PYTHON CODE:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

import math //Question 1

factorial = input("Please enter an integer for n!") #enter number for n!

factorial = int(factorial)

factorialresult = factorial

stirlingsNum = ((2 \* math.pi \* factorial)\*\*(1/2)) \* ((factorial / math.e)\*\*(factorial))

print("Stirlings Number: ", stirlingsNum) #result of stirlings number: ((2pi \* n)^(1/2)) \* ((n/e)^n)

while (factorial != 1):

factorialresult = (factorial-1) \* factorialresult

factorial = factorial - 1

print("N!: ", factorialresult) #result of n!

abserror = factorialresult - stirlingsNum

print(abserror) #result of absolute error

rerror = abserror / stirlingsNum

print(rerror) #result of relative error

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

a). BISECTION METHOD: //Question 2.

# a = 0 | b = 1 | c = ½ | d = 1 - ½ | and f(c) tells us if c becomes a or b

import math

intervalx = input("Please enter a starting point for x:")

intervalx = float(intervalx)

print("a: ", intervalx)

intervaly = input("Please enter a starting point for y:")

intervaly = float(intervaly)

print("b: ", intervaly)

functionx = math.e\*\*(intervalx) - math.sin(intervalx) - 2

functiony = math.e\*\*(intervaly) - math.sin(intervaly) - 2

while(functiony \* functionx >= 0):

print("Requirements of (f(a) \* f(b) < 0) is not met. Please reenter values:")

intervalx = input("Please enter a starting point for x:")

intervalx = float(intervalx)

print("a: ", intervalx)

intervaly = input("Please enter a starting point for y:")

intervaly = float(intervaly)

print("b: ", intervaly)

if (intervalx > intervaly):

tempx = intervaly

intervaly = intervalx

intervalx = tempx

mean = (intervaly + intervalx)/2

print("c: ", mean)

pointchange = intervaly - mean

print("(b-c): ", pointchange)

error = 10\*\*(-10)

functionC = math.e\*\*(mean) - math.sin(mean) - 2

print("f(c): ", functionC)

while(pointchange > error):

if (functionC > 0):

intervaly = mean

else:

intervalx = mean

print("a: ", intervalx)

print("b: ", intervaly)

mean = (intervaly + intervalx)/2

print("c: ", mean)

pointchange = intervaly - mean

print("(b-c): ", pointchange)

functionC = math.e\*\*(mean) - math.sin(mean) - 2

print("f(c): ", functionC)

print("Approximate root is: ", mean)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

b). BISECTION METHOD: //Question 2.

# same code layout interval is now [2, 3.5]

import math

intervalx = input("Please enter starting value for x:")

intervalx = float(intervalx)

print("a: ", intervalx)

intervaly = input("Please enter starting value for y:")

intervaly = float(intervaly)

print("b: ", intervaly)

functionx = (intervalx)\*\*(2) - 4\*(intervalx) + 4 - math.log(intervalx)

functiony = (intervaly)\*\*(2) - 4\*(intervaly) + 4 - math.log(intervaly)

while(functiony \* functionx >= 0):

print("Requirements of (f(a) \* f(b) < 0) is not met. Please reenter values:")

intervalx = input("Please enter starting value for x:")

intervalx = float(intervalx)

print("a: ", intervalx)

intervaly = input("Please enter starting value for y:")

intervaly = float(intervaly)

print("b: ", intervaly)

if (intervalx > intervaly):

tempx = intervaly

intervaly = intervalx

intervalx = tempx

mean = (intervaly + intervalx)/2

print("c: ", mean)

pointchange = intervaly - mean

print("(b-c): ", pointchange)

error = 10\*\*(-10)

functionC = mean\*\*(2) - 4\*mean + 4 - math.log(mean)

print("f(c): ", functionC)

while(pointchange > error):

if (functionC > 0):

intervaly = mean

else:

intervalx = mean

print("a: ", intervalx)

print("b: ", intervaly)

mean = (intervaly + intervalx)/2

print("c: ", mean)

pointchange = intervaly - mean

print("(b-c): ", pointchange)

functionC = mean\*\*(2)-4\*mean + 4 - math.log(mean)

print("f(c): ", functionC)

print("Approximate root is: ", mean) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

