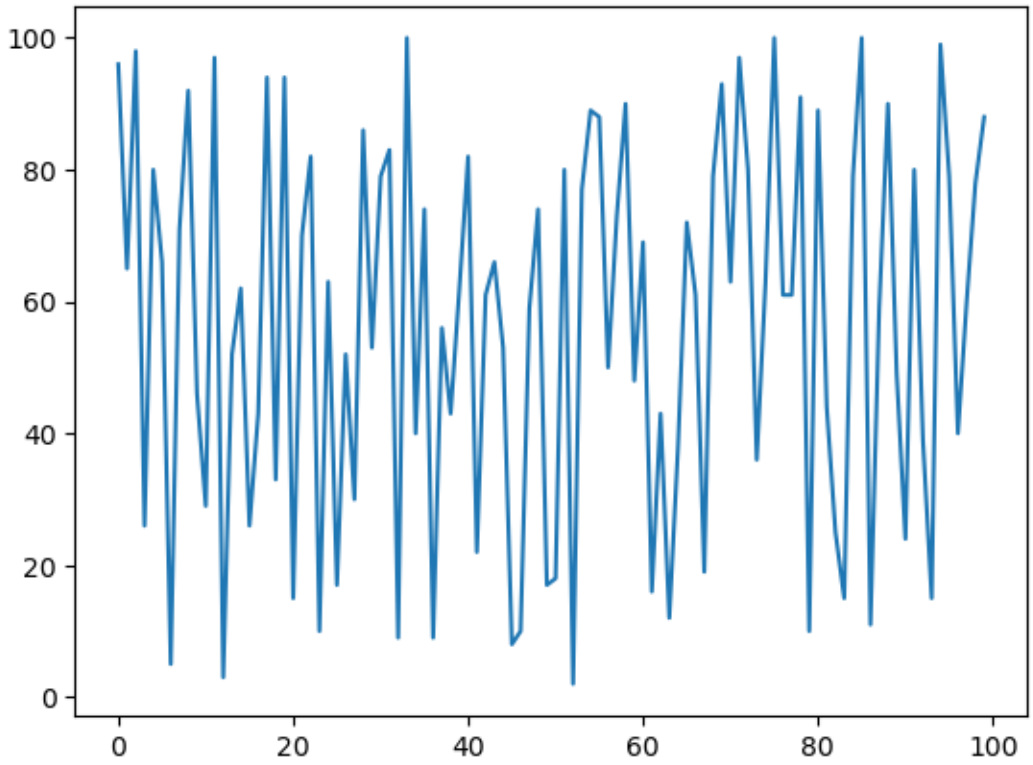
ovzrkyns7

August 18, 2025

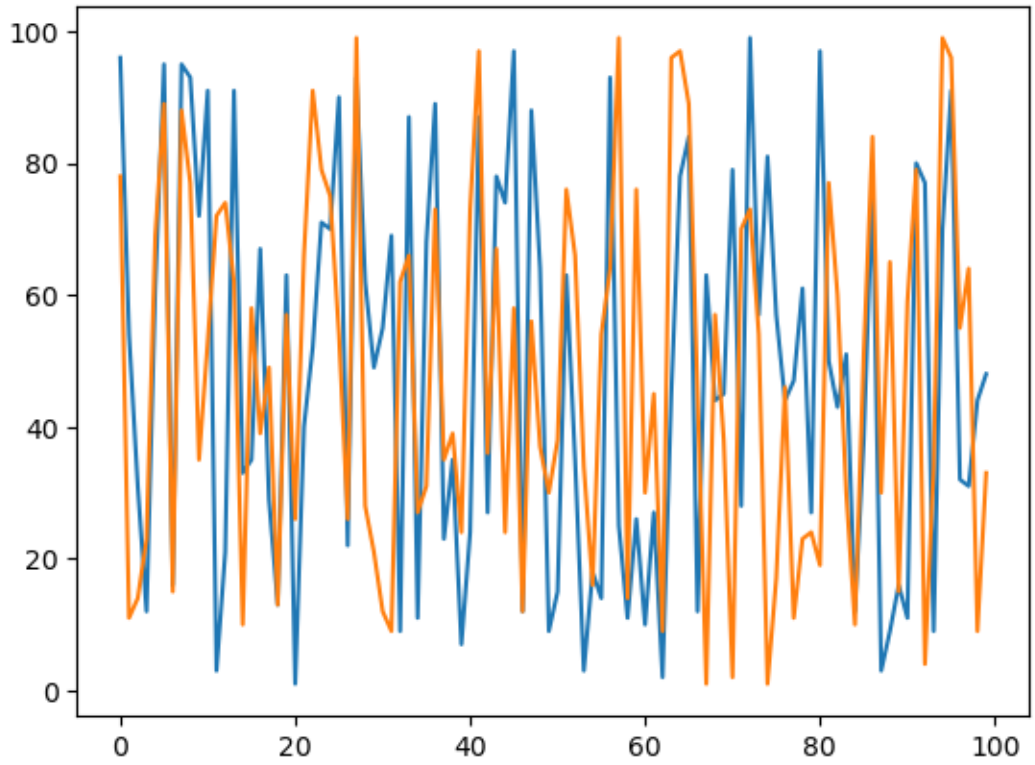
[1]:
```
student_name = "Tien Phat Nguyen"  # fill your name
student_id = "223171213"  # fill your student ID
print("Student name: " + student_name)
print("Student ID: " + student_id)
```

```
Student name: Tien Phat Nguyen
Student ID: 223171213
```

