Data Visualization using matplotlib

When data is shown in the form of pictures, it becomes easy for the user to understand it. Representing the data in the form of pictures or graphs is called 'data visualization'. For this purpose, we can use pyplot submodule of the matplotlib module. For complete information about this module, you can refer to the page:

```
https://matplotlib.org/api/pyplot_summary.html
```

Bar Graph

A bar graph represents data in the form of vertical or horizontal bars. It is useful to compare the quantities. Now, let us see how to create a bar graph with employee id numbers on X-axis and their salaries on Y-axis. We will take this data from the data frame. First of all, we should import the packages as:

```
import matplotlib.pyplot as plt   # for drawing the graph
import pandas as pd   # for creating the data frame
```

Suppose the data frame is created from 'empdata' as:

```
df = pd.DataFrame(empdata)
```

This data frame 'df' contains various pieces of data like empid, ename, sal and doj. We want to take only the 'empid' and 'sal' for the purpose of drawing the graph. This is done as:

```
x = df['empid']
y = df['sal']
```

We can draw the bar graph by calling bar() function of pyplot module as:

```
plt.bar(x, y, label='Employee data', color='red')
```

We can display a label (or title) on X-axis and Y-axis using xlabel() and ylabel) functions as:

```
plt.xlabel('Employee ids')
plt.ylabel('Employee salaries')
```

We can provide company's title using title() function as:

```
plt.title('MICROSOFT INC')
```

We can also display legend that refers to colors and labels of data especially when different sets of data are plotted as:

```
plt.legend()
```

Finally, the graph can be displayed by calling the show() function as:

```
plt.show()
```

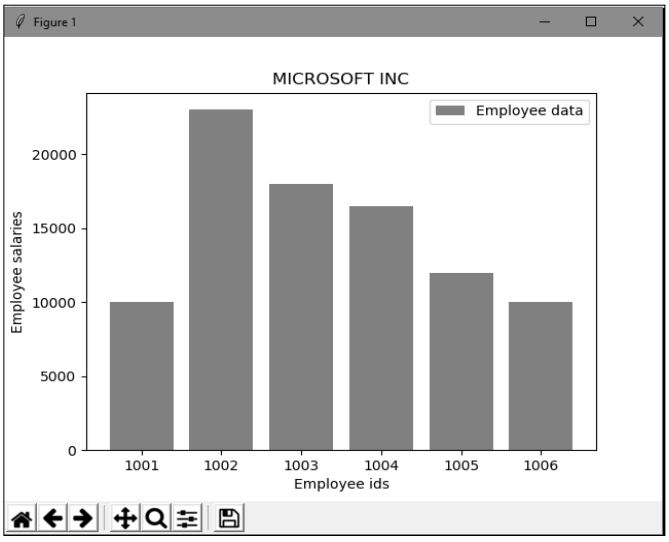
Program

Program 1: A Python program to display employee id numbers on X-axis and their salaries on Y-axis in the form a bar graph.

```
      # bar graph with ids, salaries
      import matplotlib.pyplot as plt
      import pandas as pd

      # take employee data as a dictionary
      empdata = {"empid": [1001, 1002, 1003, 1004, 1005, 1006],
      "ename": ["Ganesh Rao", "Anil Kumar", "Gaurav Gupta", "Hema Chandra",
"Laxmi Prasanna", "Anant Nag"],
      "sal": [10000, 23000.50, 18000.33, 16500.50, 12000.75, 9999.99],
      "doj": ["10-10-2000", "3-20-2002", "3-3-2002", "9-10-2000", "10-8-
2000", "9-9-1999"]}

      # create a data frame
      df = pd.DataFrame(empdata)

      # extract empid and salary data into x and y vars
      x = df['empid']
      y = df['sal']

      # create bar graph
      plt.bar(x,y, label='Employee data', color='red')

      # set x and y axis labels
      plt.xlabel('Employee ids')
      plt.ylabel('Employee salaries')

      # set company title
      plt.title('MICROSOFT INC')

      # show legend
      plt.legend()

      # display the graph
      plt.show()
```

Output:

```
      C:\> python bar.py
```



From the above output, we can infer that the employee with id number 1002 is drawing the highest salary and next to him the employee with id number 1003.

