(figsize=(20,10))

dataplot = sb.heatmap(data.corr(), cmap="YlGnBu", annot=True)

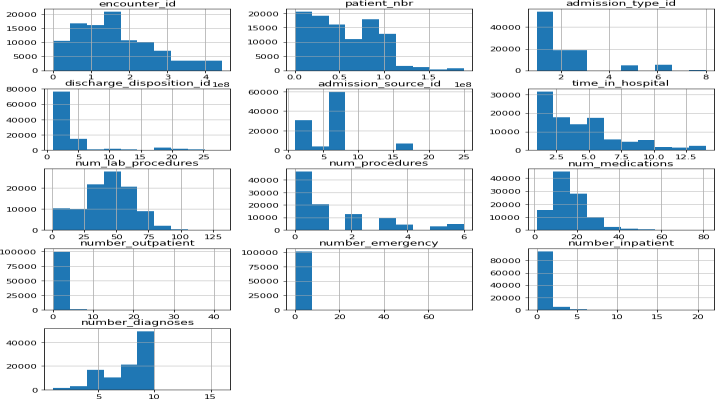
# Output



1. **Histograms Program**

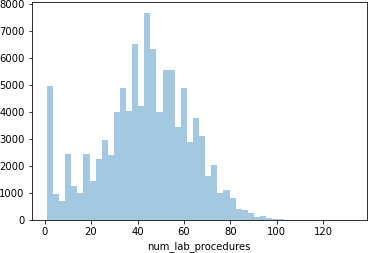
df.hist(figsize=(12,12),layout=(5,3))

# Output



# plotting histogram for carat using distplot() sb.distplot(a=df.num\_lab\_procedures, kde=False) # visualizing plot using matplotlib.pyplot library plt.show()

# Output



1. **Three dimensional plotting Program**

fig = plt.figure()

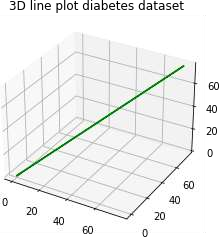
ax = plt.axes(projection ='3d') x =df['number\_emergency'] x= pd.Series(x, name= '')

y =df['number\_inpatient'] y = pd.Series(x, name='')

z =df['number\_outpatient']

z = pd.Series(x, name='') ax.plot3D(x, y, z,'green') ax.set\_title('3D line plot diabetes dataset') plt.show()

# Output



**RESULT:**

Thus apply various plotting functions on UCI data set using Python Programming

# Exp. No. 7. Visualizing Geographic Data with Basemap

**Aim:**

To visualize Geographic Data using BaseMap module in Python Programming

# Algorithm:

**Step 1:** Start the program

**Step 2:** Import the required packages

**Step 3:** Visualize Geographic Data with Basemap

**Step 4:** Display the Base map using built in method like basemap along with latitude and longitude parameters

**Step 5:** Display the Coastal lines meters and Country boundaries using built inmethods

**Step 6:** Fill the Coastal lines meters and Country boundaries with suitable colours

**Step 7:** Create a global map with a Cylindrical Equidistant Projection, Orthographic Projection, Robinson Projection

**Step 8:** Stop the program

**Create a global map with a** Ortho Projection

# Program

%matplotlib inline import numpy as np

import matplotlib.pyplot as plt from mpl\_toolkits.basemap import Basemap plt.figure(figsize=(8, 8))

m = Basemap(projection='ortho', resolution=None, lat\_0=50, lon\_0=-100) m.bluemarble(scale=0.5);

# Output



**Program**

fig = plt.figure(figsize=(8, 8))

m = Basemap(projection='lcc', resolution=None, width=8E6, height=8E6,

lat\_0=45, lon\_0=-100,)

m.etopo(scale=0.5,

alpha=0.5)

