In [1]:
```
import numpy as np
import matplotlib.pyplot as plt
import scipy.stats as stats
from scipy.stats import norm
from scipy.stats import t
import math
```

In [2]:
```
data = [3, -3, 3, 12, 15, -16, 17, 19, 23, -24, 32]
n = len(data)
vector = np.array(data)
```

# Q1 & Q2 ::

- look at the output of the block bellow for the answers
- look for the tag `Problem # ::` in the Output not the Code For the answers

In [3]:

```
sample_size = vector.size
sample_mean = np.mean(vector)

df = n - 1
prob = t.cdf(vector, df)

interval = 0.95
retvec = stats.sem(vector) * t.ppf((1 + interval)/2, df)
conft1 = ((sample_mean - retvec), (sample_mean + retvec))

print(f"Problem 1 :: Z-test or T-test?")
print(f"\tIt is better to use a t-test for 2 main reasons.\n"
     +f"\n\t1)We don't know the Standard Deviation or Population Mean."
     +f"\n\t2)We have {n} data points, and {n} < 30. This means we have a very sm
all sample size")

print(f"\nSample Information")
print(f"\tSample Size: {sample_size}")
print(f"\tSample Mean: {sample_mean}")


print(f"\nProblem 1 :: interval: {interval}")
print(f"The Number of points Corresponding to an Interval of {interval}:\n\t{conf
t1}")

interval = 0.90
retvec = stats.sem(vector) * t.ppf((1 + interval)/2, df)

conft2 = ((sample_mean - retvec), (sample_mean + retvec))

print(f"\nProblem 2 :: interval: {interval}")
print(f"The Number of points Corresponding to an Interval of {interval}:\n\t{conf
t2}")

confdelta = ((conft1[0]-conft2[0]),(conft1[1]-conft2[1]))
print(f"\nComparison: -> difference btween The confidence intervals")
print(f"\t{confdelta}")

print(f"\nAnalysis:")
print("\tUsing the Comparison value it is clear that as the confidence interval c
hanginf from 0.95 to 0.9 has")
print("\tdecreased the size of the interval by %0.3f on both sides. This means th
at both intervals are "%(abs(confdelta[0])))
print("\tvery similar becuase the interval decreased by less than 4 point over al
l.")
```

```
Problem 1 :: Z-test or T-test?
        It is better to use a t-test for 2 main reasons.

        1)We don't know the Standard Deviation or Population Mean.
        2)We have 11 data points, and 11 < 30. This means we have a very small s
ample size

Sample Information
        Sample Size: 11
        Sample Mean: 7.363636363636363

Problem 1 :: interval: 0.95
The Number of points Corresponding to an Interval of 0.95:
        (-3.9470151490654715, 18.674287876338198)

Problem 2 :: interval: 0.9
The Number of points Corresponding to an Interval of 0.9:
        (-1.8369195722533416, 16.56419229952607)

Comparison: -> difference btween The confidence intervals
        (-2.11009557681213, 2.110095576812128)

Analysis:
        Using the Comparison value it is clear that as the confidence interval c
hanginf from 0.95 to 0.9 has
        decreased the size of the interval by 2.110 on both sides. This means th
at both intervals are
        very similar becuase the interval decreased by less than 4 point over al
l.
```

# Q3 ::

- look at the output of the block bellow for the answers
- look for the tag `Problem # ::` in the Output not the Code For the answers

