Assignment: Data Visualization using Histograms in Python

Dataset: Use the following marks data:

```
result = [87, 88, 79, 80, 94, 77, 82, 82, 68, 74, 91, 80, 80, 74, 83]
```

Or load from Excel:

```
import pandas as pd
df = pd.read_excel("for_histo.xlsx")
result = df["result"].to_numpy()
```

Part A: Basic Histogram

1. Plot a basic histogram using 10 bins and a range of 50 to 100.
2. Customize the bar width using rwidth=0.8 and change the color of the bars.
3. Plot a horizontal histogram of the same data using orientation='horizontal'.

Part B: Bin and Range Manipulation

4. Plot histograms using:

- bins=5
- bins=20 Observe and note the difference in bar widths and distribution.

5. Change the range to:
    - (51, 100)
    - (60, 95) How does changing the range affect your histogram?

Part C: Density and Cumulative Histograms

6. Plot a histogram with density=True. What is shown on the y-axis?

7. Plot a histogram with cumulative=True. What changes?

8. Plot a histogram with both density=True and cumulative=True. What does this graph represent?

Part D: Comparing Two Datasets

9. Create a second dataset:

```
result2 = [72, 75, 70, 65, 80, 85, 90, 92, 78, 88, 69, 84, 81, 76, 79]
```

10. Plot both result and result2 on the same histogram with labels and colors. Use alpha for transparency and plt.legend() to differentiate.
11. Now plot the two datasets with stacked=True. What is the key difference in the display?

Assignment: Visualizing Data Using Scatter Plots in Python

Dataset: Use the following example data (or create your own with similar structure):

```
standard = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
rank =     [10, 9, 7, 5, 3, 4, 6, 2, 1, 8]
```

Part A: Basic Scatter Plot

6. Plot a basic scatter plot with `standard` on the X-axis and `rank` on the Y-axis.
7. Label the X-axis as “Standard” and Y-axis as “Rank”, and add a suitable title.

Part B: Customizing Marker Style

3. Change the marker style to square (`marker='s'`) and triangle (`marker='^'`) in two separate plots.

4. Adjust the marker size using the `s` parameter.

Part C: Color Customization

5. Use the `color='green'` parameter to color all points.

6. Use the `c=rank` option to color the points based on the rank value and add a color bar.

7. Try different `cmap` values like `'plasma'`, `'cool'`, and `'viridis'`.

Part D: Transparency and Edge Colors

8. Use `alpha=0.5` to make markers semi-transparent.

9. Add edge colors using `edgecolors='black'` and adjust `linewidths=1.5`.

Part E: Advanced Plotting

10. Create a new list `performance = [45, 55, 65, 75, 85, 95, 70, 60, 50, 80]`.

11. Use `performance` to control the size of each point using the `s` parameter.

12. Combine customization: plot `standard` vs `rank`, color by `performance`, size by `performance`, and add color bar.

Part F: Multiple Groups

13. Create a second dataset:

```
standard2 = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
rank2 =     [9, 7, 6, 5, 4, 3, 2, 1, 8, 10]
```

14. Plot both groups (`standard, rank` and `standard2, rank2`) on the same figure with different colors and markers.
15. Add a legend to distinguish between the two groups.

Assignment: Visualizing Data Using Pie Charts in Python

Dataset: Use the following dataset or create your own:

```
subjects = ['Math', 'Science', 'English', 'History', 'Computer']
marks = [80, 70, 90, 60, 100]
```

Part A: Basic Pie Chart

8. Plot a basic pie chart using the above data.
9. Add `labels=subjects` and display percentage using `autopct='%1.1f%%'`.
10. Add a title to the chart: “Subject-wise Marks Distribution”.

Part B: Customization

4. Change the start angle to `startangle=90` and observe the effect.
5. Make the chart circular by setting `plt.axis('equal')`.
6. Add `explode` to highlight the highest scoring subject.

Part C: Styling and Legends

7. Change colors of the slices using a list of custom color names.
8. Add a legend outside the chart using `bbox_to_anchor`.
9. Add a shadow to the pie chart.

Assignment: Creating Bar and Horizontal Bar Charts in Python

Dataset: Use the following or your own:

```
subjects = ['Math', 'Science', 'English', 'History', 'Computer']
marks = [80, 70, 90, 60, 100]
```

Part A: Vertical Bar Chart (bar)

11. Create a simple bar chart showing marks for each subject.
12. Add axis labels and chart title.
13. Change the color of each bar using a list of color values.
14. Add value labels on top of each bar.

Part B: Horizontal Bar Chart (barh)

5. Create a horizontal bar chart for the same data.

6. Add title and label axes appropriately.

7. Reverse the order of bars (from highest to lowest marks).

Part C: Styling and Spacing

8. Adjust the bar width using the `width` parameter.

9. Use the `edgecolor` and `linewidth` parameters to outline bars.

10. Display grid lines behind bars.

Part D: Comparative Bars 11. Create another dataset:

```
marks2 = [85, 65, 95, 70, 90]  # Marks from another exam
```

12. Create grouped bar chart to compare `marks` and `marks2` side by side.
13. Create stacked bar chart to show combined marks of both exams.