

**Group Project**

Prepared by

**Name: Humaira Anjum**

**Name: MD. Jubayer Hossain Abir**

**ID: 2017-1-60-076**

**ID: 2017-1-60-084**

**Course Title: Numerical Methods**

**Course Code: CSE-225**

**Section: 1**

Prepared for

**Dr. Mohammad Salah Uddin**

**Assistant Professor**

**Dept. of Computer Science and Engineering**

**Date of submission: 05/08/2018**

**Source Code for Bascule Bridge Problem**

import math

def g(x):

return (-0.88318\*pow(10,-2))/(-0.50598\*pow(10,-10)\*pow(x,2)+0.38292\*pow(10,-7)\*x+0.74363\*pow(10,-4))

def iteration(xi):

n=0

while(n<5):

xi\_plus=g(xi)

n+=1

xi\_minus=xi

xi=xi\_plus

return xi

def calculate(n,T,alpha,delD):

sum\_Ti=0.0

sum\_Ti\_square=0.0

sum\_Ti\_cube=0.0

sum\_Ti\_four=0.0

sum\_ai=0.0

sum\_aiTi=0.0

sum\_ai2=0.0

for i in range(0,7):

sum\_Ti=sum\_Ti+T[i]

sum\_Ti\_square=sum\_Ti\_square+pow(T[i],2)

sum\_Ti\_cube=sum\_Ti\_cube+pow(T[i],3)

sum\_Ti\_four=sum\_Ti\_four+pow(T[i],4)

sum\_ai=sum\_ai+alpha[i]

sum\_aiTi=sum\_aiTi+(alpha[i]\*T[i])

sum\_ai2=sum\_ai2+(alpha[i]\*(T[i]\*T[i]))

a=[0.0]\*3

for i in range(0,3):

a[i]=[0.0]\*3

b=[0.0]\*3

p=3

for i in range(0,p):

for j in range(0,p):

if(i+j==0):

a[i][j]=n

elif(i+j==1):

a[i][j]=sum\_Ti

elif(i+j==2):

a[i][j]=sum\_Ti\_square

elif(i+j==3):

a[i][j]=sum\_Ti\_cube

elif(i+j==4):

a[i][j]=sum\_Ti\_four

for i in range(0,p):

if(i==0):

b[i]=sum\_ai

elif(i==1):

b[i]=sum\_aiTi

elif(i==2):

b[i]=sum\_ai2

for i in range(0,p-1):

for j in range(i+1,p):

factor=a[j][i]/a[i][i]

for k in range(i+1,p):

a[j][k]=a[j][k]-factor\*a[i][k]

b[j]=b[j]-factor\*b[i]

x=[0.0]\*3

x[p-1]=b[p-1]/a[p-1][p-1]

i=p-2

while(i>=0):

value=b[i]

for j in range(i+1,p):

value=value-a[i][j]\*x[j]

x[i]=value/a[i][i]

i=i-1

print "Vector of Variable"

for i in range(0,p):

print x[i]

print "\n"

print "Therefore second order polinomial is:\n"

print "-1.2278\*10^-11\*T^2+6.1946\*10^-9\*T+6.0150\*10^-6\n"

print "Since the desired contraction is at least ",delD," and it's going to be needed in the next equation"

print "\nSo after itegrating we can find the equation where upper limit is Tf which we need to find and the lower limit is 80\n"

print "The equation is:\n"

print "-0.50598\*10^-10\*Tf^3+0.38292\*10^-7\*Tf^2+0.74363\*10^-4\*Tf+0.88318\*10^-2=0\n"

print "Now in order to find the root simple fixed iteration is being used and the found root will be the temperature to cool the trunion\n"

Tf=iteration(0)

print Tf, "temperature is needed to cool the trunion in order to get the desired contraction\n"

print "-108 degree farenheit is not going to work to contract the trunion\n"

def main():

D=12.363

print "Trunion Outside Diameter,D = ",D

print "\nhub inner Diameter = 12.358\n"

print "Diametric Clearance = 0.01\n"

delD=D-12.358

delD=delD+0.01

print "So, the desired contraction is:",delD

print "\nHere ,alpha = coefficient of thermal expansion coefficient at room temperature\n"

print "Tf = Temperature of fluid needed\n"

print "Troom = room temperature\n"

