Name:- Soumya Ranjan Panda

Roll No:- 246PH030

SUMMARY

1(a) Done

1(b) Done

1(c) Done

1(a) Done

1(e) Done

2(a) Done

2(b) Done

2(c) Done

2(d) Not done

1(A)

$f(x) = x^3 - 2x + 2 \Rightarrow f'(x) = 3x^2 - 2$

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

$x_0 = 1$

$$x_1 = 1 - \frac{1 - \cancel{2} + \cancel{2}}{3-2} = 1 - 1 = 0$$

$$x_2 = 0 - \frac{2}{-2} = 1$$

$$x_3 = 1 - \frac{1 - \cancel{2} + \cancel{2}}{3-2} = 1 - 1 = 0$$

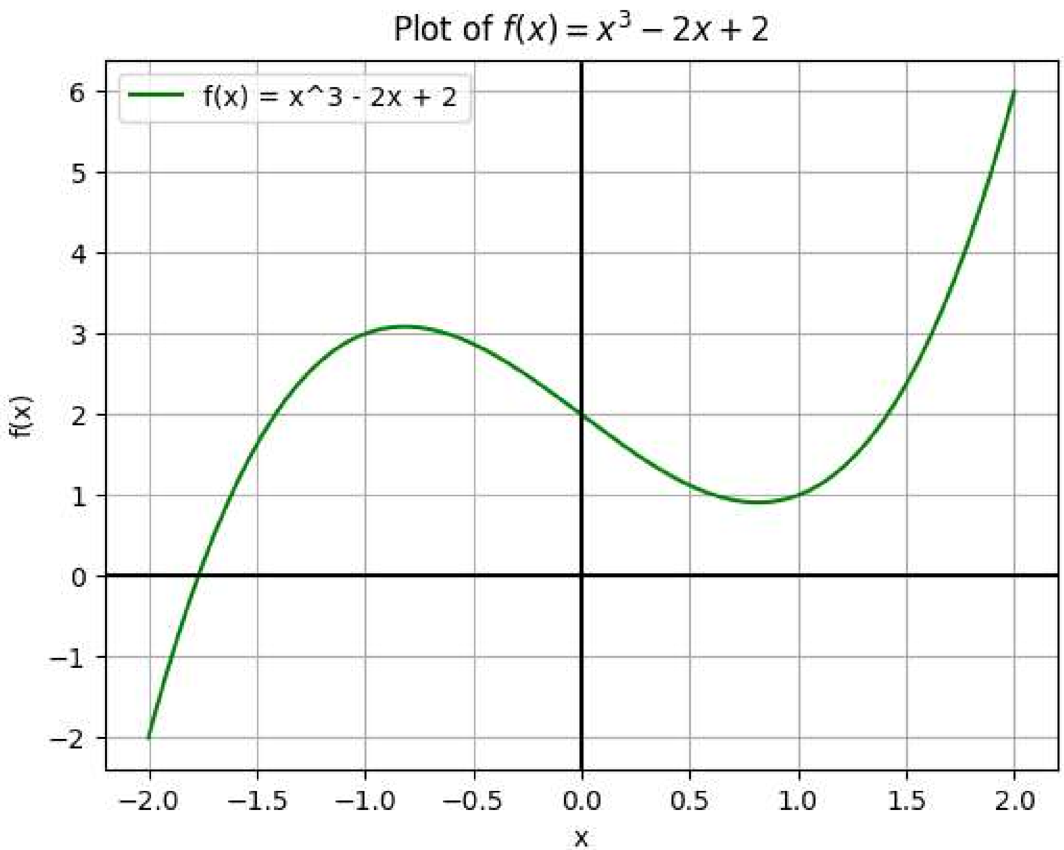
$$x_4 = 0 - \frac{2}{-2} = 1$$

$$x_5 = 1 - \frac{1 - \cancel{2} + \cancel{2}}{3-2} = 0$$

(B)

```python
import numpy as np
import matplotlib.pyplot as plt


def f(x):
    return x**3 - 2*x + 2


x = np.linspace(-2, 2, 4000)
y = f(x)


plt.plot(x, y, label="f(x) = x^3 - 2x + 2", color='g')
plt.axhline(0, color='black')
plt.axvline(0, color='black')
plt.xlabel("x")
plt.ylabel("f(x)")
```

```
plt.title("Plot of $f(x) = x^3 - 2x + 2$")

plt.legend()

plt.grid(True)

plt.show()
```



The function we plotted doesn’t intersect positive x-axis and thus has no roots on the same. It is because f’(x) is close to zero around x = 1.

