Prompt 2. Phase 1 - Direct approach.

I want you to act as a text complexity analyzer and ranker.

Your task is to analyze a set of text files and rank them based on their text complexity. Text complexity should be evaluated as a generalization of the following features:

1. **Lexical Diversity:** Measure the variety of words used in the text. (Consider using metrics like Type-Token Ratio or more sophisticated measures like Moving-Average Type-Token Ratio - MATTR).

2. **Lexical Density:** Calculate the proportion of lexical words (nouns, verbs, adjectives, adverbs) to the total number of words.

3. **Syntactic Complexity:** Assess the complexity of sentence structure. (Consider features like average sentence length, number of clauses per sentence, or parse tree depth - though simpler proxies might be more practical for a general AI).

4. **Text Coherence:** Evaluate how well the text is logically organized and flows smoothly. (This is more subjective, but AI can approximate this through discourse analysis, pronoun resolution, or vector-based semantic similarity between sentences).

5. **Named Entity Recognition (NER):** Analyze the variety and density of named entities (persons, organizations, locations, dates, etc.) present in the text. A text with a wider range and higher density of NER types might be considered more complex.

6. **Text Readability:** Calculate readability scores using established formulas (e.g., Flesch Reading Ease, Flesch-Kincaid Grade Level, SMOG, Automated Readability Index - ARI, Dale–Chall readability formula, etc.). Choose a few relevant formulas that align with your concept of complexity.

Instructions:

For each text file provided, perform the following steps:

1. **Analyze each of the six features** listed above. For each feature, generate a numerical score or metric that quantifies it. Clearly state which metric you are using for each feature.

2. **Generalize from these six feature scores to create a single, overall "Text Complexity Score" for each text file.** Explain your method of generalization. Consider using a weighted average, a summation after normalization, or another suitable method to combine these disparate measures into a single score that represents overall text complexity. **State explicitly how you are combining these features.**

3. **Normalize all Text Complexity Scores.** After calculating the overall Text Complexity Score for each file, normalize these scores to a common scale (e.g., 0 to 1, or a Z-score distribution, or rank-based normalization). Explain your normalization method. This ensures that the final scores are comparable across different files and feature scales.

4. **Rank the text files.** Based on the normalized Text Complexity Scores, rank the text files numerically, where rank 1 is assigned to the file with the highest text complexity (most complex text), rank 2 to the next most complex, and so on.

5. **Output:** For each input text file, provide the following output in a clear and organized format