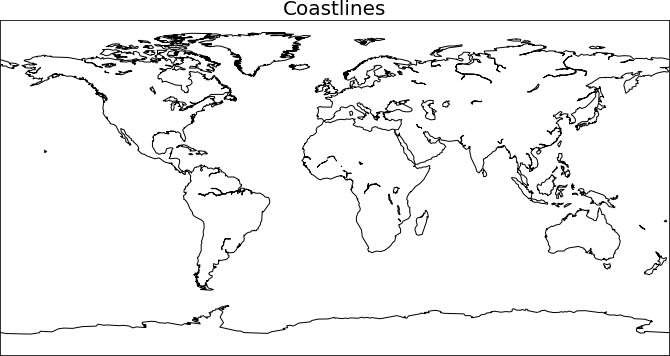
# Map (long, lat) to (x, y) for plotting x, y = m(-122.3, 47.6) plt.plot(x, y, 'ok', markersize=5) plt.text(x, y, ' INDIA', fontsize=12);

# Output

**Create a global map with a Coastlines Program**

fig = plt.figure(figsize = (12,12)) m = Basemap() m.drawcoastlines() plt.title("Coastlines", fontsize=20) plt.show()

# Output



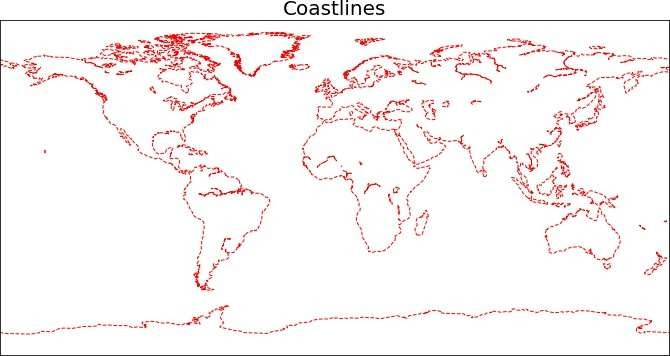
**Program**

fig = plt.figure(figsize = (12,12)) m = Basemap()

m.drawcoastlines(linewidth=1.0, linestyle='dashed', color='red') plt.title("Coastlines", fontsize=20)

plt.show()

# Output



**Create a global map with a Country boundariesProgram**

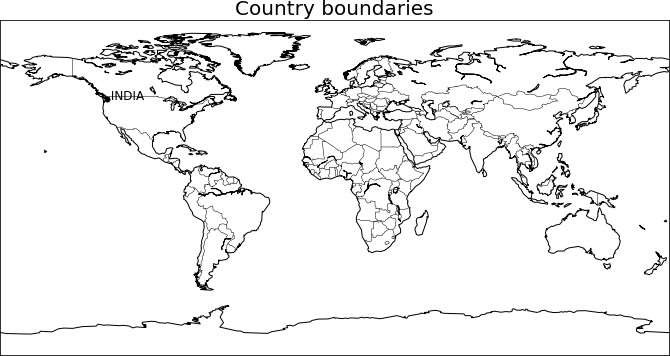
fig = plt.figure(figsize = (12,12)) m = Basemap()

m.drawcoastlines(linewidth=1.0, linestyle='solid', color='black') m.drawcountries()

plt.title("Country boundaries", fontsize=20) x, y = m(-122.3, 47.6) plt.plot(x, y, 'ok',

markersize=5) plt.text(x, y, ' INDIA', fontsize=12); plt.show()

# Output



**Create a global map with a Mercator Projection Program**

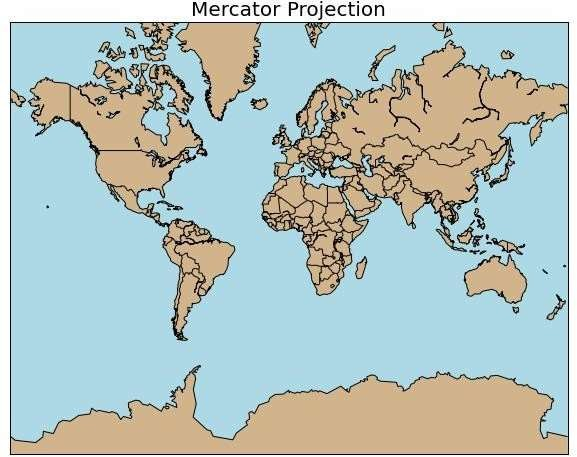
fig = plt.figure(figsize = (10,8))

m = Basemap(projection='merc',llcrnrlat=-80,urcrnrlat=80,llcrnrlon=- 180,urcrnrlon=180)