a). NEWTONS METHOD:

import math

x = float(input("Please enter value for Newtons Method:"))

func = math.e\*\*(x) - math.sin(x) - 2

derivative = math.e\*\*(x) - math.cos(x)

while(derivative == 0):

print("Derivative is 0, cannot plug into function, please reenter value:")

x = float(input("Please enter value for Newtons Method:"))

derivative = math.e\*\*(x) - math.cos(x)

funcx = x - (func / derivative)

print("x 0 :", funcx)

difference = 10

error = 10\*\*(-10)

counter = 1

while (difference > error):

x = funcx

func = math.e\*\*(x) - math.sin(x) - 2

derivative = math.e\*\*(x) - math.cos(x)

functemp = funcx

funcx = x - (func / derivative)

print("x", counter, ":", funcx)

difference = functemp - funcx

counter = counter + 1

print(funcx)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

b). NEWTONS METHOD:

import math

x = float(input("Please enter value for Newtons Method:"))

func = x\*\*(2) - 4\*x + 4 - math.log(x)

derivative = 2\*(x) - 4 - (1/x)

while(derivative == 0):

print("Derivative is 0, cannot plug into function, please reenter value:")

x = float(input("Please enter value for Newtons Method:"))

derivative = 2\*(x) - 4 - (1/x)

funcx = x - (func / derivative)

print("x 0 :", funcx)

difference = 10

error = 10\*\*(-10)

counter = 1

while (difference > error):

x = funcx

func = x\*\*(2) - 4\*x + 4 - math.log(x)

derivative = 2\*(x) - 4 - (1/x)

functemp = funcx

funcx = x - (func / derivative)

print("x", counter, ":", funcx)

difference = functemp - funcx

counter = counter + 1

print(funcx)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

a). SECANTS METHOD:

import math

x0 = float(input("Please enter x0 for Secants Method:"))

x1 = float(input("Please enter x1 for Secants Method:"))

func1 = math.e\*\*(x1) - math.sin(x1) - 2

func0 = math.e\*\*(x0) - math.sin(x0) - 2

while(func1 - func0 == 0):

print("f(x1) - f(x0) = 0, cannot plug into function, please reenter values:")

x0 = float(input("Please enter x0 for Secants Method:"))

x1 = float(input("Please enter x1 for Secants Method:"))

xn = 0

xnplusone = x1 - (func1 \* (x1 - x0)) / (func1 - func0)

print("x 0 :", xnplusone)

difference = xnplusone - xn

error = 10\*\*(-10)

counter = 1

while (difference > error):

x0 = x1

x1 = xnplusone

func1 = math.e\*\*(x1) - math.sin(x1) - 2

func0 = math.e\*\*(x0) - math.sin(x0) - 2

xn = xnplusone

xnplusone = x1 - (func1 \* (x1 - x0)) / (func1 - func0)

print("x", counter, ":", xnplusone)

difference = abs(xnplusone - xn)

counter = counter + 1

print(xnplusone)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

b). SECANTS METHOD:

import math

x0 = float(input("Please enter x0 for Secants Method:"))

x1 = float(input("Please enter x1 for Secants Method:"))

func1 = x1\*\*(2) - 4\*x1 + 4 - math.log(x1)

func0 = x0\*\*(2) - 4\*x0 + 4 - math.log(x0)

while(func1 - func0 == 0):

print("f(x1) - f(x0) = 0, cannot plug into function, please reenter values:")

x0 = float(input("Please enter x0 for Secants Method:"))

x1 = float(input("Please enter x1 for Secants Method:"))

xn = 0

xnplusone = x1 - (func1 \* (x1 - x0)) / (func1 - func0)

print("x 0 :", xnplusone)

difference = xnplusone - xn

error = 10\*\*(-10)

counter = 1

while (difference > error):

x0 = x1

x1 = xnplusone

func1 = x1\*\*(2) - 4\*x1 + 4 - math.log(x1)

func0 = x0\*\*(2) - 4\*x0 + 4 - math.log(x0)

xn = xnplusone

xnplusone = x1 - (func1 \* (x1 - x0)) / (func1 - func0)

print("x", counter, ":", xnplusone)

difference = abs(xnplusone - xn)

counter = counter + 1

print(xnplusone)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_