We can create bar graphs from more than one data set that are coming from multiple data frames. For example, we can plot the empid and salaries for 2 departments: Sales team and Production team. For this purpose, we can call the bar() function two times as:

```
plt.bar(x,y, label='Sales dept', color='red')
plt.bar(x1, y1, label='Production dept', color='green')
```

Program

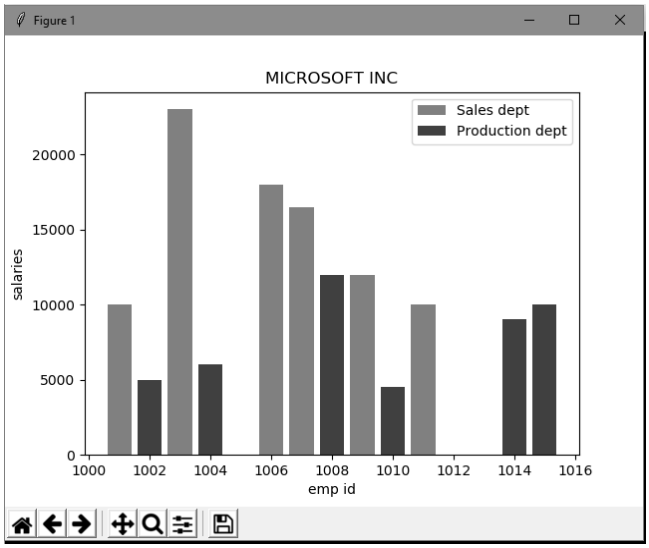
Program 2: A program to display employee id numbers on X-axis and their salaries on Y-axis in the form of a bar graph for two departments of a company.

```
# bar graph with ids, salaries for two depts
import matplotlib.pyplot as plt

# take employee ids and salaries for Sales dept
x = [1001, 1003, 1006, 1007, 1009, 1011]
y = [10000, 23000.50, 18000.33, 16500.50, 12000.75, 9999.99]

# take employee ids and salaries for Production dept
x1 = [1002, 1004, 1010, 1008, 1014, 1015]
y1 = [5000, 6000, 4500.00, 12000, 9000, 10000]

# create bar graphs
plt.bar(x,y, label='Sales dept', color='red')
plt.bar(x1, y1, label='Production dept', color='green')

# set the labels and legend
plt.xlabel('emp id')
plt.ylabel('salaries')
plt.title('MICROSOFT INC')
plt.legend()
```

```
# display the bar graph
plt.show()
```

Output:

```
C:\> python bars.py
```



From the above output, we can understand that the Sales department people are drawing more salaries when compared to those of the Production department.

Histogram

Histogram shows distributions of values. Histogram is similar to bar graph but it is useful to show values grouped in bins or intervals. For example, we can collect the age of each employee in a company and show it in the form of a histogram to know how many employees are there in the range of 0-10 years, 10-20 years, etc. For this purpose, we should take the employee ages and bins (or intervals) as:

```
emp_ages = [22,45,30,59,58,56,57,45,43,43,50,40,34,33,25,19]
bins = [0,10,20,30,40,50,60]   # of 10 years' range
```

The bins list contains 0,10,20... This indicates the ranges from 0 to 9, 10 to 19, etc., excluding the outer limit. To draw the histogram, we should use hist() function as:

```
plt.hist(emp_ages, bins, histtype='bar', rwidth=0.8, color='cyan')
```

Here, histtype is 'bar' to show the histogram in the form of bars. The other types can be 'barstacked', 'step', 'stepfilled'. The option rwidth = 0.8 indicates that the bar's width is 80%. There will be a gap of 10% space before and after the bar. If this is decreased, then the bar's width will be narrowed.

