Practice to make a polynmial in numpy.

```
import numpy as np
x=[1,2,3]
np.poly1d(x)
```

```
import numpy as np
x=[1,2,3] # here 1,2,3 are coefficients of the polynomial in descending order
Poly1=np.poly1d(x)
print(Poly1)

Poly2=np.poly1d(x,True) #another format to print polynomial
print(Poly2)
```

```
   2
1 x + 2 x + 3
   3     2
1 x - 6 x + 11 x - 6
```

Code to read data from a CSV file

```
import pandas as pd
import numpy as np

# Read data from CSV file
df = pd.read_csv('data.csv')

# Convert data to numpy arrays
x = df['x values'].values
y = df['y values'].values
```

```
import pandas as pd
import numpy as np

# Read data from CSV file
df = pd.read_csv('traffic.csv')
#df.head()

# Convert data to numpy arrays
x = df['Time'].values
y = df['No of vehicles'].values
```

New Section

Function for getting Lagrange Polynmial

```
# Function to calculate Lagrange polynomial
def lagrange_poly(x, y):
    n = len(x)
    p = np.poly1d(0.0)
    for i in range(n):
        L = np.poly1d(y[i])
        for j in range(n):
            if j != i:
                L *= np.poly1d([1.0, -x[j]]) / (x[i] - x[j])
        p += L
    return p

# Calculate Lagrange polynomial
# cuz had to make 6 degree polymonial -> selected first seven values
p = lagrange_poly(x[0:7], y[0:7])
print(p)
```

```
         6         5         4      3         2
0.07778 x - 1.433 x + 10.03 x - 33 x + 50.89 x - 29.57 x + 9
```

For Interpolating at a specific point

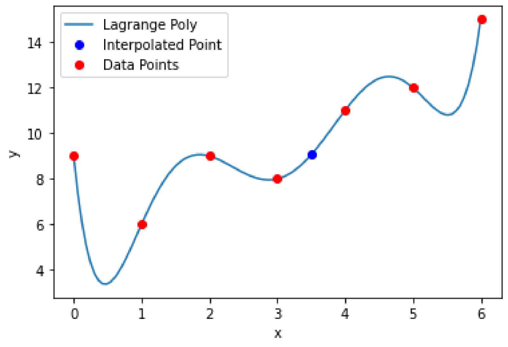
```
# Interpolate at a specific point
point = float(input("Enter x-coordinate to interpolate: "))
interp_value = p(point)

# Print Lagrange polynomial and interpolated value
print("Lagrange polynomial is:")
print(p)
print("Interpolated value at x =", point, "is:", interp_value)
```

```
Enter x-coordinate to interpolate: 3.5
Lagrange polynomial is:
         6         5         4      3         2
0.07778 x - 1.433 x + 10.03 x - 33 x + 50.89 x - 29.57 x + 9
Interpolated value at x = 3.5 is: 9.054687499999055
```

