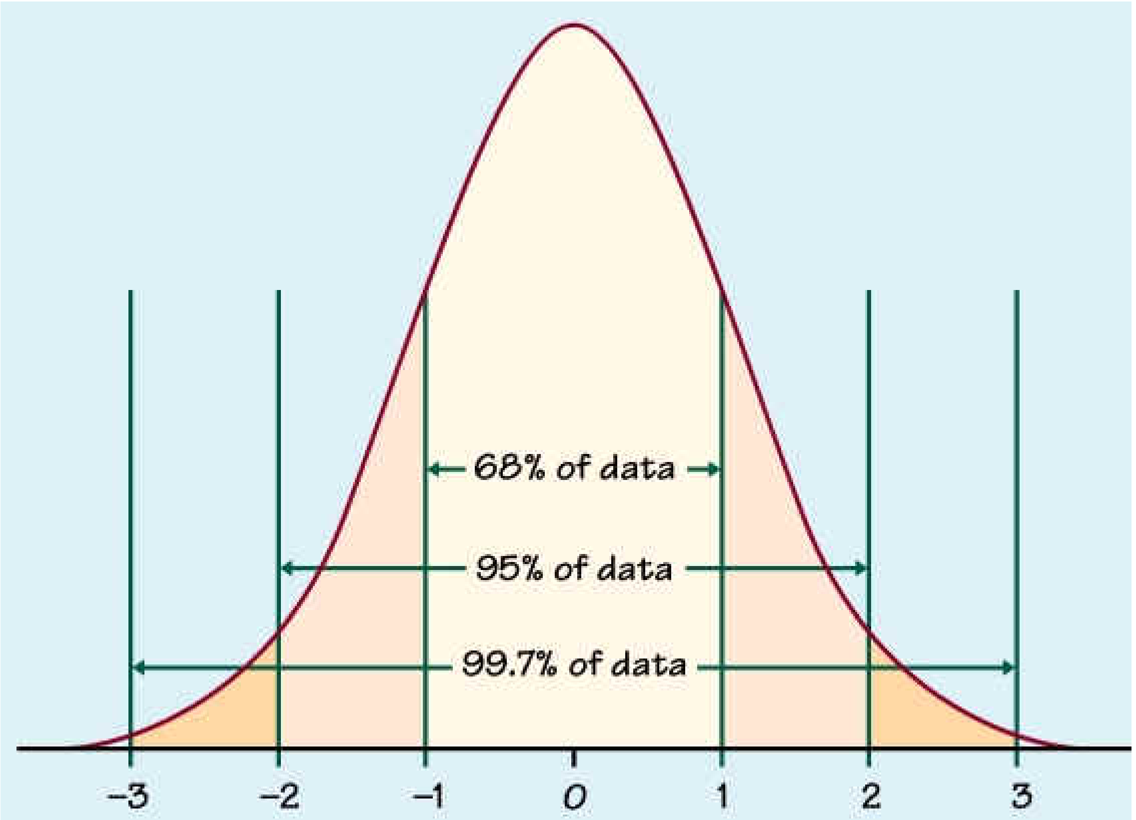
---------------------------Z-Scores in Python---------------------------



A measure of how many standard deviations below or above the population mean a raw score is called z score. It will be positive if the value lies above the mean and negative if it lies below the mean.

It is also known as standard score. It indicates how many standard deviations an entity is, from the mean. In order to use a z-score, the mean μ and also the population standard deviation σ should be known.

A z score helps to calculate the probability of a score occurring within a standard normal distribution. It also enables us to compare two scores that are from different samples. We use the following formula to calculate a z-score:

Formula :-

$$z = (X - \mu)/\sigma$$

Details are in below:-

where:

➤ *X is a single raw data value*

➤ *μ is the population mean*

➤ *σ is the population standard deviation*

Uses

- Understand where a data point fits into a distribution.
- Compare observations between dissimilar variables.
- Identify outliers
- Calculate probabilities and percentiles using the standard normal distribution.

How to Calculate Z-Scores in Python

Here we will see how to use Z score in python.

