Experiment:1.1

Aim: Write a program to implement normal distribution and histograms for data visualization.

Tools/Software Required: Google Colab or any language compiler Python, Jupyter Notebook, Dataset for visulization

Description: The experiment involves writing a program to implement the normal distribution and histograms for data visualization. The aim of this experiment is to gain practical experience in generating and visualizing data that follows a normal distribution, as well as understanding the concept of histograms as a graphical representation of data.

To conduct the experiment, the program should perform the following steps:

1.**Data Generation**: Generate a dataset of random numbers that follow a normal distribution. The program should allow the user to specify the mean and standard deviation for the distribution or provide default values.

2.**Normal Distribution Plot**: Plot the generated dataset on a graph, representing the normal distribution curve. The x-axis of the graph represents the values in the dataset, while the y-axis represents the probability density or frequency of occurrence of those values. The program should use appropriate libraries or functions to generate the curve and plot it accurately.

3.**Histogram Generation**: Create a histogram to visualize the distribution of the dataset. Divide the range of values into equal intervals or bins and count the number of occurrences within each bin. Plot the histogram, where the x-axis represents the intervals or bins, and the y-axis represents the frequency or count of values falling within each bin. The program should ensure that the histogram is properly scaled and labeled for clear visualization.

4.**Program Interactivity**: Implement user-friendly interactions in the program. Allow the user to input the desired parameters, such as the number of data points, mean, and standard deviation, as well as adjust the number of bins for the histogram. Provide options for displaying the generated plots, saving them as image files, or exporting the data for further analysis.

5.**Documentation and Analysis**: Include appropriate documentation within the program code to explain its purpose, functionality, and any necessary instructions for running the program. Additionally, provide an analysis or interpretation of the generated plots, discussing the shape, central tendency, and spread of the data based on the normal distribution and histogram.

Steps:

Step 1: Setting Up the Environment

Ensure that Python and necessary libraries (such as NumPy, Pandas, and Matplotlib) are installed on your computer.

Launch Jupyter Notebook or your preferred Python IDE.

Step 2: Importing Required Libraries

Begin by importing the necessary libraries like pandas for data manipulation and visualization:

Step 3: Loading the Dataset

If you have a dataset to visualize, load it using the appropriate function (pd.read_csv(), pd.read_excel(), etc.). Make sure the dataset is in a compatible format (CSV, Excel, etc.).

If you don't have a dataset, generate a random dataset using NumPy. For example, to generate 1000 data points from a normal distribution with mean 0 and standard deviation 1:

Step 4: Creating a Histogram

1.To create a histogram, use the plt.hist() function, specifying the dataset and the number of bins:

2.Customize the plot by modifying the labels, title, and appearance of the histogram as needed.

Step 5: Implementing Normal Distribution Plot

1.To plot the normal distribution curve, use the plt.plot() function with the appropriate mean and standard deviation values**:**

Customize the plot by modifying labels, title, and appearance.

Step 6: Analyzing Data Distribution

1.Utilize the histogram and normal distribution plot to analyze the distribution of the dataset.

2.Observe the shape of the histogram to identify any skewness or asymmetry in the data.

3.Compare the histogram with the normal distribution plot to assess how closely the data follows a normal distribution.

4.Calculate key statistical properties such as mean, median, mode, standard deviation, skewness, and kurtosis using appropriate functions(np.mean(), np.median(), np.std(), scipy.stats.skew(), scipy.stats.kurtosis(), etc.).

Step 7: Comparing Data Sets

If you have multiple datasets, repeat Steps 3 to 6 for each dataset.

Compare and contrast the histograms and normal distribution plots of different datasets to identify variations in their distributions.

Code:

```
import matplotlib.pyplot as plt
import numpy as np
%matplotlib inline
height = np.arange(150,200,2)
height
```

```
plt.hist(height)
plt.hist(height, rwidth = 0.9)
```