[2]:
```
import random
import matplotlib.pyplot as plt

n_values = 100
y_values = []

# Create data (y_values) randomly between 1 and 100.
for i in range(n_values):
    y_values.append(random.randint(1, 100))

x_values = range(n_values)  # X is sequence of values 0-99
plt.plot(x_values, y_values)
plt.show()
```



[3]:
```
# Plot 2 variables
#

n_values = 100
y_values_1 = []
y_values_2 = []

# Create data (y_values) randomly between 1 and 100.
for i in range(n_values):
    y_values_1.append(random.randint(1, 100))
    y_values_2.append(random.randint(1, 100))

x_values = range(n_values)  # X is sequence of values 0-99
plt.plot(x_values, y_values_1)
plt.plot(x_values, y_values_2)  # call plot again draws in the same graph.
plt.show()
```



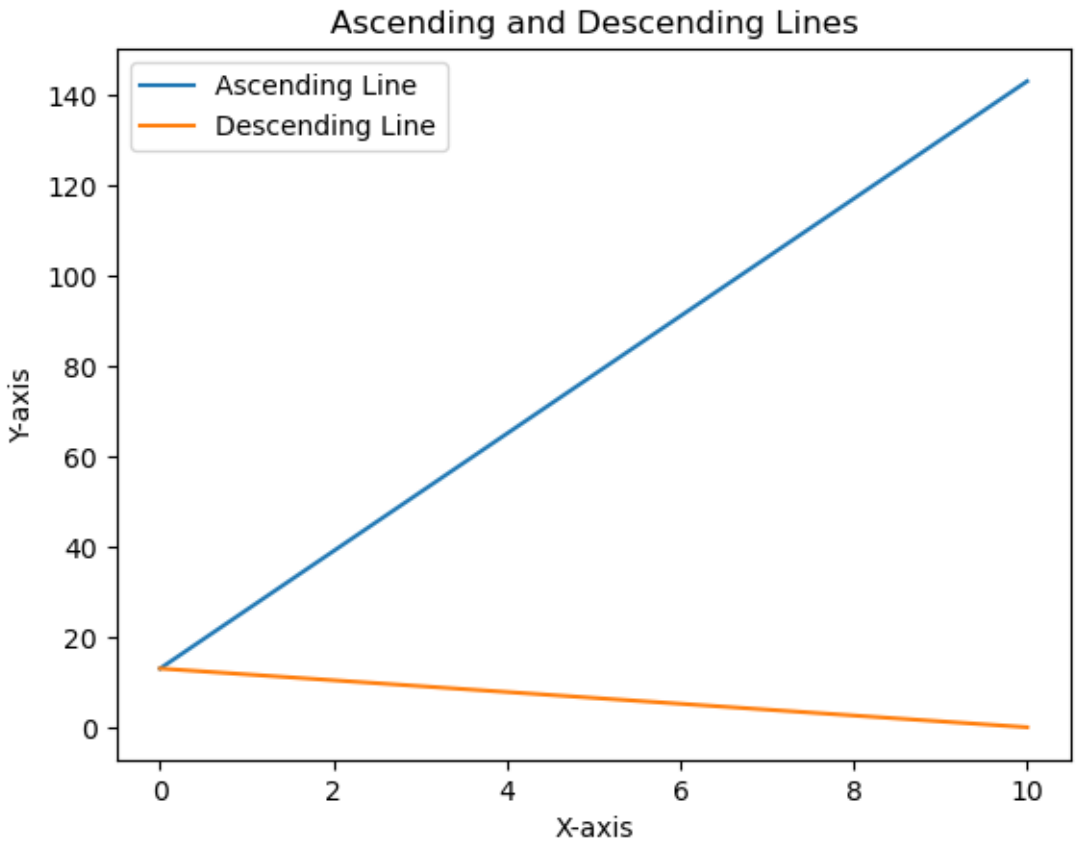
[16]:
```python
#
# Activity 1: Create data so that the plot draws an
# ascending line (y_values increase at any rate).
#
import numpy as np

x_ascending = np.linspace(0, 10, 50)
y_ascending = 13 * x_ascending + 13
print("Sample data points for ascending line:")
for i in range(0, len(x_ascending), 10):
    print(f"x = {x_ascending[i]:.2f}, y = {y_ascending[i]:.2f}")

print(f"Y-values range: {y_ascending.min():.2f} to {y_ascending.max():.2f}")
```

```
Sample data points for ascending line:
x = 0.00, y = 13.00
x = 2.04, y = 39.53
x = 4.08, y = 66.06
x = 6.12, y = 92.59
x = 8.16, y = 119.12
Y-values range: 13.00 to 143.00
```