In [4]:

```
#use a z test not 2 test because std is known
data = [3, -3, 3, 12, 15, -16, 17, 19, 23, -24, 32]
n = len(data)
vector = np.array(data)
sample_size = vector.size
sample_mean = np.mean(vector)
interval = 0.95
stdev = np.sqrt(16.836)

confz1 = ((sample_mean + (stdev/np.sqrt(sample_size))*(norm.ppf((1-interval)/2
))), (sample_mean - (stdev/np.sqrt(sample_size))*(norm.ppf((1-interval)/2))))

print(f"Problem 3 :: Reasoning:\n"
     + f"\tNow that the standard devation of {stdev} is known, it is more optimal
to use a z-test\n")

print(f"Sample Information")
print(f"\tSample Size: {sample_size}")
print(f"\tSample Mean: {sample_mean}")
print(f"\tStandard Deviation: {stdev}")

print(f"\nProblem 3 :: interval: {interval}")
print(f"The Number of points Corresponding to an Interval of {interval}:\n\t{conf
z1}")

sample_size = vector.size
sample_mean = np.mean(vector)
interval = 0.90
stdev = 16.836

confz2 = ((sample_mean + (stdev/np.sqrt(sample_size))*(norm.ppf((1-interval)/2
))), (sample_mean - (stdev/np.sqrt(sample_size))*(norm.ppf((1-interval)/2))))

print(f"\nProblem 3 :: interval: {interval}")
print(f"The Number of points Corresponding to an Interval of {interval}:\n\t{conf
z2}")


print("\nProblem 3 :: What is different or similar?")
print(f"Starting with the case of both the intervals found using a z-test:")
print(f"\tDecreasing the Interval Decreased the range of the lower and the upper
 bound")
print(f"t-test vs z-test:")
print(f"\tThe intervals found in both cases are similar. This however is only bec
ause the"+
    f"\n\tstandard deviation between both cases are very similar. The differences
most likely stem"+
    f"\n\tfrom the fact that z-test assumes a normal distribution while the t-tes
t does not, this"+
    f"\n\tis most likely why the 0.95 confidence interval is much thiner in this
 case")
```

```
Problem 3 :: Reasoning:
        Now that the standard devation of 4.10316950661315 is known, it is more
optimal to use a z-test

Sample Information
        Sample Size: 11
        Sample Mean: 7.363636363636363
        Standard Deviation: 4.10316950661315

Problem 3 :: interval: 0.95
The Number of points Corresponding to an Interval of 0.95:
        (4.9388626965002045, 9.788410030772521)

Problem 3 :: interval: 0.9
The Number of points Corresponding to an Interval of 0.9:
        (-0.9860436314852805, 15.713316358758007)

Problem 3 :: What is different or similar?
Starting with the case of both the intervals found using a z-test:
        Decreasing the Interval Decreased the range of the lower and the upper b
ound
t-test vs z-test:
        The intervals found in both cases are similar. This however is only beca
use the
        standard deviation between both cases are very similar. The differences
most likely stem
        from the fact that z-test assumes a normal distribution while the t-test
does not, this
        is most likely why the 0.95 confidence interval is much thiner in this c
ase
```

# Q4 ::

- look at the output of the block bellow for the answers
- look for the tag Problem # :: in the Output not the Code For the answers

In [5]:
```
df = n - 1
sample_size = np.size(vector)
sample_stdev = np.std(vector, ddof = 1)
sample_mean = np.mean(vector)
print(f"Sample Information")
print(f"\tSample Size: {sample_size}")
print(f"\tSample Mean: {sample_mean}")
print(f"\tStandard Deviation: {sample_stdev}")

vector = np.sort(vector)
prob = t.cdf(vector, df)
t_c = -sample_mean * -1/(sample_stdev/np.sqrt(sample_size))
prob_tc = t.cdf(t_c, df)
interval = abs(prob_tc * 2 * -1 + 1)
retvec = stats.sem(vector) * t.ppf((1 + interval)/2, df)
conf = ((sample_mean - retvec), (sample_mean + retvec))

print(f"\nProblem 4 :: The confidence required to assume they win on avg: \n\t{in
terval * 100}%")
print(f"\nProblem 4 :: The points at this interval are: \n\t{conf}")
```

```
Sample Information
	Sample Size: 11
	Sample Mean: 7.363636363636363
	Standard Deviation: 16.836108382121605

Problem 4 :: The confidence required to assume they win on avg:
	82.24752127627733%

Problem 4 :: The points at this interval are:
	(3.89910326248355e-13, 14.727272727272336)
```