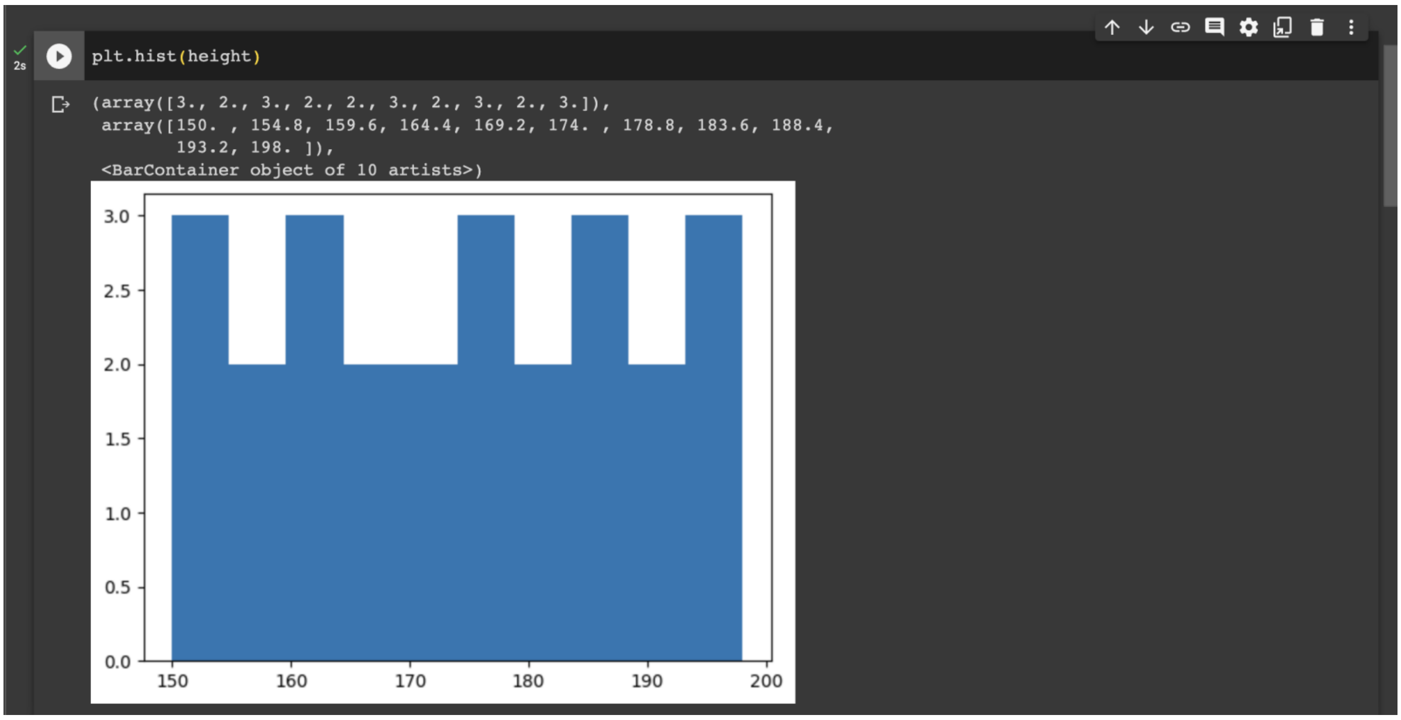
```
bin = [150,165,175,190,200]
plt.hist(height, rwidth = .9, bins=bin)
```

```
from matplotlib import legend
height_m = np.arange(150,200,2)
height_f = np.arange(140,180,2)
plt.hist([height_m, height_f],rwidth = .9,
label=['male','female'])
```

```
plt.legend()
```

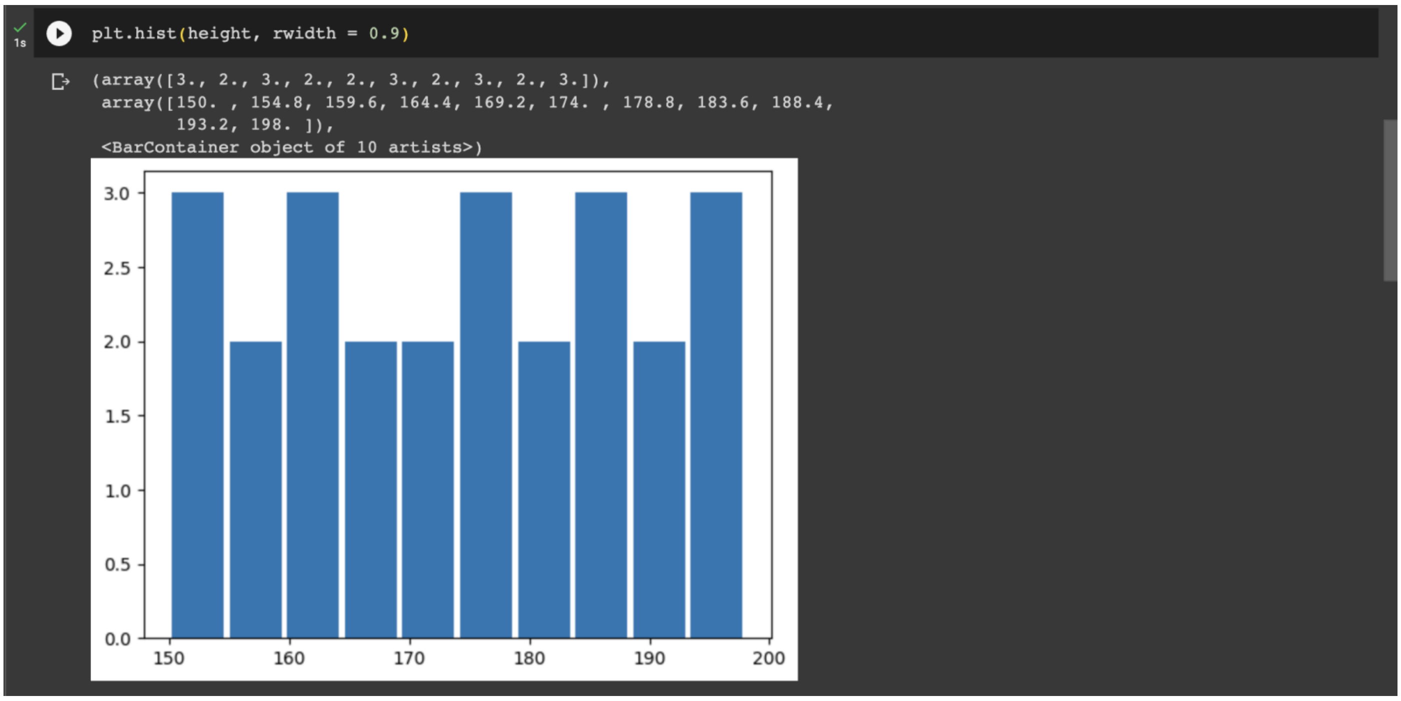
```
rand_n = np.random.randn(1000)
plt.hist(rand_n)
```

Output:



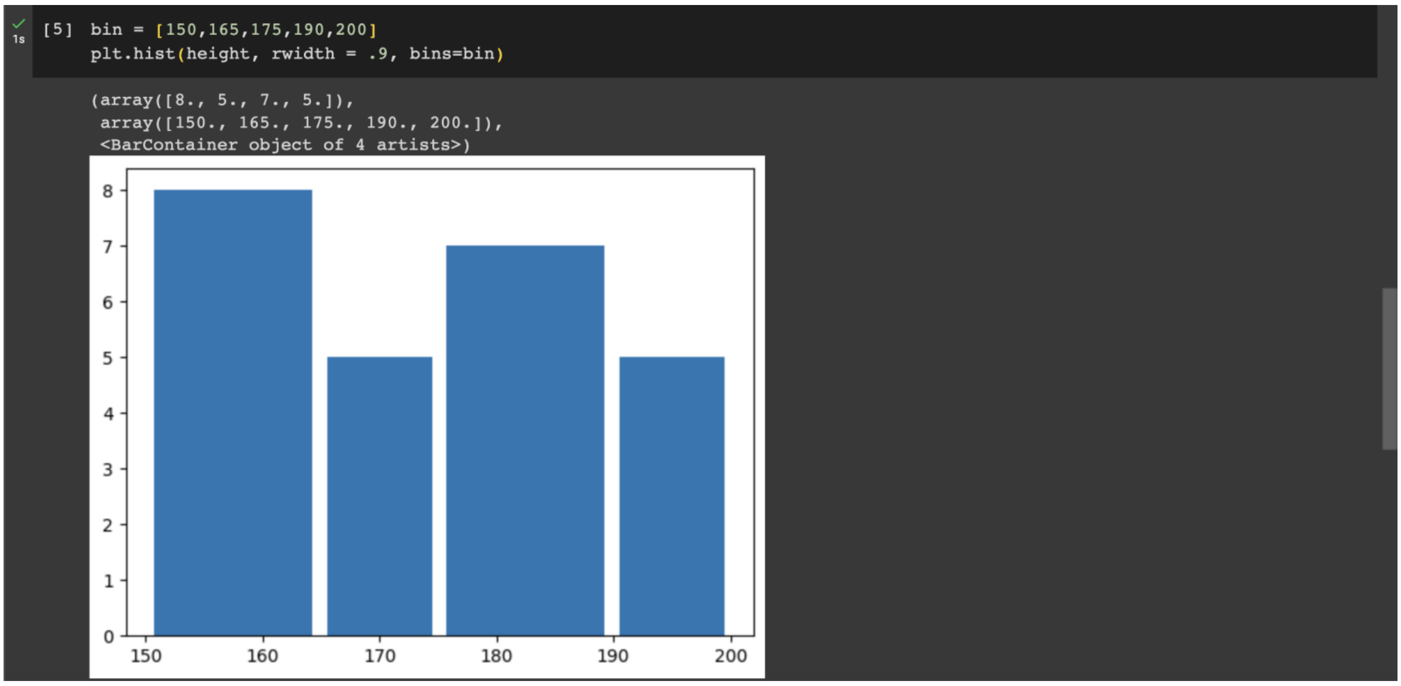
```
plt.hist(height)
```

```
(array([3., 2., 3., 2., 2., 3., 2., 3., 2., 3.]),
 array([150. , 154.8, 159.6, 164.4, 169.2, 174. , 178.8, 183.6, 188.4,
        193.2, 198. ]),
 <BarContainer object of 10 artists>)
```



