INTERPOLATION BY NEWTONS FORWARD AND BACKWARD METHOD

SCILAB ALGORITHM

```
//Newtons Forward-Backward Difference Interpolation
 clear;clc;
 x=1:1:8
 y=[1 8 27 64 125 216 343 512]
 xi=input('Enter the value of x ')
 n=length(x)
 h=x(2)-x(1)
 T=zeros(n,n)       //no need
 for i=1:n-1
    T(1:n-i,i)=diff(y',i) //forward
    //T2(i+1:n,i)=diff(y',i)   //backward
 end
//disp(T1)
//disp(T2)
u=(xi-x(1))/h           //forward
 //u=(xi-x(n))/h            //backward
 q=u
 yi=y(1)                 //forward
 //yi=y(n)                //backward
 for j=1:n-1
    yi=yi+(q*T(1,j))/factorial(j)  //forward
    //yi=yi+(q*T2(n,j))/factorial(j)  //backward
    q=q*(u-j)              //forward
    //q=q*(u+j)           //backward
 end
 printf("Interpolated Value at x=:%0.3f is %0.3f ",xi,yi)

scf(0)
clf
sizes = [50 100 150 200 250 300 350 400]
colors = ['g','b','w','y','r','g','y','m']
for i=1:length(x)
   scatter(x(i), y(i), sizes(i), colors(i), "fill")
end
scatter(xi, yi, 350, 'y', "fill")
xstring(xi+1, yi+40, "Interpolated Value")
xarrows([xi+0.1, xi+1], [yi, yi+40], 2, 2)
xpoly([min(x) max(x)], [yi yi], "lines", 2)
set(gca(),"foreground",5)
xstring(xi+0.2, yi+2, "y = " + string(yi))
xpoly([xi xi], [0 max(y)+50], "lines", 3)
xstring(xi-0.5, 0, "x = " + string(xi))
xlabel("Distance")
ylabel("Temperature")
title("Distance vs Temperature")
```

Python

```
import math as m
import numpy as np
import matplotlib.pyplot as plt
x = [10, 20, 30, 40, 50, 60, 70, 80]
y = [1, 8, 27, 64, 125, 216, 343, 512]
xi = 34
n = len(x)
h = x[1] - x[0]
u=(xi - x[0]) / h         #for forward
#u=(xi - x[-1]) / h       #for backward
yi = y[0]                 #for forward
#yi=y[-1]                 #for backward
T1 = np.zeros((n, n))
T1[:, 0] = y
for i in range(1, n):
    T1[0:n-i, i] = np.diff(y,i) #for forward difference
#     T1[i:n, i] = np.diff(y,i) #for Backward difference
print(T1)
q = u
for j in range(1, n):
    yi += (q * T1[0][j]) / m.factorial(j)  #for forward
    #yi += (q * T1[-1][j]) / m.factorial(j) #for backward
    q *= (u-j)  #for forward
    #q *= (u+j)  #for backward
plt.annotate('Interpolated Value',
             xy=(xi, yi),              # point to annotate
             xytext=(xi + 1, yi + 40), # position of text
             arrowprops=dict(arrowstyle='->', color='navy'),
             color='red',fontsize=12)
sizes=[50,100,150,200,250,300,350,400]
c=['g','b','ivory','khaki','r','g','khaki','m']
plt.title('Distance vs Temperature')
plt.scatter(x,y, s=sizes, color=c)
plt.scatter(xi,yi,s=350,color='y')
plt.xlabel('Distance')
plt.ylabel('Temperature')
plt.axhline(yi, color='m', linestyle='-.')
plt.text(xi + 0.2, yi + 2, f'y = {yi:.2f}', color='m',
fontsize=10)
plt.axvline(xi,color='y', linestyle=':')
plt.text(xi - 0.5, 0, f'x = {xi:.2f}', color='y', fontsize=10,
rotation=90)
plt.tight_layout()
plt.show()
```