**Algorithm Approach: Vulnerability Prioritization**

**Overview**

This document outlines the approach used for prioritizing vulnerabilities in the provided dataset. The algorithm assigns a priority score to each vulnerability based on several factors, including CVSS score, severity, due date, source weight, and fix availability.

**Factors and Weights**

The prioritization formula considers the following weighted factors:

1. **CVSS Score (Weight: 3)**
   * The CVSS score (ranging from 0 to 10) represents the base severity of a vulnerability.
   * A higher score means a more severe vulnerability.
2. **Severity Level (Weight: 2)**
   * The severity level is mapped to numerical values as follows:
     + Critical: 4
     + High: 3
     + Medium: 2
     + Low: 1
3. **Fix Availability (Weight: 2)**
   * If a fixed version is available, the vulnerability receives a boost in priority.
   * Mapping:
     + Fixable: 1
     + Not Fixable: 0
4. **Source of Vulnerability (Weight: 1.5)**
   * Different sources of vulnerabilities are assigned different weights:
     + AWS: 1.2
     + GitHub: 1.0
     + Other sources can be assigned weights based on relevance.
5. **Time Until Due Date (Weight: -0.1)**
   * The closer the due date, the higher the priority.
   * Negative weight ensures that vulnerabilities closer to the due date are prioritized over those with a later due date.

**Priority Score Formula**

The priority score is calculated using the following formula:

Priority Score=(CVSS×3)+(Severity Score×2)+(Fix Available×2)+(Source Weight×1.5)−(Days Until Due×0.1)\text{Priority Score} = (\text{CVSS} \times 3) + (\text{Severity Score} \times 2) + (\text{Fix Available} \times 2) + (\text{Source Weight} \times 1.5) - (\text{Days Until Due} \times 0.1)

**Time Complexity Analysis**

The algorithm involves the following steps:

1. **Data Cleaning and Preprocessing** - O(N)
2. **Mapping Severity and Fixability** - O(N)
3. **Converting Dates and Computing Time Differences** - O(N)
4. **Sorting the Dataframe Based on Priority Score** - O(N log N)

Thus, the overall time complexity of the prioritization algorithm is:

O(N log N)

where **N** is the number of vulnerabilities in the dataset.

**Conclusion**

This approach ensures that vulnerabilities with high severity, closer due dates, and available fixes are prioritized while maintaining computational efficiency.