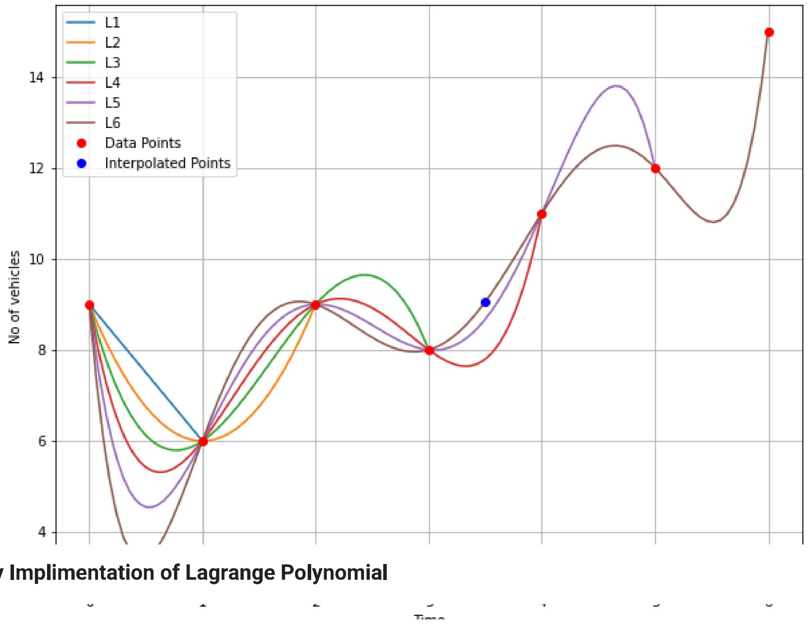
Plotting of Lagrange Polynomial

```
import matplotlib.pyplot as plt
xi=3.5
yi=9.054687499999055
p = lagrange_poly(x[0:7], y[0:7])
print(p)
xp=np.linspace(0, 6, 100)
yp=p(xp)

plt.plot(xp, yp, label='Lagrange Poly')
plt.plot(xi, yi, 'bo', label='Interpolated Point')
plt.plot(x[0:7], y[0:7], 'ro', label='Data Points')
plt.xlabel('x')
plt.ylabel('y')
plt.legend()
plt.show()
```

```
         6         5         4      3         2
0.07778 x - 1.433 x + 10.03 x - 33 x + 50.89 x - 29.57 x + 9
```



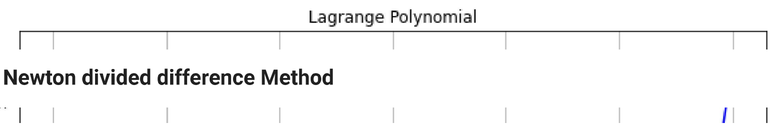
```
fig = plt.figure(figsize = (10,8))

n=6
for i in range(1,n+1,1):
  p = lagrange_poly(x[0:i+1], y[0:i+1])
  xp=np.linspace(0,x[i],100)
  yp=p(xp)
  plt.plot(xp, yp, label = f"L{i}")
plt.plot(x[0:7],y[0:7],'ro',label="Data Points")
plt.plot(xi,yi,'bo',label="Interpolated Points")
plt.xlabel('Time')
plt.ylabel('No of vehicles')
plt.legend()
plt.grid()
plt.show()
```



Scipy Implimentation of Lagrange Polynomial

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.interpolate import lagrange

# Define the Lagrange Polynomial
f = lagrange(x[0:7], y[0:7])

# Find P(3.5) by evaluating the polynomial at x=3.5
p_val = f(3.5)
print("P(3.5) =", p_val)

# Print the polynomial coefficients
print("Lagrange Polynomial:", np.poly1d(f).coefficients)

# Plot the Lagrange Polynomial and the data points
x_new = np.linspace(0, 6, 1000)
y_new = f(x_new)
fig = plt.figure(figsize = (10,8))
plt.plot(x_new, y_new, 'b', x[0:7], y[0:7], 'ro')
plt.plot(3.5, p_val, 'go', markersize=10)
plt.title('Lagrange Polynomial')
plt.grid()
plt.xlabel('Time')
plt.ylabel('No of vehicles')
plt.show()
```

