Q1. If you have any, what are your choices for increasing the comparison between different figures on the same graph?

Ans1- To increase the comparison between different figures on the same graph, you can consider the following options:

Use different colors or patterns for each figure to make them visually distinct.

Add labels, legends, or annotations to clarify what each figure represents.

Adjust the axis scales or limits to ensure that all figures are visible and properly scaled.

Q2. Can you explain the benefit of compound interest over a higher rate of interest that does not compound after reading this chapter?

Ans2- Compound interest offers the benefit of earning interest not only on the initial principal amount but also on the accumulated interest from previous periods. This compounding effect allows investments to grow faster over time compared to simple interest, where interest is only calculated on the initial principal. Compound interest can lead to exponential growth, making it more advantageous for long-term investments, such as savings accounts, loans, or investments, as it maximizes returns and increases wealth over time.

Q3. What is a histogram, exactly? Name a numpy method for creating such a graph.

Ans3- A histogram is a graphical representation of the distribution of data, showing the frequency or count of values within specific intervals or bins. Each bar in a histogram represents the number of data points falling within a particular range. In NumPy, you can create a histogram using the numpy.histogram() method. It takes data as input and returns an array of histogram values and bin edges, which can then be plotted using libraries like Matplotlib to visualize the distribution.

Q4. If necessary, how do you change the aspect ratios between the X and Y axes?

Ans4- To change the aspect ratios between the X and Y axes in a plot, you can use Matplotlib, a popular plotting library in Python. You can achieve this by setting the aspect ratio of the plot using the matplotlib.pyplot.gca().set\_aspect() method. By specifying the aspect ratio as a ratio or a string (e.g., 'equal' for equal aspect ratio), you can control how the X and Y axes are scaled, ensuring that the plot maintains the desired aspect ratio.

Q5. Compare and contrast the three types of array multiplication between two numpy arrays: dot product, outer product, and regular multiplication of two numpy arrays.

Ans5- In NumPy, there are three types of array multiplication between two arrays:

Dot Product: This is performed using numpy.dot() or the @ operator in Python 3. It calculates the sum of the element-wise products of two arrays and is used for matrix multiplication.

Outer Product: The outer product is obtained using numpy.outer(), and it computes the product of all pairs of elements from two arrays, resulting in a new array.

Regular Multiplication: Regular multiplication (\* operator) performs element-wise multiplication between two arrays, where each element in the result is the product of the corresponding elements in the input arrays. It is not matrix multiplication.

Q6. Before you buy a home, which numpy function will you use to measure your monthly mortgage payment?

Ans6- To calculate your monthly mortgage payment before buying a home, you can use the NumPy financial function numpy.pmt(rate, nper, pv). Here, rate represents the monthly interest rate, nper is the number of monthly payments, and pv is the principal amount (loan amount). This function computes the monthly payment needed to pay off a loan with equal monthly payments over the specified loan term, taking into account the interest rate.

Q7. Can string data be stored in numpy arrays? If so, list at least one restriction that applies to this data.

Ans7- Yes, string data can be stored in NumPy arrays using the data type numpy.str\_ or numpy.string\_. However, there are some restrictions:

NumPy arrays are designed for homogeneous data, so all elements in the array should have the same length (fixed-length strings) to ensure efficient storage.

If you try to store variable-length strings or strings with varying lengths in a NumPy array, it may result in truncation or padding with null bytes, leading to inefficient memory usage. To handle variable-length strings, consider using other data structures or libraries like pandas.