```python
#Matematyka Konkretna
#Laboratorium 7
#Biegun Daniel https://github.com/S1Daniel/MK
#Wariant 2

import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D

# Modified function to minimize
def funkcja(x, y):
    return np.cos(x + 3*y)**np.sin(x)

# Modified gradient descent function
def gradient_descent(learning_rate, iterations):
    x = np.random.uniform(0, 2*np.pi)
    y = np.random.uniform(0, 2*np.pi)

    history = []

    for _ in range(iterations):
        # Numerical gradients
        df_dx = -np.sin(x + 3*y) * np.sin(x) - 3 * np.cos(x + 3*y) *
np.cos(x) * np.sin(x)
        df_dy = -3 * np.cos(x + 3*y) * np.sin(x)

        x = x - learning_rate * df_dx
        y = y - learning_rate * df_dy
        history.append([x, y, funkcja(x, y)])

    return np.array(history)

# Optimization
learning_rate = 0.01
iterations = 100
history = gradient_descent(learning_rate, iterations)

# Visualization of the function
x_vals = np.linspace(0, 2*np.pi, 100)
y_vals = np.linspace(0, 2*np.pi, 100)
X, Y = np.meshgrid(x_vals, y_vals)
Z = funkcja(X, Y)

# 3D plot initialization
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.plot_surface(X, Y, Z, cmap='viridis', alpha=0.8, edgecolor='k')

# Initial point
ax.scatter(history[0, 0], history[0, 1], history[0, 2], color='red',
```

```
marker='o', s=100, label='Start')

# Trajectory
ax.plot(history[:, 0], history[:, 1], history[:, 2], color='blue',
marker='o', label='Minimization')

# Final minimum point
ax.scatter(history[-1, 0], history[-1, 1], history[-1, 2],
color='green', marker='o', s=100, label='Minimum')

ax.set_xlabel('X')
ax.set_ylabel('Y')
ax.set_zlabel('f(X, Y)')
ax.legend()

plt.show()


C:\Users\macie\AppData\Local\Temp\ipykernel_7960\4259914912.py:12:
RuntimeWarning: invalid value encountered in power
  return np.cos(x + 3*y)**np.sin(x)
```