GROUP 7

Python 3 Program to Interpolate Using Newton Forward Difference Interpolation

Method 1: By prompting the user to input the values for x and y flexibly;

In [1]:
```
# Method 1: By prompting the user to input the values for x and y flexibly;

"""
Created on Mon Oct 25 09:21:09 2021

@author: OPEYEMI IBRAHIM
"""
# Python 3 Program to Interpolate Using Newton Forward Difference Interpolation

# Method 1: By prompting the user to input the values for x and y flexibly;

# Importing NumPy Library
import numpy as np

def u_cal(u, n):

    temp = u;
    for i in range(1, n):
        temp = temp * (u - i);
    return temp;

# calculating factorial of given number n
def fact(n):
    f = 1;
    for i in range(2, n + 1):
        f *= i;
    return f;

# Reading number of unknowns
n = int(input('\nEnter number of data points: '))

# Making numpy array of n & n x n size and initializing
# to zero for storing x and y value along with differences of y
x = np.zeros((n))
y = np.zeros((n,n))


# Reading data points
print('\nEnter data for x and y: ')
for i in range(n):
    x[i] = float(input( 'x['+str(i)+']='))
    y[i][0] = float(input( 'y['+str(i)+']='))

# Generating forward difference table
for i in range(1,n):
    for j in range(0,n-i):
        y[j][i] = y[j+1][i-1] - y[j][i-1]


print ('\nOUTPUT RESULT:\n')
```

```
print('\nFORWARD DIFFERENCE TABLE\n');

for i in range(n):
    print('%0.2f' %(x[i]), end='')
    for j in range(1, n):
        print('\t\t%0.2f' %(y[i][j]), end='')
    print()


# Value to interpolate at
value = 1.5;

# initializing u and sum
sum = y[0][0];
u = (value - x[0]) / (x[1] - x[0]);
for i in range(1,n):
    sum = sum + (u_cal(u, i) * y[0][i]) / fact(i);

print("\nValue at", value, "is", round(sum, 6));

# print a line '-' 50 times after running codes for Method 1: By prompting the user
print('-'*50)

#--------------------------------------------------#
```

```
Enter number of data points: 8

Enter data for x and y:
x[0]=1
y[0]=1
x[1]=2
y[1]=8
x[2]=3
y[2]=27
x[3]=4
y[3]=64
x[4]=5
y[4]=125
x[5]=6
y[5]=216
x[6]=7
y[6]=343
x[7]=8
y[7]=512

OUTPUT RESULT:


FORWARD DIFFERENCE TABLE

1.00            7.00            12.00           6.00            0.00            0.00
0.00            0.00
2.00            19.00           18.00           6.00            0.00            0.00
0.00            0.00
3.00            37.00           24.00           6.00            0.00            0.00
0.00            0.00
4.00            61.00           30.00           6.00            0.00            0.00
0.00            0.00
5.00            91.00           36.00           6.00            0.00            0.00
0.00            0.00
6.00            127.00          42.00           0.00            0.00            0.00
0.00            0.00
7.00            169.00          0.00            0.00            0.00            0.00
0.00            0.00
8.00            0.00            0.00            0.00            0.00            0.00
0.00            0.00
```

```
Value at 1.5 is 3.375
--------------------------------------------------
```

Method 2: By inputing our values for x and y rigidly into our codes:

In [2]:

```python
# Method 2: By inputing our values for x and y rigidly into our codes:

"""
Created on Mon Oct 25 09:21:09 2021

@author: OPEYEMI IBRAHIM
"""
# Python 3 Program to Interpolate Using Newton Forward Difference Interpolation

# Method 2: By inputing our values for x and y rigidly into our codes:

# calculating u mentioned in the formula for Newton forward difference method

def u_cal(u, n):

    temp = u;
    for i in range(1, n):
        temp = temp * (u - i);
    return temp;

# calculating factorial of given number n
def fact(n):
    f = 1;
    for i in range(2, n + 1):
        f *= i;
    return f;


# Number of values given
n = 8;
# Inputing our values for x rigidly into our codes:
x = [1, 2, 3, 4, 5, 6, 7, 8];

# y[][] is used for difference table
# with y[][0] used for input
# Inputing our values for y rigidly into our codes:

y = [[0 for i in range(n)]
        for j in range(n)];
y[0][0] = 1;
y[1][0] = 8;
y[2][0] = 27;
y[3][0] = 64;
y[4][0] = 125;
y[5][0] = 216;
y[6][0] = 343;
y[7][0] = 512;

# Calculating the forward difference table
for i in range(1, n):
    for j in range(n - i):
        y[j][i] = y[j + 1][i - 1] - y[j][i - 1];


print ('\nOUTPUT RESULT:\n')

print('\nFORWARD DIFFERENCE TABLE\n');
```

```
# Displaying the forward difference table
for i in range(n):
    print(x[i], end = "\t");
    for j in range(n - i):
        print(y[i][j], end = "\t");
    print("");

# Value to interpolate at
value = 1.5;

# initializing u and sum
sum = y[0][0];
u = (value - x[0]) / (x[1] - x[0]);
for i in range(1,n):
    sum = sum + (u_cal(u, i) * y[0][i]) / fact(i);

print("\nValue at", value, "is", round(sum, 6));

# print a line '-' 50 times after running codes for Method 2: By inputing our values
print('-'*50)

#-------------------------------------------------#
```

```
OUTPUT RESULT:


FORWARD DIFFERENCE TABLE

1	1	7	12	6	0	0	0	0
2	8	19	18	6	0	0	0
3	27	37	24	6	0	0
4	64	61	30	6	0
5	125	91	36	6
6	216	127	42
7	343	169
8	512

Value at 1.5 is 3.375
--------------------------------------------------
```