[15]:
```
#
# Activity 2: Create data so that the plot draws a
# descending line (y_values decrease at any rate).
#
x_descending = np.linspace(0, 10, 50)
y_descending = -1.3 * x_descending + 13

print("Sample data points for descending line:")
for i in range(0, len(x_descending), 10):
    print(f"x = {x_descending[i]:.2f}, y = {y_descending[i]:.2f}")

print(f"Y-values range: {y_descending.max():.2f} to {y_descending.min():.2f}")
```

```
Sample data points for descending line:
x = 0.00, y = 13.00
x = 2.04, y = 10.35
x = 4.08, y = 7.69
x = 6.12, y = 5.04
x = 8.16, y = 2.39
Y-values range: 13.00 to 0.00
```

[10]:
```
plt.plot(x_ascending, y_ascending, label='Ascending Line')
plt.plot(x_descending, y_descending, label='Descending Line')
plt.xlabel('X-axis')
plt.ylabel('Y-axis')
plt.title('Ascending and Descending Lines')
plt.legend()
plt.show()
```

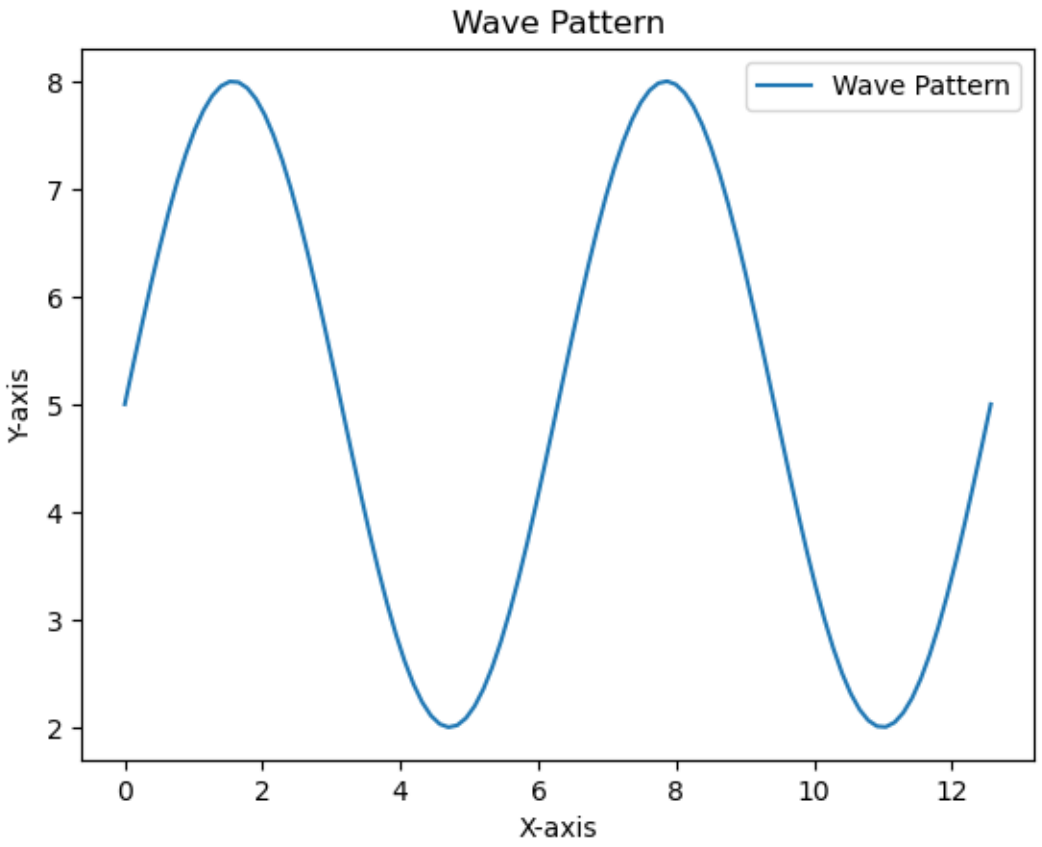


[ ]:
```python
#
# Activity 3: Create data so that the plot draws a
# wave. You can consider using Python's math libarary, which has
# a sin function (detail https://www.w3schools.com/python/ref_math_sin.asp).
#
import math
x_wave = np.linspace(0, 4 * math.pi, 100)
y_wave = [3 * math.sin(x) + 5 for x in x_wave]

print("Sample data points for wave pattern:")
for i in range(0, len(x_wave), 20):
    print(f"x = {x_wave[i]:.2f}, y = {y_wave[i]:.2f}")

print(f"Y-values range: {min(y_wave):.2f} to {max(y_wave):.2f}")
```

```
Sample data points for wave pattern:
x = 0.00, y = 5.00
x = 2.54, y = 6.70
x = 5.08, y = 2.20
x = 7.62, y = 7.92
```

```
x = 10.15, y = 3.00
Y-values range: 2.00 to 8.00
```

[17]:
```
plt.plot(x_wave, y_wave, label='Wave Pattern')
plt.xlabel('X-axis')
plt.ylabel('Y-axis')
plt.title('Wave Pattern')
plt.legend()
plt.show()
```



[ ]: