NC Lab 4

11762 Muhammad Kashif

Secant Method

S#	Function	Intervals	Tolerance	No. of Iteration	Roots
1	x-0.8-0.2.sin(x)	0,1	0.01	3	0.964
		0,1	0.001	4	0.9643
		0,1	0.0001	4	0.9643
2	x^2-4x+4-ln(x)	0.5,1.5	0.01	4	1.41245
		0.5,1.5	0.001	5	1.41239
		0.5,1.5	0.0001	5	1.41239

Newton Raphson Method

S#	Function	Intervals	Tolerance	No. of Iteration	Roots
1	Cos(x)-1.3x	1	0.01	4	0.62418
		1	0.001	4	0.62418
		1	0.0001	5	0.624184
2	x.Cos(x)-2.x^2+3.x-1	1	0.01	5	1.25662
		1	0.001	5	1.25662
		1	0.0001	6	1.256623

Task 1 Secant Method

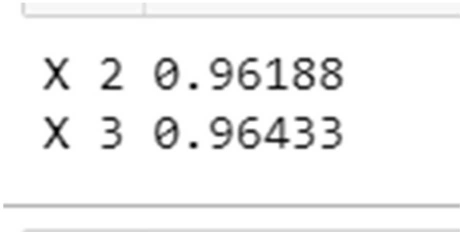
Question x-0.8-0.2*m.sin(x)

Code

```
import math as m
import symbol as s
def f(x):
    return x-0.8-0.2*m.sin(x)

def Secant(x0,x1):
    for i in range(15):
        if (f(x1)-f(x0)) != 0:
            x = x1 - ((f(x1)*(x1-x0))/(f(x1)-f(x0)))
            x= round(x,5)
            print("X",i+2,x)
            x0 = x1
            x1 = x
            tol = abs((x1-x0)/x1)
            if  tol <= 0.01:
                break
        else:
            break
Secant(0,1)
```

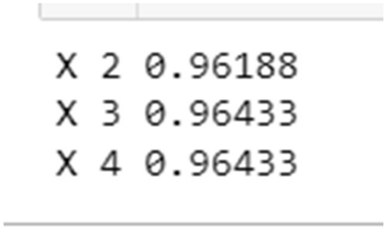
Output

Tol = 0.01



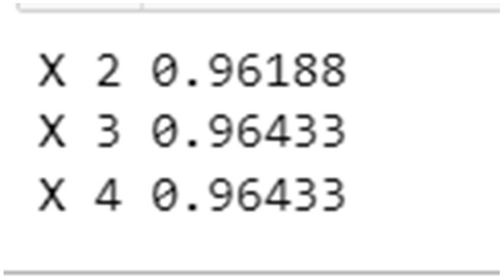
```
X 2 0.96188
X 3 0.96433
```

Tol = 0.001



```
X 2 0.96188
X 3 0.96433
X 4 0.96433
```

Tol = 0.0001



```
X 2 0.96188
X 3 0.96433
X 4 0.96433
```

Task 2

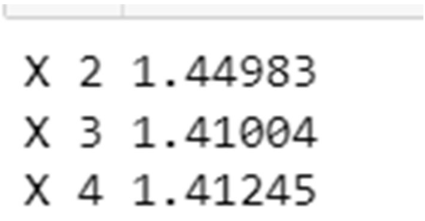
Question x**2-(4*x)+4-(m.log(x))

Code

```
import math as m
import symbol as s
def f(x):
   return x**2-(4*x)+4-(m.log(x))

def Secant(x0,x1):
   for i in range(15):
      if (f(x1)-f(x0)) != 0:
         x = x1 - ((f(x1)*(x1-x0))/(f(x1)-f(x0)))
         x= round(x,5)
         print("X",i+2,x)
         x0 = x1
         x1 = x
         tol = abs((x1-x0)/x1)
         if  tol <= 0.0001:
            break
      else:
         break
Secant(0.5,1.5)
```

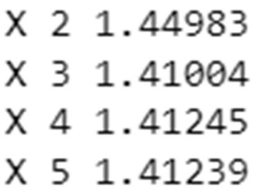
Output

Tol = 0.01



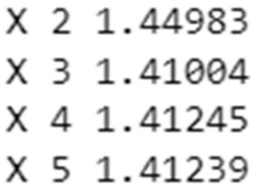
```
X 2 1.44983
X 3 1.41004
X 4 1.41245
```

