LAB – 2 : INTERPOLATION / EXTRAPOLATION USING NEWTON'S FORWARD AND BACKWARD DIFFERENCE FORMULA

Let $y = f(x)$ be a function of $'x'$ in the given interval

x	x_0	x_1	x_2	x_3	$\cdots$	x_n
y	y_0	y_1	y_2	y_3	$\cdots$	y_n

FORWARD DIFFERENCE TABLE

x	y	Δy	$\Delta^2 y$	$\Delta^3 y$	$\cdots$	$\Delta^n y$
x_0	y_0	$\Delta y_0 = y_1 - y_0$	$\Delta^2 y_0 = \Delta y_1 - \Delta y_0$	$\Delta^3 y_0 = \Delta^2 y_1 - \Delta^2 y_0$	$\cdots$	$\Delta^n y_0 = \Delta^{n-1} y_1 - \Delta^{n-1} y_0$
x_1	y_1	$\Delta y_1 = y_2 - y_1$	$\Delta^2 y_1 = \Delta y_2 - \Delta y_1$	$\vdots$	$\cdots$	
x_2	y_2	$\Delta y_2 = y_3 - y_2$	$\vdots$	$\Delta^3 y_{n-3} = \Delta^2 y_{n-2} - \Delta^2 y_{n-3}$		
x_3	y_3	$\vdots$	$\Delta^2 y_{n-2} = \Delta y_{n-1} - \Delta y_{n-2}$			
$\vdots$	$\vdots$	$\Delta y_{n-1} = y_n - y_{n-1}$				
x_n	y_n					

NEWTON'S FORWARD INTERPOLATION FORMULA

$$y = y_0 + p\Delta y_0 + \frac{p(p-1)}{2!}\Delta^2 y_0 + \frac{p(p-1)(p-2)}{3!}\Delta^3 y_0 + \cdots + \frac{p(p-1)(p-2)\cdots(p-n+1)}{n!}\Delta^n y_0$$

Where $p = \dfrac{x - x_0}{h}$ and $h = x_n - x_{n-1}$

BACKWARD DIFFERENCE TABLE

x	y	∇y	$\nabla^2 y$	$\nabla^3 y$	$\cdots$	$\Delta^n y$
x_0	y_0	$\nabla y_1 = y_1 - y_0$	$\nabla^2 y_2 = \nabla y_2 - \nabla y_1$	$\nabla^3 y_3 = \nabla^2 y_3 - \nabla^2 y_2$	$\cdots$	$\nabla^n y_n = \nabla^{n-1} y_n - \nabla^{n-1} y_{n-1}$
x_1	y_1	$\nabla y_2 = y_2 - y_1$	$\nabla^2 y_3 = \nabla y_3 - \nabla y_2$	$\vdots$	$\cdots$	
x_2	y_2	$\nabla y_3 = y_3 - y_2$	$\vdots$	$\nabla^3 y_n = \nabla^2 y_n - \nabla^2 y_{n-1}$		
x_3	y_3	$\vdots$	$\nabla^2 y_n = \nabla y_n - \nabla y_{n-1}$			
$\vdots$	$\vdots$	$\nabla y_n = y_n - y_{n-1}$				
x_n	y_n					

NEWTON'S BACKWARD INTERPOLATION FORMULA

$$y = y_n + p\nabla y_n + \frac{p(p+1)}{2!}\nabla^2 y_n + \frac{p(p+1)(p+2)}{3!}\nabla^3 y_0 + \cdots + \frac{p(p+1)(p+2)\cdots(p+n-1)}{n!}\nabla^n y_n$$

Where $p = \dfrac{x - x_n}{h}$ and $h = x_n - x_{n-1}$

Program 1: Program to find the interpolating polynomial and $y(4)$ using the following data

x	1	2	3	4	5
y	10	26	58	112	194

```python
#Newton forward difference formula
from sympy import *
from numpy import *
n=int(input('Enter the number of data points: '))
x=zeros(n)
y=zeros((n,n))
f=[]

for i in range(0,n):
    x[i]=float(input(f'x{i}='))
    y[0][i]=float(input(f'y{i}='))

for j in range(0,n-1):
    for i in range(0,n-1-j):
        y[j+1][i]=y[j][i+1]-y[j][i]

for i in range(0,n):
    print(x[i], end=' ')
    for j in range(0,n-i):
        print('\t\t',y[j][i], end=' ')
    print()
h=x[1]-x[0]
t=Symbol('t')
p=(t-x[0])/h
pterms=1
for k in range (0,n-1):
    pterms=((p-k)/(k+1))*pterms
    f.append(pterms)
rhs=y[0][0]
for j in range (1,n):
    rhs=(rhs+f[j-1]*y[j][0])
print('The interpolating polynomial is:')
display(simplify(rhs))
xi=float(input('Enter value of x at which y should be determined: '))
print(f'y({xi}) = %0.4f'%rhs.subs(t,xi))
```

OUTPUT

```
Enter the number of data points: 5
x0=1
y0=10
x1=2
y1=26
x2=3
y2=58
x3=4
y3=112
x4=5
y4=194
1.0                 10.0                16.0                16.0                6.0
0.0
2.0                 26.0                32.0                22.0                6.0
3.0                 58.0                54.0                28.0
4.0                 112.0               82.0
5.0                 194.0
The interpolating polynomial is:
```

$$1.0t^3 + 2.0t^2 + 3.0t + 4.0$$

```
Enter value of x at which y should be determined: 1.4
y(1.4) = 14.8640
```

Program 2: Program to find the interpolating polynomial and $y(4)$ using the following data

x	0	1	2	3
y	1	2	1	10

```
#Newton backward difference formula
from sympy import *
from numpy import *
n=int(input('Enter the number of data points: '))
x=zeros(n)
y=zeros((n,n))
f=[]

for i in range(0,n):
    x[i]=float(input(f'x{i}='))
    y[0][i]=float(input(f'y{i}='))

for j in range(0,n-1):
    for i in range(0,n-1-j):
        y[j+1][i]=y[j][i+1]-y[j][i]

for i in range(0,n):
    print(x[i], end=' ')
    for j in range(0,n-i):
        print('\t\t',y[j][i], end=' ')
    print()
h=x[1]-x[0]
t=Symbol('t')
p=(t-x[n-1])/h
pterms=1
for k in range (0,n-1):
    pterms=((p+k)/(k+1))*pterms
    f.append(pterms)

rhs=y[0][n-1]
for j in range (1,n):
    rhs=(rhs+f[j-1]*y[j][n-1-j])
print('The interpolating polynomial is:')
display(simplify(rhs))
xi=float(input('Enter value of x at which y should be determined: '))
print(f'y({xi}) = %0.4f'%rhs.subs(t,xi))
```

OUTPUT

```
Enter the number of data points: 4
x0=0
y0=1
x1=1
y1=2
x2=2
y2=1
x3=3
y3=10
0.0             1.0             1.0             -2.0            12.0
1.0             2.0             -1.0            10.0
2.0             1.0             9.0
3.0             10.0
The interpolating polynomial is:
```

$$2.0t^3 - 7.0t^2 + 6.0t + 1.0$$

```
Enter value of x at which y should be determined: 4
y(4.0) = 41.0000
```

EXERCISE PROBLEMS

1. The area $'y'$ of the circle for different diameter $'x'$ are given below

x	80	85	90	95	100
y	5026	5674	6362	7088	7854

Find $y(82)$ and $y(98)$.

2. The population of a city is given by the table

year	1980	1990	2000	2010	2020
Population In thousand	19.96	39.65	58.81	77.21	94.61

Find the population in the year 1985 and 2015.