Python Task

1. Read and Display a Log File

Simulate analyzing log data.

```
with open("server_logs.txt", "r") as file:
    for line in file.readlines()[:10]:  # print first 10 lines
        print(line.strip())
```

Purpose: Understand file I/O and log inspection.

2. Extract IP Addresses from Logs

```
import re

log_data = """
192.168.0.10 - user1 login success
10.0.0.12 - user2 failed login
172.16.0.5 - admin login success
"""

ips = re.findall(r'\d+\.\d+\.\d+\.\d+', log_data)
print("Extracted IPs:", ips)
```

Purpose: Learn regex (crucial for parsing logs, network packets, etc.)

3. Count Number of Failed Logins

```
logs = [
    "user1 login success",
    "user2 failed login",
    "user3 failed login",
    "user4 login success"
]

failed = sum("failed" in log for log in logs)
print("Failed logins:", failed)
```

Purpose: Basic anomaly counting — used in intrusion detection.

4. Visualize Attack Frequency (Matplotlib)

```
import matplotlib.pyplot as plt

attacks = ["Brute Force", "Phishing", "Malware", "DDoS"]
counts = [30, 20, 50, 10]
```

```
plt.bar(attacks, counts)
plt.title("Attack Frequency")
plt.xlabel("Attack Type")
plt.ylabel("Count")
plt.show()
```

Purpose: Visual data analysis for reports/dashboards.

5. Create a Simple Threat Scoring Function

```
def threat_score(failed_logins, malware_alerts, ddos_signals):
    score = (failed_logins * 1.5) + (malware_alerts * 2) + (ddos_signals * 3)
    return score

print("Threat Score:", threat_score(3, 2, 1))
```

Purpose: Foundation for AI-based scoring and risk prediction.

6. Basic Data Cleaning for Threat Dataset

```
import pandas as pd
```

```
data = {
    'ip': ['192.168.1.1', '10.0.0.1', None, '172.16.0.2'],
    'status': ['ok', 'fail', 'ok', 'fail']
}
df = pd.DataFrame(data)
df = df.dropna()  # remove missing values
print(df)
```

Purpose: Preprocessing — key for AI model input data.

7. Basic Machine Learning Threat Classification (Intro)

```
from sklearn.tree import DecisionTreeClassifier
import numpy as np

# Features: [failed_logins, malware_alerts, ddos_signals]
X = np.array([[1, 0, 0], [5, 1, 0], [2, 0, 1], [10, 2, 3]])
# Labels: 0 = safe, 1 = threat
y = np.array([0, 1, 1, 1])

model = DecisionTreeClassifier()
model.fit(X, y)
```

```
print(model.predict([[2, 1, 0]]))  # Predict for new case
```

Purpose: Introduces AI-based classification logic.