We will also learn plotting for Z scores.

We can calculate z-scores in Python using scipy.stats.zscore, which uses the following syntax:

scipy.stats.zscore(a, axis=0, ddof=0, nan_policy='propagate')

where:

➤ **a**: *an array like object containing data*

➤ **axis**: *the axis along which to calculate the z-scores. Default is 0.*

➤ **ddof**: *degrees of freedom correction in the calculation of the standard deviation. Default is 0.*

➤ **nan_policy**: *how to handle when input contains nan. Default is propagate, which returns nan. 'raise' throws an error and 'omit' performs calculations ignoring nan values.*

The following examples illustrate how to use this function to calculate z-scores for one-dimensional numpy arrays, multi-dimensional numpy arrays, and Pandas DataFrames. We also visualize the zscores with vairous plot.

We will do it step by step process:

Numpy One-Dimensional Arrays

Below are the steps:-

Step 1: Import modules.

In [34]:
```
import pandas as pd
import numpy as np
import scipy.stats as stats
import matplotlib.pyplot as plt
```

Step 2: Create an array of values.

In [35]:
```
data = np.array([5, 7, 7, 12, 14, 14, 15, 16, 19, 22])
data
```

Out[35]:
```
array([ 5,  7,  7, 12, 14, 14, 15, 16, 19, 22])
```

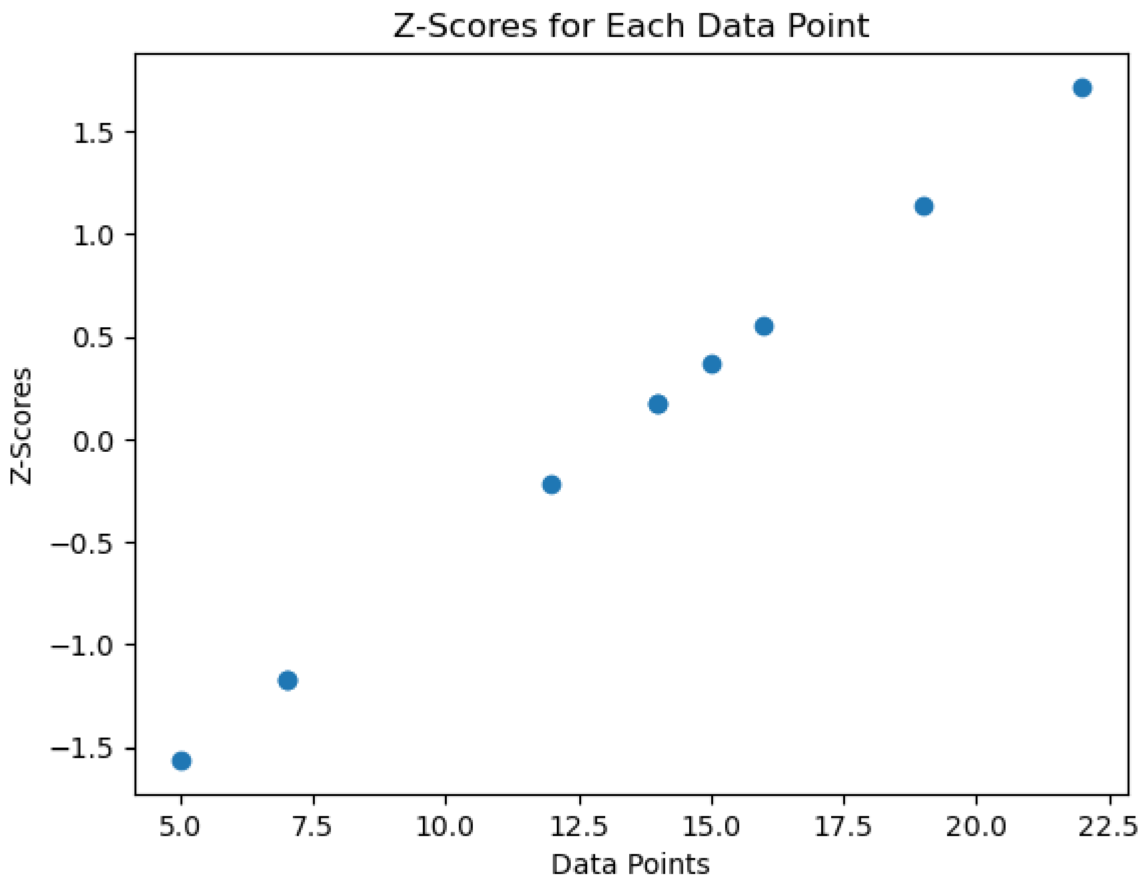
Step 3: Calculate the z-scores for each value in the array.