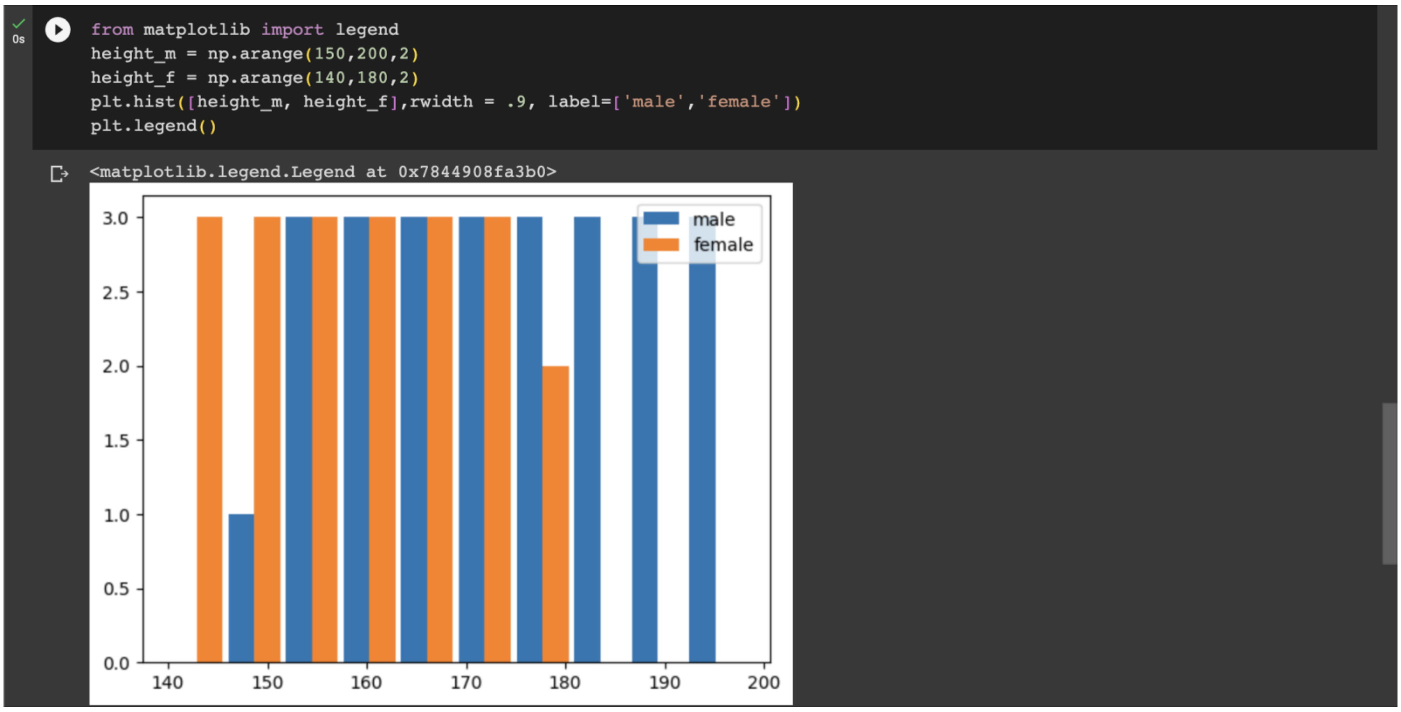
```
plt.hist(height, rwidth = 0.9)

(array([3., 2., 3., 2., 2., 3., 2., 3., 2., 3.]),
 array([150. , 154.8, 159.6, 164.4, 169.2, 174. , 178.8, 183.6, 188.4,
        193.2, 198. ]),
 <BarContainer object of 10 artists>)
```