76

m.drawcoastlines() m.fillcontinents(color='tan',lake\_color='lightblue') m.drawcountries(linewidth=1, linestyle='solid', color='k' ) m.drawmapboundary(fill\_color='lightblue') plt.title("Mercator Projection", fontsize=20)

# Output



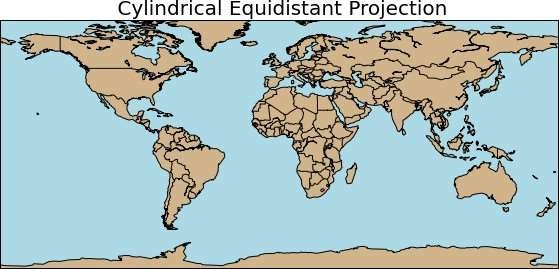
**Create a global map with a Cylindrical Equidistant Projection Program**

fig = plt.figure(figsize = (10,8))

m = Basemap(projection='cyl',llcrnrlat=-80,urcrnrlat=80,llcrnrlon=- 180,urcrnrlon=180) m.drawcoastlines() m.fillcontinents(color='tan',lake\_color='lightblue') m.drawcountries(linewidth=1, linestyle='solid', color='k' ) m.drawmapboundary(fill\_color='lightblue')

plt.title(" Cylindrical Equidistant Projection", fontsize=20)

# Output



**Create a global map with Orthographic Projection Program**

fig = plt.figure(figsize = (10,8))

m = Basemap(projection='ortho', lon\_0 = 25, lat\_0 = 10) m.drawcoastlines() m.fillcontinents(color='tan',lake\_color='lightblue') m.drawcountries(linewidth=1, linestyle='solid', color='k' ) m.drawmapboundary(fill\_color='lightblue') plt.title("Orthographic Projection", fontsize=18)

# Output

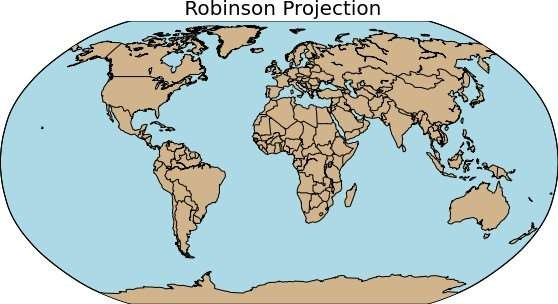
**Create a global map with a Robinson Projection Program**

fig = plt.figure(figsize = (10,8))

m = Basemap(projection='robin',llcrnrlat=-80,urcrnrlat=80,llcrnrlon=- 180,urcrnrlon=180, lon\_0 = 0, lat\_0 = 0)

m.drawcoastlines() m.fillcontinents(color='tan',lake\_color='lightblue') m.drawcountries(linewidth=1, linestyle='solid', color='k' ) m.drawmapboundary(fill\_color='lightblue') plt.title(" Robinson Projection", fontsize=20)

# Output



**RESULT**

Thus Geographic Data has been visualized using Base Map module in Python Programming successfully.

# VIVA VOICE

1. Benefits of Data Preparations
2. What is bias and list its types?
3. Write short notes on Discrete and continuous variable
4. Define standard normal curve with equation and graph.
5. List out the types of non linear relationship.
6. Difference between Linear and multiple regressions.
7. What is .loc( ), .iloc( ), .ix( ) ?
8. Differentiate append() and concat() in pandas.
9. Write Short notes on KDE
10. Define seaborn plots.
11. Identify the steps of data science process
12. What is data cleaning?
13. Differentiate Histogram and bar graph
14. What is Z-Score?
15. Define Causation.
16. Compare Correlation and Regression.
17. Where is NumPy used?
18. “List is mutable”- Justify with example
19. What is density plot?
20. Write the significance of Data Visualization.