In [36]:
```python
Zscore = stats.zscore(data)
Zscore
```

Out[36]:
```
array([-1.56203089, -1.17634425, -1.17634425, -0.21212765,  0.17355899,
        0.17355899,  0.36640231,  0.55924563,  1.13777559,  1.71630554])
```

Step 4: Plotting the z-scores.

In [26]:
```python
plt.plot(data, Zscore, 'o')
plt.xlabel('Data Points')
plt.ylabel('Z-Scores')
plt.title('Z-Scores for Each Data Point')
plt.show()
```



Z-score is a statistical measurement that describes a value's relationship to the mean of a group of values. Each z-score tells us how many standard deviations away an individual value is from the mean.

Z-score is measured in terms of standard deviations from the mean. If a Z-score is 0, it indicates that the data point's score is identical to the mean score. A Z-score of 1.0 would indicate a value that is one standard deviation from the mean. Z-scores may be positive or negative, with a positive value indicating the score is above the mean and a negative score indicating it is below the mean. Z-scores are measures of an instrument's variability and can be used by traders to help determine volatility

For example:

- The first value of “5” in the array is -1.56203089 standard deviations below the mean.
- The fourth value of “12” in the array is -0.21212765 standard deviations below the mean.
- The last value of “22” in the array is 1.71630554 standard deviations above the mean.

Numpy Multi-Dimensional Arrays

If we have a multi-dimensional array, we can use the axis parameter to specify that we want to calculate each z-score relative to its own array. For example, suppose we have the following multi-dimensional array:

In [37]:
```
newdata = np.array([[5, 6, 7, 7, 8],
                    [8, 8, 8, 9, 9],
                    [2, 2, 4, 4, 5]])
```

We can use the following syntax to calculate the z-scores for each array:

In [38]:
```
z_scores = stats.zscore(newdata, axis=1)
z_scores
```

Out[38]:
```
array([[-1.56892908, -0.58834841,  0.39223227,  0.39223227,  1.37281295],
       [-0.81649658, -0.81649658, -0.81649658,  1.22474487,  1.22474487],
       [-1.16666667, -1.16666667,  0.5       ,  0.5       ,  1.33333333]])
```