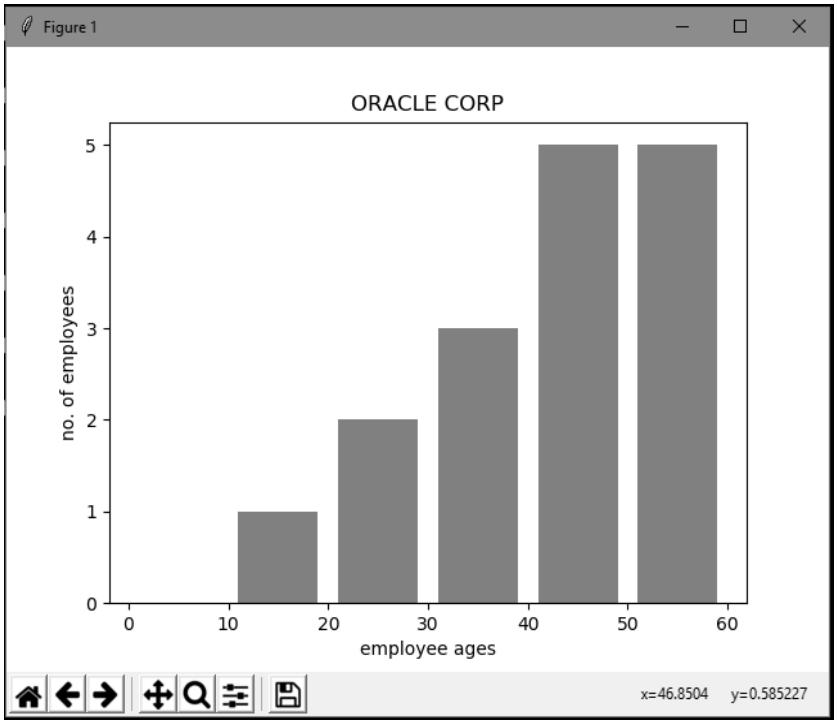
Program

Program 3: A program to display a histogram showing the number of employees in specific age groups.

```
# histogram of employee ages
import matplotlib.pyplot as plt

# take individual employee ages and range of ages
emp_ages = [22,45,30,59,58,56,57,45,43,43,50,40,34,33,25,19]
bins = [0,10,20,30,40,50,60]

# create histogram of bar type
plt.hist(emp_ages, bins, histtype='bar', rwidth=0.8, color='cyan')

# set labels
plt.xlabel('employee ages')
plt.ylabel('no. of employees')
plt.title('ORACLE CORP')
plt.legend()

# draw the histogram
plt.show()
```

Output:

```
C:\> python histo.py
```



From the output, we can understand that this company has more employees in the age group of 40 to 59 years. It infers that most employees in the company are above the age of 40.

Creating a Pie Chart

A pie chart shows a circle that is divided into sectors and each sector represents a proportion of the whole. For example, we can take different departments in a company and their employees. Suppose, there are 4 departments and their employees are in the percentages of 50%, 20%, 15% and 15%. These can be represented as slices in a pie chart.

To create a pie chart, we can use pie() function as:

```
      plt.pie(slices, labels=depts, colors=cols, startangle=90, explode=(0,
0.2, 0, 0), shadow=True, autopct='%.1f%%' )
```

Here, labels=depts indicates a list of labels. colors=cols indicates a list of colors. startangle=90 indicates that the pie chart will start at 90 degrees (12 o'clock position). If this option is not mentioned, then it starts by default at 0 degrees (3 o'clock position).

The option explode=(0,0.2, 0,0) indicates whether the slices in the pie chart should stand out or not. Here, 0 indicates that the corresponding slide will not come out of the chart. But the next 0.2 indicates that the second slice should come out slightly. Its values 0.1 to 1 indicate how much it should come out of the chart.

The option shadow=True indicates that the pie chart should be displayed with a shadow. This will improve the look of the chart. The option autopct='%.1f%%' indicates how to display the percentages on the slices. Here, %.1 indicates that the percentage value should be displayed with 1 digit after decimal point. The next two % symbols indicate that only one symbol is to be displayed.