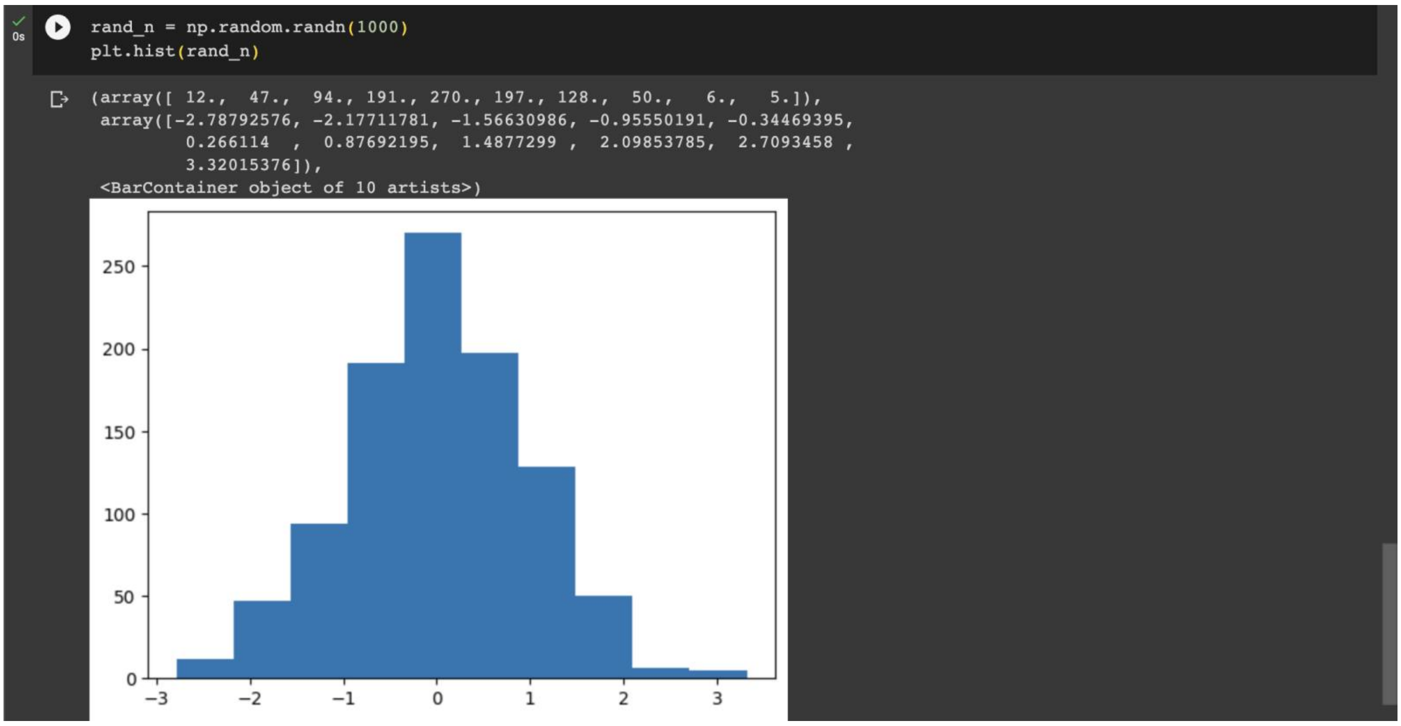
```
[5] bin = [150,165,175,190,200]
    plt.hist(height, rwidth = .9, bins=bin)

(array([8., 5., 7., 5.]),
 array([150., 165., 175., 190., 200.]),
 <BarContainer object of 4 artists>)
```



```
from matplotlib import legend
height_m = np.arange(150,200,2)
height_f = np.arange(140,180,2)
plt.hist([height_m, height_f],rwidth = .9, label=['male','female'])
plt.legend()
```

```
<matplotlib.legend.Legend at 0x7844908fa3b0>
```



```
rand_n = np.random.randn(1000)
plt.hist(rand_n)
```

```
(array([ 12.,  47.,  94., 191., 270., 197., 128.,  50.,   6.,   5.]),
 array([-2.78792576, -2.17711781, -1.56630986, -0.95550191, -0.34469395,
         0.266114  ,  0.87692195,  1.4877299 ,  2.09853785,  2.7093458 ,
         3.32015376]),
 <BarContainer object of 10 artists>)
```

Learning Outcomes:

1.Understand the concept of normal distribution and its characteristics.

2.Demonstrate knowledge of histograms as a graphical representation of data distribution.

3.Gain proficiency in programming and implementing normal distribution and histogram algorithms.

4.Apply statistical measures to analyze and interpret data using normal distribution and histograms.

5.Visualize and communicate data patterns and distribution characteristics effectively.