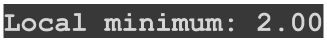
Experiment 03

Code:

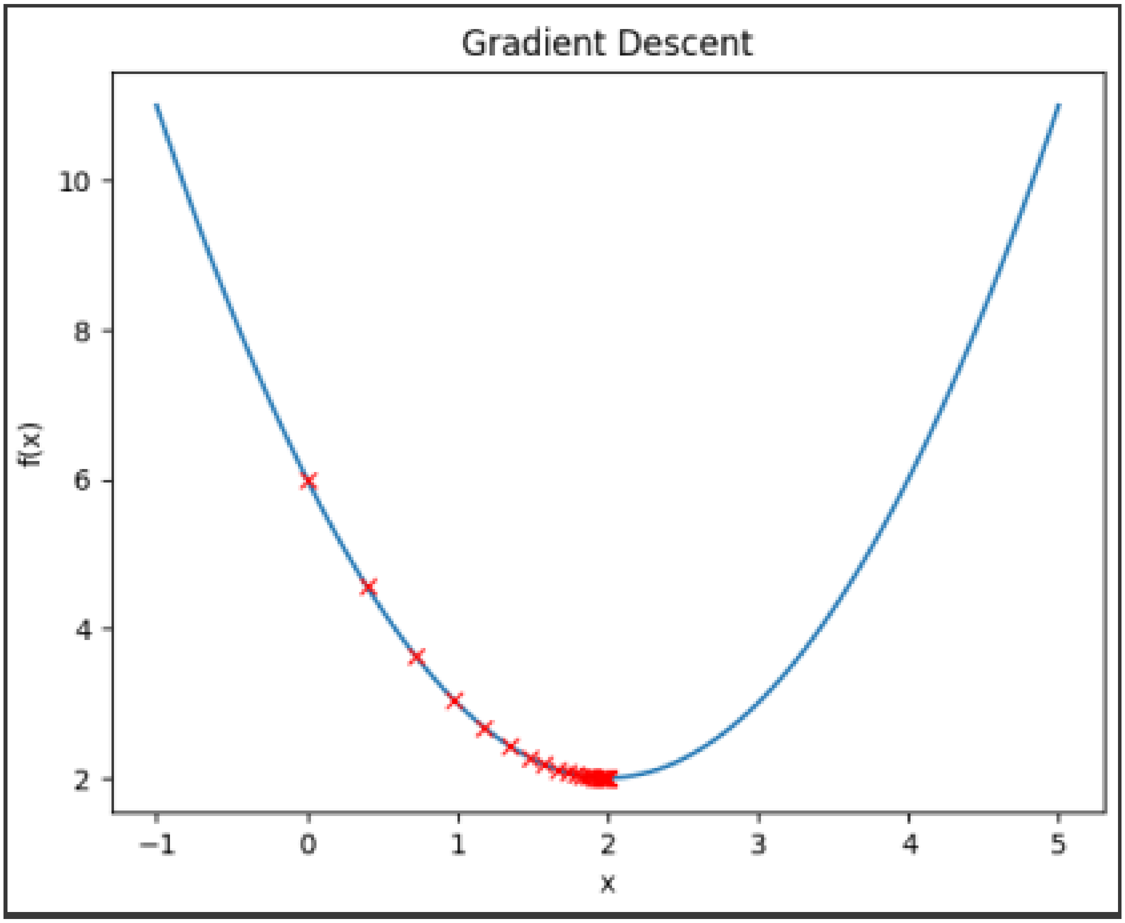
```
import numpy as np
import matplotlib.pyplot as plt

def f(x):
    return x**2 - 4*x + 6

def df(x):
    return 2*x - 4

def gradient_descent(initial_x, learning_rate, num_iterations):
    x = initial_x
    x_history = [x]

    for i in range(num_iterations):
        gradient = df(x)
        x = x - learning_rate * gradient
        x_history.append(x)

    return x, x_history

initial_x = 0
learning_rate = 0.1
num_iterations = 50

x, x_history = gradient_descent(initial_x, learning_rate, num_iterations)

print("Local minimum: {:.2f}".format(x))

# Create a range of x values to plot
x_vals = np.linspace(-1, 5, 100)

# Plot the function f(x)
plt.plot(x_vals, f(x_vals))

# Plot the values of x at each iteration
plt.plot(x_history, f(np.array(x_history)), 'rx')

# Label the axes and add a title
```

```
plt.xlabel('x')
plt.ylabel('f(x)')
plt.title('Gradient Descent')

# Show the plot
plt.show()
```

Output:



```
Local minimum: 2.00
```


Gradient Descent
f(x)
x