Troom=80.0

n=7.0

T=[80.0,40.0,-40.0,-120.0,-200.0,-280.0,-340.0]

alpha=[0.00000647,0.00000624,0.00000572,0.00000509,0.00000430,0.00000333,0.00000245]

calculate(n,T,alpha,delD)

if \_\_name\_\_=="\_\_main\_\_":

main()

**Algorithm for Bascule Bridge problem**

Step 1 -> Start

Step 2 -> D = 12.363

Step 3-> delD = D – 12.358 + 0.01

Step 4-.> n=7.0

T=[80.0,40.0,-40.0,-120.0,-200.0,-280.0,-340.0],

alpha=[0.00000647,0.00000624,0.00000572,0.00000509,0.00000430,0.00000333

,0.00000245]

Step 5-> calculate(n,T,alpha,delD)

Step 6-> i = 0,

sum\_Ti = 0.0

sum\_Ti\_square = 0.0

sum\_Ti\_cube = 0.0

sum\_Ti\_four = 0.0

sum\_ai = 0.0

sum\_aiTi = 0.0

sum\_ai2 = 0.0

Step 7-> sum\_Ti = sum\_Ti + T[i]

sum\_Ti\_square = sum\_Ti\_square + pow(T[i],2)

sum\_Ti\_cube = sum\_Ti\_cube + pow(T[i],3)

sum\_Ti\_four = sum\_Ti\_four + pow(T[i],4)

sum\_ai = sum\_ai + alpha[i]

sum\_aiTi = sum\_aiTi + (alpha[i] \* T[i])

sum\_ai2 = sum\_ai2 + (alpha[i] \* (T[i] \* T[i]))

Step 8-> i = i+1

Step 9-> if i < 7 then go to step 7 else go to step 10

Step 10-> i = 0

p = 3

Step 11-> j = 0

Step 12-> if i + j = 0 then a[i][j] = n

elif i + j = 1 then a[i][j] = sum\_Ti

elif i + j = 2 then a[i][j] = sum\_Ti\_square

elif i + j = 3 then a[i][j] = sum\_Ti\_cube

elif i +j = 4 then a[i][j] = sum\_Ti\_four

Step 13-> j = j + 1

Step 14-> if j < p then go to step 12 then go to step 15

Step 15-> i = i +1

Step 16-> if i < p then go to step 12 else go to step 17

Step 17-> i = 0

Step 18-> if i = 0 then b[i] = sum\_ai

elif i = 1 then b[i] = sum\_aiTi

elif i = 2 then b[i] = sum\_ai2

Step 19-> i = i + 1

Step 20-> if i < p then go to step 18 else go to step 21

Step 21-> i = 0

Step 22-> j = i + 1

Step 23-> factor = a[j][i] / a[i][j]

Step 24->k = i + 1

Step 25-> a[j][k] = a[j][k] – factor \* a[i][k]

Step 26-> k = k + 1

Step 27-> if k < p then go to step 25 else go to 28

Step 28-> b[j] = b[j] – factor \* b[i]

Step 29-> j = j + 1

Step 30-> if j < p then go to step 23 else i = i + 1 and go to step 31

Step 31-> if i < p – 1 then go to step 22 else go to 32

Step 32-> x[p-1] = b [p-1] / a[p-1][p-1]

i = p – 2

Step 33-> value = b[i]

Step 34-> j = i + 1

Step 35-> value = value – a[i][j] \* x[j]

Step 36-> if j < p then go to step 35 else go to step 36

Step 37-> x[i] = value/a[i][i]

Step 38-> i = i – 1

Step 39-> if i >= 0 then go to step 33 else go to step 40

Step 40-> i = 0

Step 41-> print x[i]

Step 42-> i = i + 1

Step 43-> if i < p then go to step 41 else go to step 44

Step 44-> Tf = iteration(0)

Step 45-> print Tf

Step 46-> End