Tol = 0.001



```
X 2 1.44983
X 3 1.41004
X 4 1.41245
X 5 1.41239
```

Tol = 0.0001



```
X 2 1.44983
X 3 1.41004
X 4 1.41245
X 5 1.41239
```

Task 3 Newton Raphson Method

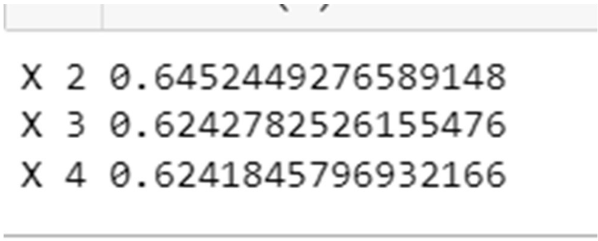
Question m.cos(x)-1.3*x

Code

```
import math as m
import symbol as s
def f(x):
    return m.cos(x)-1.3*x

def fd(x):
    return -m.sin(x)-1.3

def Newtons(x0):
    for i in range(5):
        if f(x0) != 0:
            x = x0 -(f(x0)/fd(x0))
            print("X",i+2,x)
            tol = abs((x0-x)/x0)
            if tol <= 0.0001:
                break
            x0 = x
        else:
            break

Newtons(1)
```

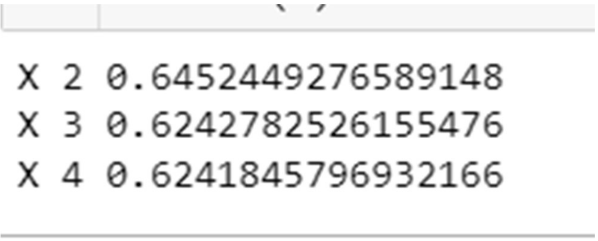
Output

Tol = 0.01



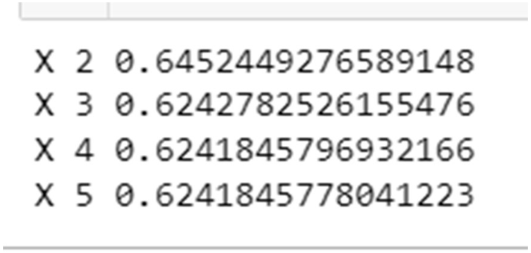
```
X 2 0.6452449276589148
X 3 0.6242782526155476
X 4 0.6241845796932166
```

Tol = 0.001



```
X 2 0.6452449276589148
X 3 0.6242782526155476
X 4 0.6241845796932166
```

Tol = 0.0001



```
X 2 0.6452449276589148
X 3 0.6242782526155476
X 4 0.6241845796932166
X 5 0.6241845778041223
```

Task 4 Newton Raphson Method

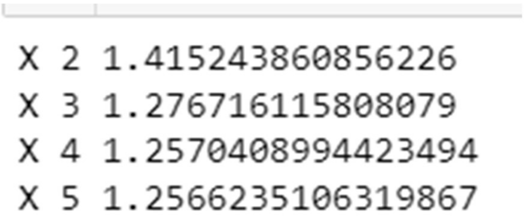
Question x*m.cos(x)-(2*x**2)+(3*x)-1

Code

```
import math as m
import symbol as s
def f(x):
    return x*m.cos(x)-(2*x**2)+(3*x)-1

def fd(x):
    return -x*m.sin(x) - (4*x) + m.cos(x) + 3

def Newtons(x0):
    for i in range(15):
        if f(x0) != 0:
            x = x0 -(f(x0)/fd(x0))
            print("X",i+2,x)
            tol = abs((x0-x))
            if tol <= 0.0001:
                break
            x0 = x
        else:
            break

Newtons(1)
```

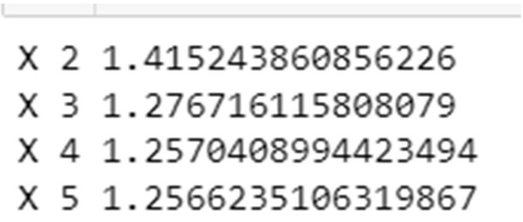
Output

Tol = 0.01



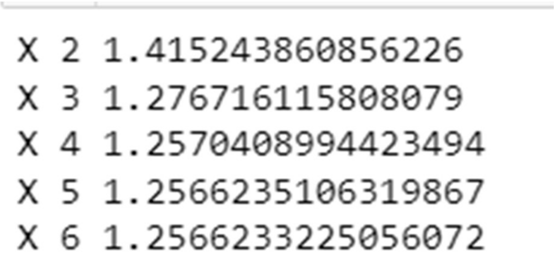
```
X 2 1.415243860856226
X 3 1.276716115808079
X 4 1.2570408994423494
X 5 1.2566235106319867
```

Tol = 0.001



```
X 2 1.415243860856226
X 3 1.276716115808079
X 4 1.2570408994423494
X 5 1.2566235106319867
```

Tol = 0.0001



```
X 2 1.415243860856226
X 3 1.276716115808079
X 4 1.2570408994423494
X 5 1.2566235106319867
X 6 1.2566233225056072
```