**AI Powered code completion**

The manual implementation of the sorting function uses Python's sorted() method with a lambda that directly accesses the target key. While simple and efficient for complete datasets, it raises a KeyError if the specified key is missing from any dictionary.

The AI-suggested version from GitHub Copilot introduces robustness by using .get(sort\_key, 0) instead of direct access. This ensures that dictionaries missing the sort key default to 0 during sorting, preventing runtime errors.

In terms of efficiency, both approaches use the same underlying sorting mechanism (Timsort, O(n log n) complexity). However, the AI-suggested code adds minor overhead with the .get() method, but this overhead is negligible for most datasets.

The AI-generated code demonstrates better real-world readiness, gracefully handling incomplete data, a common scenario in production. Therefore, **the AI version is more efficient in robustness and fault-tolerance**, while the manual version is slightly more efficient in strictly controlled datasets.

**Efficiency Comparison**

While both the AI-suggested and human-written sorting functions achieve the same result with similar time complexity of **O(n log n)**, the AI-generated version is slightly more efficient due to its compact use of lambda expressions and minimal function call overhead (Python Docs, 2024). The human-friendly version, though more readable and easier for beginners to understand, introduces an additional function call (get\_sort\_value), which slightly increases execution time in large datasets (Van Rossum, 2023). However, this difference is often negligible in practical applications.

**Therefore, the AI-suggested version is more efficient in terms of computational performance, while the human-written version is more efficient in terms of readability and learning clarity.**