Program

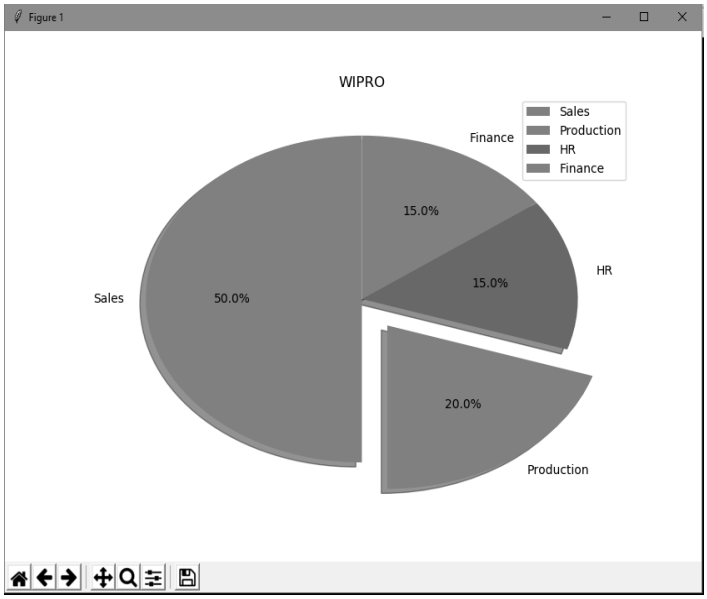
Program 4: A program to display a pie chart showing the percentage of employees in each department of a company.

```
      # to display employees of different depts in pie chart
      import matplotlib.pyplot as plt

      # take the percentages of employees of 4 departments
      slices = [50, 20, 15, 15]

      # take the departments names
      depts = ['Sales', 'Production', 'HR', 'Finance']

      # take the colors for each department
      cols = ['magenta', 'cyan', 'brown', 'gold']

      # create a pie chart
      plt.pie(slices, labels=depts, colors=cols, startangle=90, explode=(0,
0.2, 0, 0), shadow=True, autopct='%.1f%%' )

      # set titles and legend
      plt.title('WIPRO')
```

```
plt.legend()

# show the pie chart
plt.show()
```

Output:

```
C:\> python pie.py
```



The graph clearly shows that 50% of the employees in the company are from the Sales department. Also, we want to grab the focus on Production department. Hence, that slice is dragged out.

Creating Line Graph

A line graph is a simple graph that shows the results in the form of lines. To create a line graph, we need x and y coordinates. For example,

```
plt.plot(x, y, 'colorname')
```

Here, the plot() function creates the line graph. x and y represent lists of coordinates for X- and Y-axes. 'colorname' indicates the color name to be used for the line.

Program

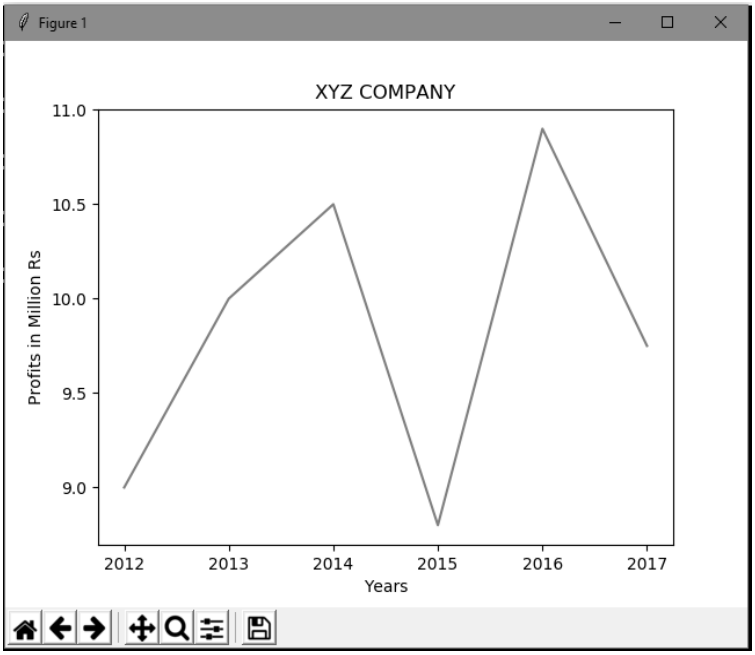
Program 5: A program to create a line graph to show the profits of a company in various years.

```
# to display profits of a company year-wise
import matplotlib.pyplot as plt

years = ['2012', '2013', '2014', '2015', '2016', '2017']
profits = [9, 10, 10.5, 8.8, 10.9, 9.75]
```

```
# create the line graph
plt.plot(years, profits, 'blue')

# set title and labels
plt.title('XYZ COMPANY')
plt.xlabel('Years')
plt.ylabel('Profits in Million Rs')

# show the line chart
plt.show()
```

Output

```
C:\> python line.py
```



From the above output, we can understand that the profits of the company are the highest during the year 2016 and the least during the year 2015.