

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 $\epsilon = 0.05$.

Note:- You need to carry out the n iteration such that the length of the last iteration i.e.

$$I_n = x_{un} - x_{ln} \leq \epsilon$$

```
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)
```

```

21     else:
22         x_l = x_p
23         x_u = x_u
24         x_p = x_q
25         x_q = x_l + 0.618*(x_u - x_l)
26         E_p = f_x(x_p)
27         E_q = f_x(x_q)
28
29     # print(f'''
30     # x_l : {x_l}
31     # x_u : {x_u}
32     # x_p : {x_p}
33     # x_q : {x_q}
34     # E_p : {E_p}
35     # E_q : {E_q}''')
36
37     return x_l,x_u
38

```

```

1 x_l1 = 0
2 x_u1 = 4
3 e = 0.05
4 x_l,x_u = golden_search_method(x_l1,x_u1,e)
5 print(f'''x_l      : {x_l}
6 f(x_l) : {f_x(x_l)}
7
8 x_u      : {x_u}
9 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

```

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

1 def fibonacci_search_method(x_l1,x_u1,e,n):
2     k = 1
3     x_l,x_u = x_l1,x_u1
4     x_p = x_u1 - (fibo(n-k)/fibo(n-k+1))*(x_u1 - x_l1)
5     x_q = x_l1 + (fibo(n-k)/fibo(n-k+1))*(x_u1 - x_l1)
6
7     E_p = f_x(x_p)
8     E_q = f_x(x_q)
9     k += 1

```

```

10
11 while k <= n-2:
12     if E_p <= E_q:
13         x_l = x_l
14         x_u = x_q
15         x_q = x_p
16         x_p = x_u - (fibo(n-k)/fibo(n-k+1))*(x_u - x_l)
17         E_p = f_x(x_p)
18         E_q = f_x(x_q)
19     else:
20         x_l = x_p
21         x_u = x_u
22         x_p = x_q
23         x_q = x_l + (fibo(n-k)/fibo(n-k+1))*(x_u - x_l)
24         E_p = f_x(x_p)
25         E_q = f_x(x_q)
26     k += 1
27
28 # Last two iteration where we need to add 'e' in the n-2th iteration
29 for i in range(2):
30     if E_p >= E_q:
31         x_l = x_l
32         x_u = x_q
33         x_q = x_p
34         x_p = x_u - (fibo(n-k)/fibo(n-k+1))*(x_u - x_l) - e
35         E_p = f_x(x_p)
36         E_q = f_x(x_q)
37     else:
38         x_l = x_p
39         x_u = x_u
40         x_p = x_q
41         x_q = x_l + (fibo(n-k)/fibo(n-k+1))*(x_u - x_l) + e
42         E_p = f_x(x_p)
43         E_q = f_x(x_q)
44
45 return x_l, x_u
46

```

```

1 x_l1 = 0
2 x_u1 = 4
3 e = 0.05
4 n = 20
5 fx_l, fx_u = fibonacci_search_method(x_l1, x_u1, e, n)
6 print(f'''x_l      : {fx_l}
7 f(x_l) : {f_x(fx_l)}
8
9 x_u      : {fx_u}
10 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

[Colab notebook](#) [Cancel contract here](#)

×