(C)

From the previous graph, we noticed that the graph intersects X-axis roughly at -1.7. Now, we take the initial value as -1.7

```
import numpy as np

def f(x):

    return x**3 - 2*x + 2


def df(x):

    return 3*x**2 - 2


def newton_raphson():

    x = float(input("Enter your guess:- "))

    tol = 10**-8

    max_iter = 100
```

```
    for i in range(max_iter):
        fx = f(x)
        dfx = df(x)
        if dfx == 0:
            print("Choose a different starting point.")
            return None

        x_1 = x - (fx / dfx)

        if abs(x_1 - x) < tol:
            return x_1, i + 1
        x = x_1

    return x, max_iter

root, iteration_count = newton_raphson()
print("Final Root:- " , root )
```

```
print("Total Iterations:- " , iteration_count )
```

OUTPUT:

```
Enter your guess:- -1.7

Final Root:-  -1.7692923542386314

Total Iterations:-  4
```

(D)

```
import numpy as np

def f(x):
    return x**3 - 2*x + 2


def bisection():
    a = float(input("Enter your lower guess:- "))
    b = float(input("Enter your upper guess:- "))
    tol = 10**-8
```

```
if f(a) * f(b) >= 0:
    print("Try a different interval")
    return 0

count = 0
while (b - a) / 2 > tol:
    c = (a + b) / 2
    if f(c) == 0:
        return c, count
    elif f(a) * f(c) < 0:
        b = c
    else:
        a = c
    count += 1

return (a + b) / 2, count
```

```
root, count_num = bisection()

print("Root:- " , root)

print("Iterations:- " , count_num)
```

OUTPUT:

```
Enter your lower guess:- -2.5

Enter your upper guess:- -1.5

Root:-  -1.7692923471331596

Iterations:-  26
```

(E)

```
import numpy as np

import scipy.optimize as opt


def  f(x):
```

```
    return x**3 - 2*x + 2


def df(x):

    return 3*x**2 - 2


initial_guess = -1.7

root =opt.newton(f, initial_guess, fprime=df)


print("Root:- " , root)
```