```
P(3.5) = 9.054687499999055
Lagrange Polynomial: [  0.07777778  -1.43333333  10.02777778 -33.          50.89444444
 -29.56666667   9.        ]
```



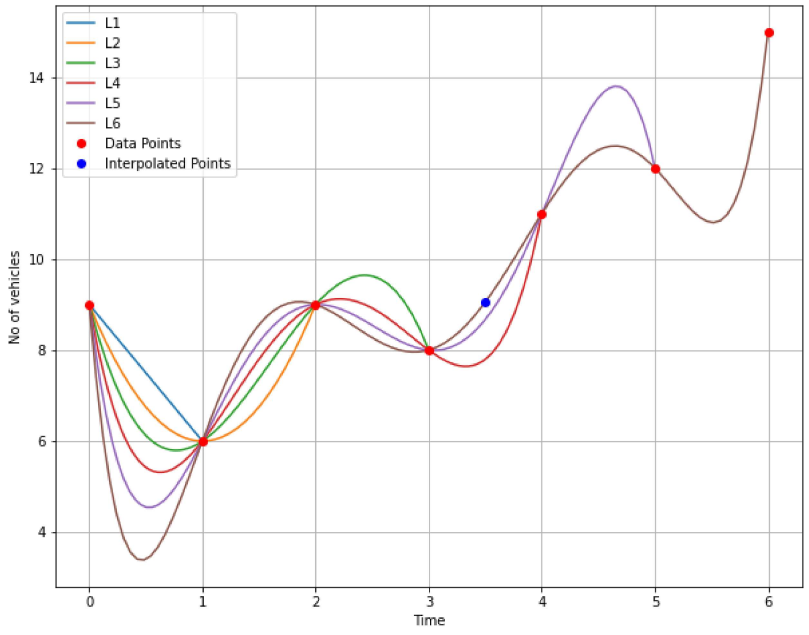
Code for Newton divided difference Method

```
import numpy as np
def divided_difference_table(x, y):
    n = len(x)
    F = [[0] * n for i in range(n)]
    for i in range(n):
        F[i][0] = y[i]
    for j in range(1, n):
        for i in range(j, n):
            F[i][j] = (F[i][j-1] - F[i-1][j-1]) / (x[i] - x[i-j])
    return F
def newton_div_dif_poly(x,y,xi):
   F=divided_difference_table(x,y) # Saving divided difference in a variable F
   n=len(x)
   prod=np.poly1d(1)
   N=np.poly1d(F[0][0])
   for i in range(1,n):
     prod=np.poly1d(x[0:i],True)
     N+=np.poly1d(F[i][i]*(prod.c))


   return (N)

ndd = newton_div_dif_poly(x[0:7], y[0:7], 3.5)


# Interpolate at a specific point
point = float(input("Enter x-coordinate to interpolate: "))
interp_value = ndd(point)

# Print the polynomial and interpolated value
print("polynomial is:")
print(ndd)
print("Interpolated value at x =", point, "is:", interp_value)
```

```
Enter x-coordinate to interpolate: 3.5
polynomial is:
         6         5         4      3         2
0.07778 x - 1.433 x + 10.03 x - 33 x + 50.89 x - 29.57 x + 9
Interpolated value at x = 3.5 is: 9.054687500000261
```

```
import matplotlib.pyplot as plt
xi=3.5
yi=9.054687499999055
p = newton_div_dif_poly(x[0:7], y[0:7], 3.5)
xp=np.linspace(0, 6,100)
yp=ndd(xp)

plt.plot(xp, yp, label='Newton dd Poly')
plt.plot(xi, yi, 'bo', label='Interpolated Point')
plt.plot(x[0:7], y[0:7], 'ro', label='Data Points')
plt.xlabel('x')
plt.ylabel('y')
plt.legend()
plt.show()
```

```
fig = plt.figure(figsize = (10,8))

n=6
for i in range(1,n+1,1):
  p = newton_div_dif_poly(x[0:i+1], y[0:i+1], 3.5)
  xp=np.linspace(0,x[i],100)
  yp=p(xp)
  plt.plot(xp, yp, label = f"L{i}")
plt.plot(x[0:7],y[0:7],'ro',label="Data Points")
plt.plot(xi,yi,'bo',label="Interpolated Points")
plt.xlabel('Time')
plt.ylabel('No of vehicles')
plt.legend()
plt.grid()
plt.show()
```



Lab Task: Given is a Traffic.xlsx file

1- Upload this file on Colab. (This file contains no of vehicles at a junction at each hour.)

2- Predict no of vehicles at 3.5 hour(Consider it a 24 hour clock) by using Lagrange with the help of a 6 degree polynomial.

3- Plot above polynomial with interpolating point.

4- Repeat above task (2-3) again with Newton Divided difference Method.

✓ 1s completed at 5:05 PM