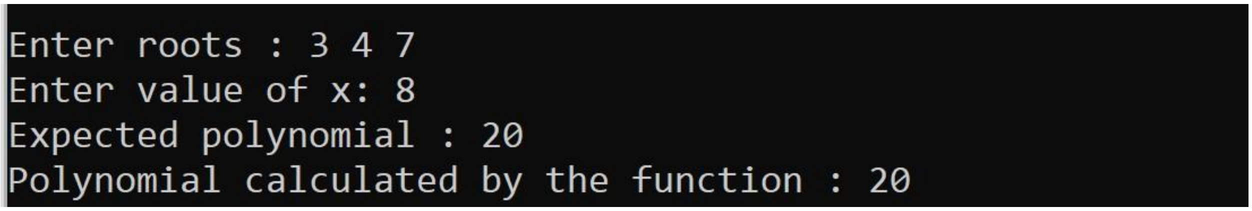
Sukti Tiwari
Dr. Su Yan
3/12/2022

HOMEWORK 3

3.10

```
import numpy as np
def calc_poly(x,roots):
    """
    Computes polynomial
    """
    p = 1
    for i in range(len(roots)):
        p *= (x - roots[i])
    return p
def test_poly(x,roots):
    """
    Test function for poly
    """
    p=1
    assert calc_poly(x, roots) == np.prod([p*(x-roots[i]) for i in range(len(roots))]), "Calculation is wrong"

def main():
    roots = [1,2,5]
    x = 3
    p=1
    exp = np.prod([p*(x-roots[i]) for i in range(len(roots))])
    print("Expected polynomial :",exp)
    print("Polynomial calculated by the function :",calc_poly(x,roots))

    test_poly(x,roots)

main()
import numpy as np
def calc_poly(x,roots):
    """
    Computes polynomial
 """
    p = 1
    for i in range(len(roots)):
        p *= (x - roots[i])
    return p
def test_poly(x,roots):
    """
    Test function for poly
    """
    p=1
    assert calc_poly(x, roots) == np.prod([p*(x-roots[i]) for i in range(len(roots))]), "Calculation is wrong"

def main():
    roots = list(map(int,input("\nEnter roots : ").strip().split()))
    x = int(input("Enter value of x: "))
    p=1
    exp = np.prod([p*(x-roots[i]) for i in range(len(roots))])
    print("Expected polynomial :",exp)
    print("Polynomial calculated by the function :",calc_poly(x,roots))

    test_poly(x,roots)

main()
```

Output

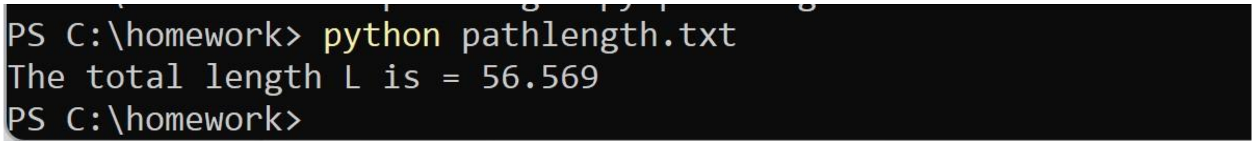


```
Enter roots : 3 4 7
Enter value of x: 8
Expected polynomial : 20
Polynomial calculated by the function : 20
```

3.17

```
import math

#definition of the pathlength() function...
def pathlength(x,y):

    #declare the L and set it to 0...
    L = 0

    #find the length of x and y...
    lenX = len(x)
    lenY = len(y)

    #use the for loop to find the expression...
    for i in range(1,lenX):
        L += math.sqrt( (x[i] - x[i-1]) ** 2 + (y[i] - y[i-1]) ** 2 )

    #handle the last pair of distance...
    L += math.sqrt( (x[lenX - 1] - x[lenX - 2]) ** 2 + (y[lenY - 1] - y[lenY - 2]) ** 2 )
    #return the L...
    return L

#definition of the test_pathlength() to test the above function...
def test_pathlength():

    #set some dummy data in the lists, x and y...
    x = [10,20,30,40]
    y = [10,20,30,40]

    #call the pathlength() function and store the returned value...
    result = pathlength(x,y)

    #print the final result...
    print("The total length L is = {:.3f}".format(result))

test_pathlength()
```

output:

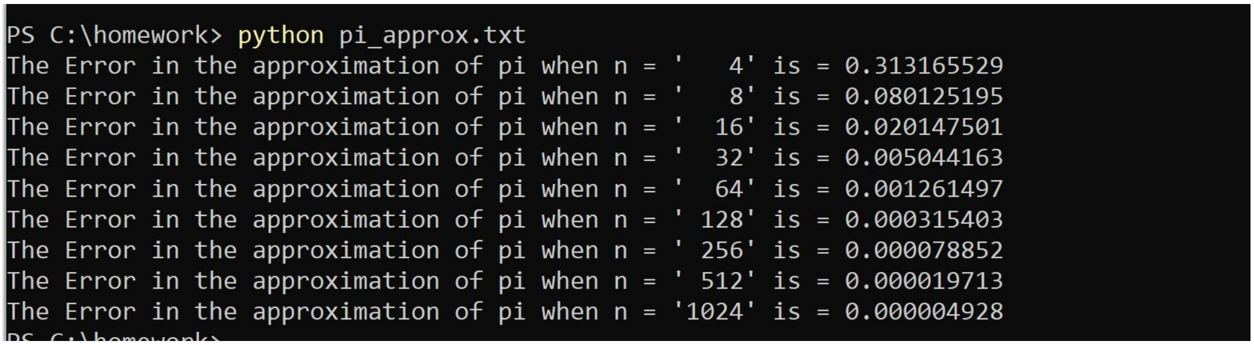


```
PS C:\homework> python pathlength.txt
The total length L is = 56.569
PS C:\homework>
```

3.18

```
import math

def pathlength(x,y):

    #declare the L and set it to 0...
    L = 0

    #find the length of x and y...
    lenX = len(x)
    lenY = len(y)

    #use the for loop to find the expression...
    for i in range(1,lenX):
        L += math.sqrt( (x[i] - x[i-1]) ** 2 + (y[i] - y[i-1]) ** 2 )

    #handle the last pair of distance...
    L += math.sqrt( (x[lenX - 1] - x[lenX - 2]) ** 2 + (y[lenY - 1] - y[lenY - 2]) ** 2 )

    #return the L...
    return L

def test_pathlength():


    for k in range(2,11):

        #declare empty x and y lists...
        x = []
        y = []

        #find the value of n...
        n = 2**k

        #generate the points as per the given formula...
        for i in range(n):
            x.append( (1/2) * math.cos( 2 * math.pi * i / n))
            y.append( (1/2) * math.sin( 2 * math.pi * i / n))


        result = pathlength(x,y)

        #print the final result...
        print("The Error in the approximation of pi when n = '{:4d}' is = {:.9f}".format(n, math.pi -
result))

test_pathlength()
```

Output



```
PS C:\homework> python pi_approx.txt
The Error in the approximation of pi when n = '   4' is = 0.313165529
The Error in the approximation of pi when n = '   8' is = 0.080125195
The Error in the approximation of pi when n = '  16' is = 0.020147501
The Error in the approximation of pi when n = '  32' is = 0.005044163
The Error in the approximation of pi when n = '  64' is = 0.001261497
The Error in the approximation of pi when n = ' 128' is = 0.000315403
The Error in the approximation of pi when n = ' 256' is = 0.000078852
The Error in the approximation of pi when n = ' 512' is = 0.000019713
The Error in the approximation of pi when n = '1024' is = 0.000004928
PS C:\homework>
```