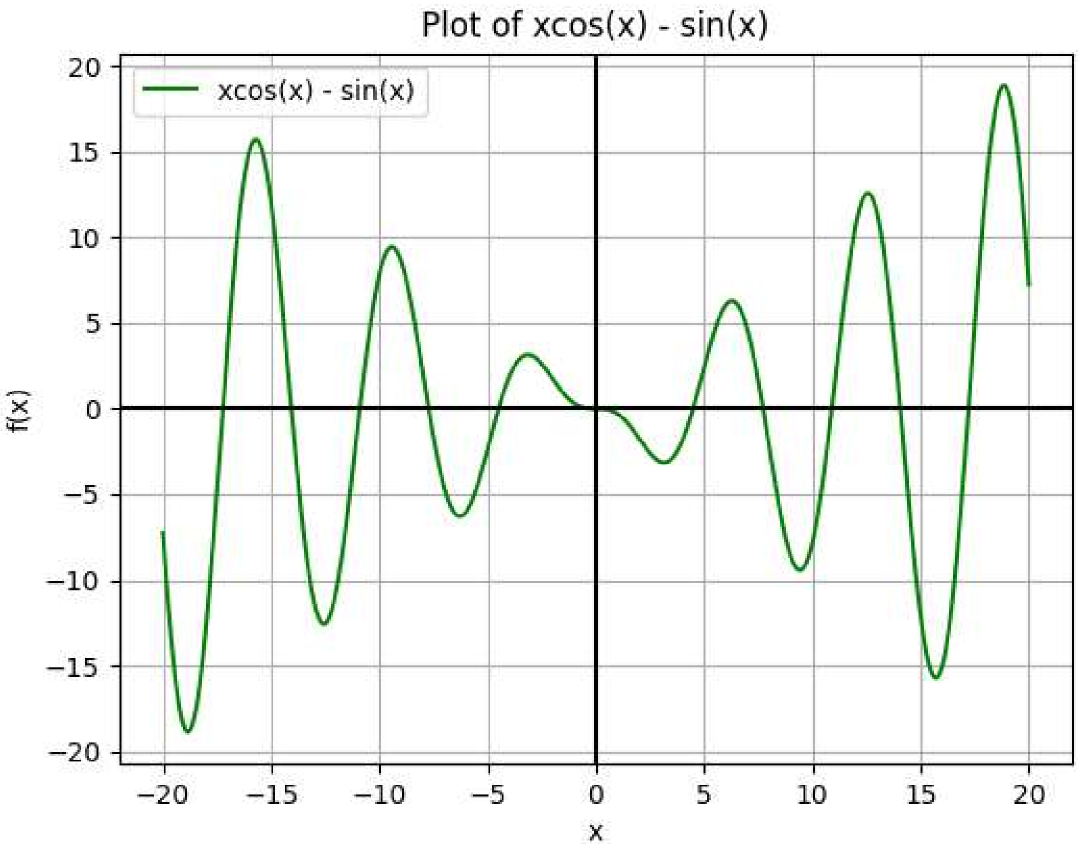
OUTPUT:

Root:- -1.7692923542386314

2(A)

```
import numpy as np

import matplotlib.pyplot as plt
```

```
# Define function
def f(x):
    return x * np.cos(x) - np.sin(x)

x = np.linspace(-20, 20, 1000)
y = f(x)

plt.plot(x, y, label="xcos(x) - sin(x)", color='g')
plt.axhline(0, color='black')
plt.axvline(0, color='black')
plt.grid()
plt.legend()
plt.title("Plot of xcos(x) - sin(x)")
plt.xlabel("x")
plt.ylabel("f(x)")
plt.show()
```



There are 11 roots between -20 to 20

Those are roughly -17, -14, -11, -8, -4.5, 0, 4.5, 7.5, 11, 14, 17

(B)

import numpy as np

```
def f(x):
    return x * np.cos(x) - np.sin(x)


def df(x):
    return -x * np.sin(x)


def newton_raphson():
    x = float(input("Enter your guess:- "))
    tol = 10**-8
    max_iter = 100
    for i in range(max_iter):
        fx = f(x)
        dfx = df(x)
        if dfx == 0:
            print("Take a new initial point…")
            return None
```

```
        x_1 = x - (fx / dfx)

        if abs(x_1 - x) < tol:
            return x_1, i + 1
        x = x_1

    return x, max_iter

root, iteration_count = newton_raphson()
print(f"Final Root:- " , root )
print("Total Iterations:- " , iteration_count )
```

OUTPUT:

Enter your guess:- 0

```
Choose a different starting point.


Enter your guess:- 2

Final Root:-  1.8749296567218637e-08

Total Iterations:-  45


Enter your guess:- 3

Final Root:-  -4.493409457909064

Total Iterations:-  5


Enter your guess:- 3.8

Final Root:-  4.493409457909064

Total Iterations:-  5


Enter your guess:- 5

Final Root:-  4.493409457909064

Total Iterations:-  4
```

(C)

```
import numpy as np

import scipy.optimize as opt


def f(x):

    return x*np.cos(x)-np.sin(x)


def df(x):

    return -x * np.sin(x)


initial_guess = float(input("Enter your initial guess:- "))

root = opt.newton(f, initial_guess, fprime=df)


print("Root:- " , root)
```

Result:-

For 0, Root:- 0.0

For 2, Root:- 2.4336055780755966e-08

For 3, Root:- -4.493409457909064

For 3.8, Root:- 4.493409457909064

For 5, Root:- 4.493409457909064

Except for initial value 2, roots for other initial values are same in both by N-R method and Scipy optimise method.