id - 202218061

ASSIGNMENT - 05

NUMERICAL METHOD

Write a program for Golden Search method and Fibonacci search method to find the maximum of the function given below within the interval $x_{l1} = 0$ and $x_{u1} = 4$.

$$f(x) = 2 \sin x - \frac{x^2}{10}$$

with $\varepsilon = 0.05$.

Note:- You need to carry out the n iteration such that the length of the last iteration i,e. $I_n = x_{un} - x_{ln} \leq \varepsilon$

```
1 import math
```

Golden Rule Search Method

```
 1 def f_x(x):
 2   return ((2*(math.sin(x))) - ((x**2)/10))
 3   # return x**2
 4
 5 def golden_search_method(x_l1,x_u1,e):
 6
 7   x_l,x_u = x_l1,x_u1
 8   x_p = x_u1 - 0.618*(x_u1 - x_l1)
 9   x_q = x_l1 + 0.618*(x_u1 - x_l1)
10
11   E_p = f_x(x_p)
12   E_q = f_x(x_q)
13   while x_u - x_l > e:
14     if E_p <= E_q:
15       x_l = x_l
16       x_u = x_q
17       x_q = x_p
18       x_p = x_u - 0.618*(x_u - x_l)
19       E_p = f_x(x_p)
20       E_q = f_x(x_q)
```

```
    else:
      x_l = x_p
      x_u = x_u
      x_p = x_q
      x_q = x_l + 0.618*(x_u - x_l)
      E_p = f_x(x_p)
      E_q = f_x(x_q)

    # print(f'''
    # x_l : {x_l}
    # x_u : {x_u}
    # x_p : {x_p}
    # x_q : {x_q}
    # E_p : {E_p}
    # E_q : {E_q}''')

  return x_l,x_u

```

```
x_l1 = 0
x_u1 = 4
e = 0.05
x_l,x_u = golden_search_method(x_l1,x_u1,e)
print(f'''x_l    : {x_l}
f(x_l) : {f_x(x_l)}

x_u    : {x_u}
f(x_u) : {f_x(x_u)}''')
```

```
x_l    : 3.967463052550272
f(x_l) : -3.044352545744955

x_u    : 4
f(x_u) : -3.1136049906158565
```

Fibonacci Search Method

```
def fibo(n):
  if n == 0 or n == 1:
    return 1
  else:
    return fibo(n-1)+fibo(n-2)
```

```
def fibonacci_search_method(x_l1,x_u1,e,n):
  k = 1
  x_l,x_u = x_l1,x_u1
  x_p = x_u1 - (fibo(n-k)/fibo(n-k+1))*(x_u1 - x_l1)
  x_q = x_l1 + (fibo(n-k)/fibo(n-k+1))*(x_u1 - x_l1)

  E_p = f_x(x_p)
  E_q = f_x(x_q)
  k += 1
```

```
  while k <= n-2:
    if E_p <= E_q:
      x_l = x_l
      x_u = x_q
      x_q = x_p
      x_p = x_u - (fibo(n-k)/fibo(n-k+1))*(x_u - x_l)
      E_p = f_x(x_p)
      E_q = f_x(x_q)
    else:
      x_l = x_p
      x_u = x_u
      x_p = x_q
      x_q = x_l + (fibo(n-k)/fibo(n-k+1))*(x_u - x_l)
      E_p = f_x(x_p)
      E_q = f_x(x_q)
    k += 1

  # Last two iteration where we need to add 'e' in the n-2th iteration
  for i in range(2):
    if E_p >= E_q:
      x_l = x_l
      x_u = x_q
      x_q = x_p
      x_p = x_u - (fibo(n-k)/fibo(n-k+1))*(x_u - x_l) - e
      E_p = f_x(x_p)
      E_q = f_x(x_q)
    else:
      x_l = x_p
      x_u = x_u
      x_p = x_q
      x_q = x_l + (fibo(n-k)/fibo(n-k+1))*(x_u - x_l) + e
      E_p = f_x(x_p)
      E_q = f_x(x_q)

  return x_l,x_u
```

```
x_l1 = 0
x_u1 = 4
e = 0.05
n = 20
fx_l,fx_u = fibonacci_search_method(x_l1,x_u1,e,n)
print(f'''x_l    : {fx_l}
f(x_l) : {f_x(fx_l)}

x_u    : {fx_u}
f(x_u) : {f_x(fx_u)}''')
```

```
x_l    : 3.998903709117486
f(x_l) : -3.1112940017325714

x_u    : 3.9992691394116573
f(x_u) : -3.1120645066718566
```

```
1
```