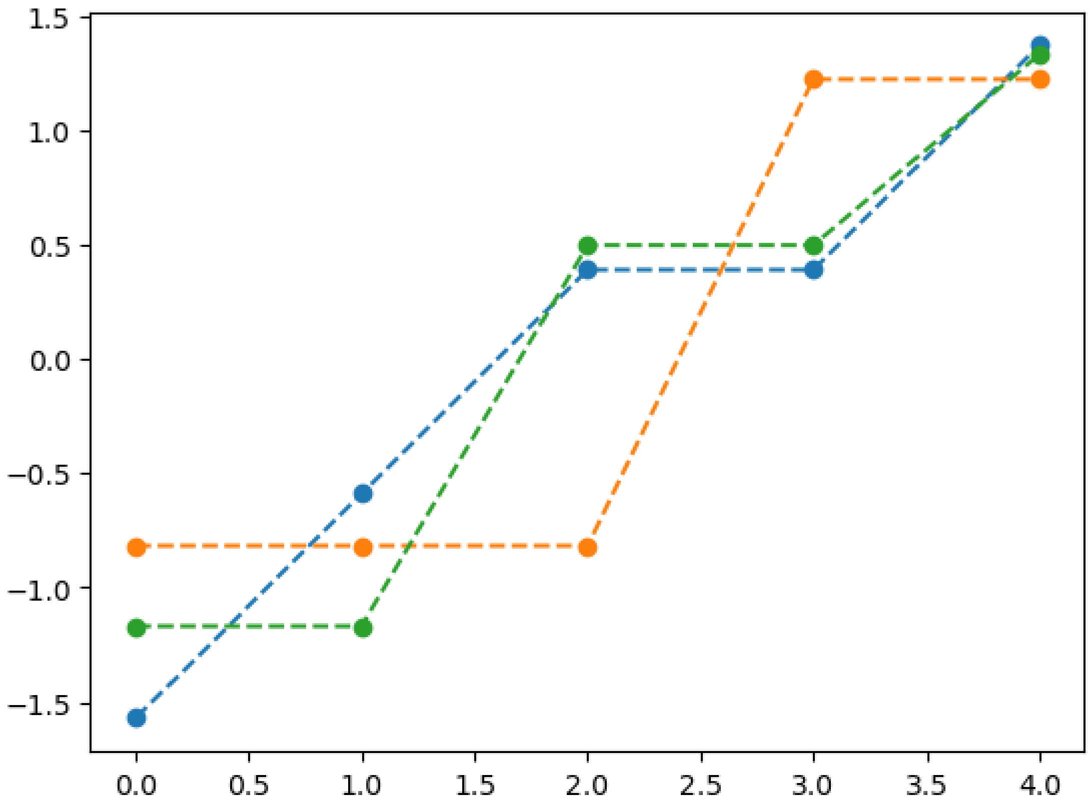
The z-scores for each individual value are shown relative to the array they’re in.

For example:

- The first value of “5” in the first array is -1.56892908 standard deviations below the mean of its array.
- The first value of “8” in the second array is -0.81649658 standard deviations below the mean of its array.
- The Last value of “2” in the third array is 1.33333333 standard deviations above the mean of its array.
- The Last value of “9” in the second array is 1.22474487 standard deviations above the mean of its array.

Plotting z-scores

In [39]:
```
for i in range(z_scores.shape[0]):
    plt.plot(z_scores[i], 'o--')
plt.show()
```



Pandas DataFrames

Suppose we have a Pandas DataFrame:

In [40]:
```
pdata = pd.DataFrame(np.random.randint(0, 10, size=(5, 3)), columns=['A', 'B', 'C'])
pdata
```

Out[40]:

	A	B	C
0	1	9	4
1	8	2	5
2	6	7	0
3	7	1	8
4	8	3	2

We can use the apply function to calculate the z-score of individual values by column:

In [41]:
```
z_scores1 = pdata.apply(stats.zscore)
z_scores1
```

Out[41]:

