**EX.NO: 05**

**Implementation of T-Test – one sample t-test**

**Date:**

**AIM:**

To write a program using Implementation of T-Test – one sample t – test

**Algorithm:**

**Step 1:** Create some dummy age data for the population of voters in the entire country

**Step 2:** Create Sample of voters in Minnesota and test the whether the average age of voters Minnesota differs from the population

**Step 3:** Conduct a t-test at a 95% confidence level and see if it correctly rejects the null hypothesis that the sample comes from the same distribution as the population.

**Step 4:** If the t-statistic lies outside the quantiles of the t-distribution corresponding to our confidence level and degrees of freedom, we reject the null hypothesis.

**Step 5:** Calculate the chances of seeing a result as extreme as the one being observed (known as the p-value) by passing the t-statistic in as the quantile to the stats.t.cdf() function

**Case:**

**5.A**

population\_ages1 = stats.poisson.rvs(loc=18, mu=35, size=150000)

population\_ages2 = stats.poisson.rvs(loc=18, mu=10, size=100000)

population\_ages = np.concatenate((population\_ages1, population\_ages2))

minnesota\_ages1 = stats.poisson.rvs(loc=18, mu=30, size=30)

minnesota\_ages2 = stats.poisson.rvs(loc=18, mu=10, size=20)

minnesota\_ages = np.concatenate((minnesota\_ages1, minnesota\_ages2))

print(population\_ages.mean() )

print(minnesota\_ages.mean() )

**5.B**

stats.ttest\_1samp(a = minnesota\_ages, # Sample data

popmean = population\_ages.mean()) # Pop mean

**5.C**

stats.t.ppf(q=0.025, # Quantile to check

df=49) # Degrees of freedom

**5.D**

stats.t.ppf(q=0.975, df=49)

**5.E**

stats.t.cdf(x= -2.5742, # T-test statistic

df= 49) \* 2 # Multiply by two for two tailed test \*

**5.F**

stats.t.interval(0.95, # Confidence level

df = 49, # Degrees of freedom

loc = minnesota\_ages.mean(), # Sample mean

scale= sigma) # Standard dev estimate

**5.G**

stats.t.interval(alpha = 0.99, # Confidence level

df = 49, # Degrees of freedom

loc = minnesota\_ages.mean(), # Sample mean

scale= sigma) # Standard dev estimate

**Output;**

**Results ;**