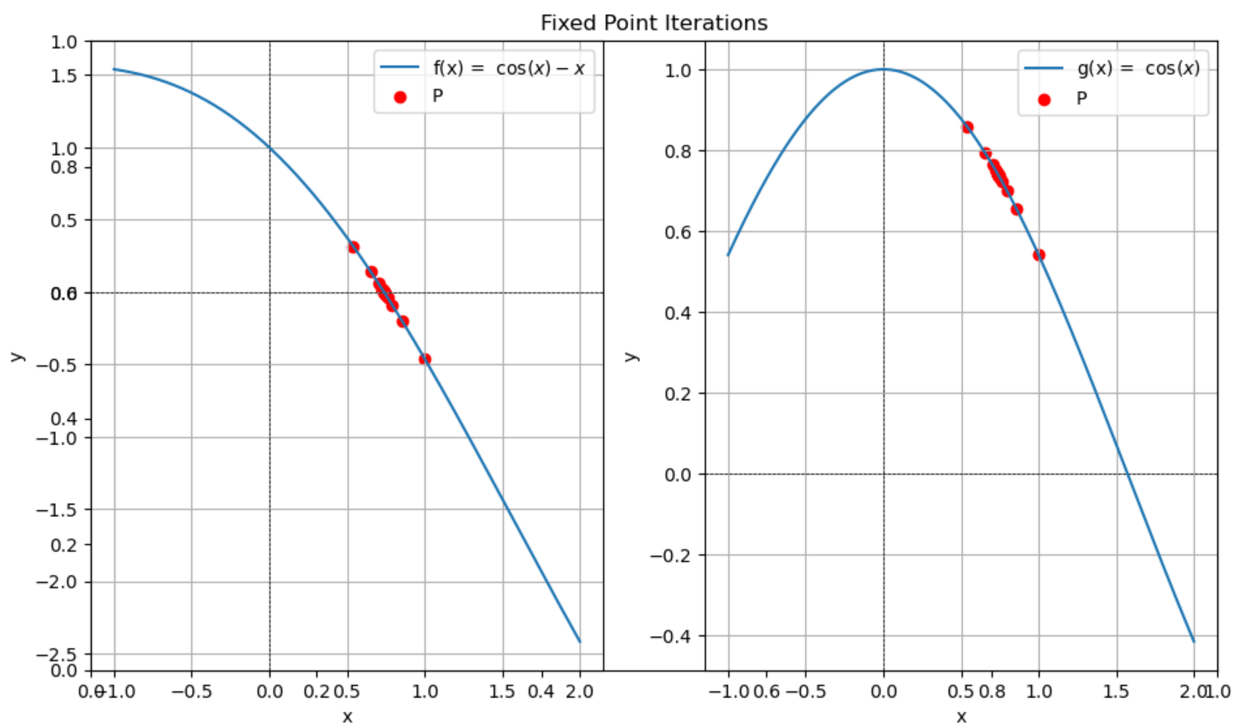
Task 1

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

# f(x) = 2*x**3 - 2*x - 5 = 0
f = lambda x: np.cos(x) - x

g = lambda x: np.cos(x)

p_arr = []

TOL = 0.0001
# max iterations
N = 100
i = 1
# guess p0
p0 = 0
# initializing p with zero
p = 0
while i <= N:
    p = g(p0)
    p_arr.append(p)
    if abs(p-p0) < TOL:
        print(f"Root: {p} with {i} iterations.")
        print(p)
        break;
    i = i + 1
    p0 = p

else:
    print(f"The method failed after {N} iterations")



x_graph = np.linspace(-1,2,400)
f_graph = f(x_graph)
g_graph = g(x_graph)
np_p = np.array(p_arr)

plt.figure(figsize=(10,6))
plt.title("Fixed Point Iterations")

plt.subplot(1,2,1)
plt.xlabel('x')
plt.ylabel('y')
plt.axhline(0, color='black', lw=0.5, ls='--')
plt.axvline(0, color='black', lw=0.5, ls='--')
plt.plot(x_graph, f_graph, label="f(x) =  $\\cos(x) - x$")
plt.scatter(np_p, f(np_p), label="P", color="red")
```

```
plt.legend()
plt.grid()

plt.subplot(1,2,2)
plt.xlabel('x')
plt.ylabel('y')
plt.axhline(0, color='black', lw=0.5, ls='--')
plt.axvline(0, color='black', lw=0.5, ls='--')
plt.plot(x_graph, g_graph, label="g(x) =  $\\cos(x)$")
plt.scatter(np_p, g(np_p), label="P", color="red")
plt.legend()
plt.grid()
plt.tight_layout()

plt.show()
```

```
Root: 0.7390547907469175 with 24 iterations.
0.7390547907469175
```



Task 2

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

# Define the function and its derivative
f = lambda x: np.exp(x) - 2
df = lambda x: np.exp(x)
```

```python
# Initial guess
x0 = 0.5
TOL = 1e-7
N = 100

# List to store x values for plotting
x_values = [x0]

# Iteration
for n in range(N):
    f_x0 = f(x0)
    df_x0 = df(x0)

    if df_x0 == 0:
        print("Derivative is zero. No solution found.")
        break

    # Update the approximation
    x1 = x0 - f_x0 / df_x0

    # Store the x value
    x_values.append(x1)

    # Check for convergence
    if abs(x1 - x0) < TOL:
        print(f"Root found: {x1}")
        break

    x0 = x1
else:
    print("Maximum iterations reached. No solution found.")

# Convert x_values to a NumPy array for plotting
np_p = np.array(x_values)

# Plotting
x_range = np.linspace(-1, 2, 400)
y_range = f(x_range)

plt.figure(figsize=(10, 6))
plt.plot(x_range, y_range, label='f(x) = e^x - 2', color='green')
plt.axhline(0, color='black', lw=0.5, ls='--')
plt.axvline(0, color='black', lw=0.5, ls='--')

# Scatter plot for the x values
plt.scatter(np_p, f(np_p), label="P", color="red")

plt.title("Newton's Method for Finding Roots")
plt.xlabel('x')
```

```
plt.ylabel('f(x)')
plt.legend()
plt.grid()
plt.ylim(-3, 3)
plt.show()

Root found: 0.6931471805599454
```