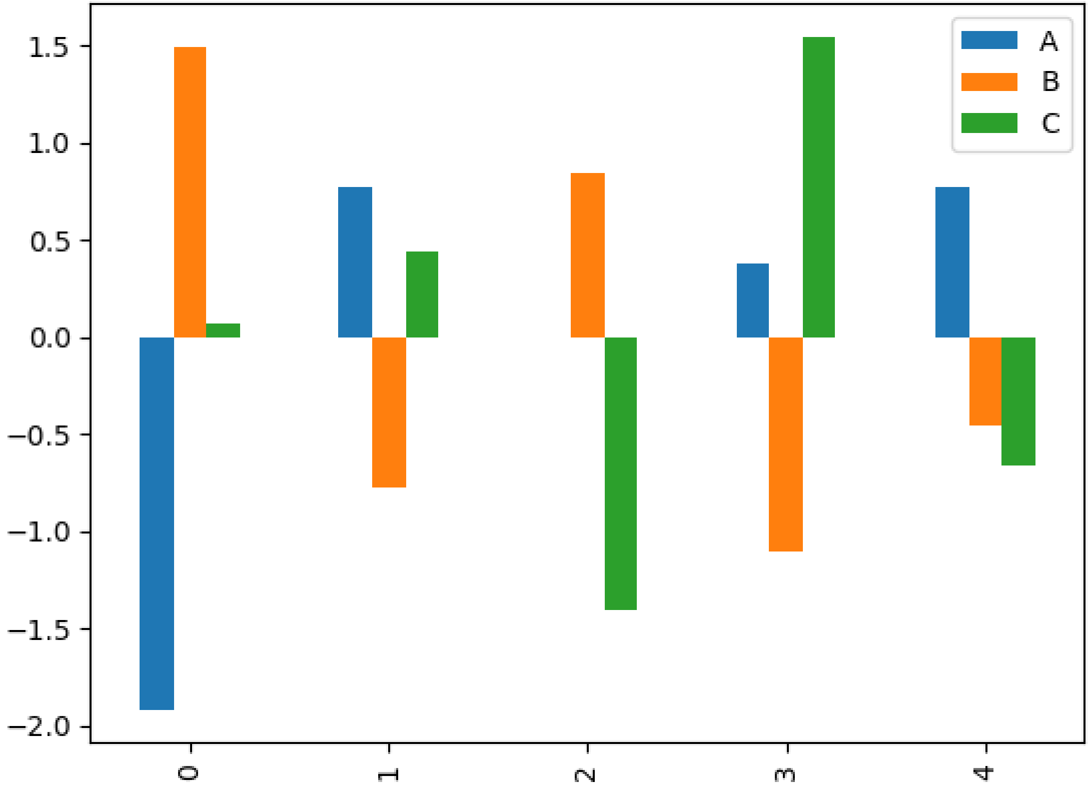
	A	B	C
0	-1.917412	1.497172	0.073721
1	0.766965	-0.781133	0.442326
2	0.000000	0.846228	-1.400699
3	0.383482	-1.106606	1.548141
4	0.766965	-0.455661	-0.663489

The z-scores for each individual value are shown relative to the column they're in. For example:

- The first value of "2" in the first column is -0.267261 standard deviations below the mean value of its column.
- The third value of "0" in the second column is -1.703886 standard deviations below the mean value of its column.
- The first value of "7" in the third column is 1.319824 standard deviations above the mean value of its column.

Plotting z-scores

In [42]:
```
z_scores1.plot(kind="bar")
plt.show()
```



Z-Scores vs. Standard Deviation

In most large data sets (assuming a normal distribution of data), 99.7% of values lie between -3 and 3 standard deviations, 95% between -2 and 2 standard deviations, and 68% between -1 and 1 standard deviations.

Standard deviation indicates the amount of variability (or dispersion) within a given data set. For instance, if a sample of normally distributed data had a standard deviation of 3.1, and another had one of 6.3, the model with a standard deviation (SD) of 6.3 is more dispersed and would graph with a lower peak than the sample with an SD of 3.1.

A distribution curve has negative and positive sides, so there are positive and negative standard deviations and z-scores. However, this has no relevance to the value itself other than indicating which side of the mean it is on. A negative value means it is on the left of the mean, and a positive value indicates it is on the right.

The z-score shows the number of standard deviations a given data point lies from the mean. So, standard deviation must be calculated first because the z-score uses it to communicate a data point's variability.

Z-Score in Real Life

A z-score is used in many real-life applications, such as medical evaluations, test scoring, business decision-making, and investing and trading opportunity measurements. Traders that use statistical measures like z-scores to evaluate trading opportunities are called quant traders

What Is a Good Z-Score?

The higher (or lower) a z-score is, the further away from the mean the point is. This isn't necessarily good or bad; it merely shows where the data lies in a normally distributed sample. This means it comes down to preference when evaluating an investment or opportunity. For example, some investors use a z-score range of -3.0 to 3.0 because 99.7% of normally distributed data falls in this range, while others might use -1.5 to 1.5 because they prefer scores closer to the mean.

Why Is Z-Score So Important?

A z-score is important because it tells where your data lies in the data distribution. For example, if a z-score is 1.5, it is 1.5 standard deviations away from the mean. Because 68% of your data lies within one standard deviation (if it is normally distributed), 1.5 might be considered too far from average for your comfort.

--------------------------------The End----------------------------------

```
In [ ]:
```