Test Cases:

 **Meet Data Size Requirement:** All your "large" datasets (data\_100k\_random.txt, data\_100k\_sorted\_asc.txt, data\_100k\_sorted\_desc.txt, data\_100k\_sparse.txt, data\_large\_duplicates.txt, data\_negative\_numbers.txt) explicitly state they contain 100,000 elements, directly fulfilling the "at least 100,000 tuples or rows or data points" requirement.

 **Test int Data Type:** All files contain integers, which aligns with your proposal's schema ("single column of int type").

 **Comprehensive Distribution Testing (for Search Algorithms):**

* data\_100k\_random.txt: Tests average-case performance for both Jump and Interpolation search on typical, unsorted-then-sorted data.
* data\_100k\_sorted\_asc.txt: Tests best-case scenarios for sorted data.
* data\_100k\_sorted\_desc.txt: Tests the loadAndSortDatasetFromFile's sorting capability on initially inverted data.
* data\_100k\_sparse.txt: This is especially crucial for comparing Jump Search and Interpolation Search. Interpolation Search performs best on uniformly distributed data, so sparse data will highlight its worst-case behavior (which can degrade to linear time, O(N)), providing a strong point of comparison with Jump Search's more consistent O(N​) performance.
* data\_large\_duplicates.txt: Tests how loading and sorting handle duplicate values, ensuring the final sorted array is correct, even if the search algorithms only find the first occurrence.

 **Excellent Edge Case Coverage:**

* data\_empty.txt: Tests the loadAndSortDatasetFromFile function's handling of an empty file, ensuring it doesn't crash and correctly reports an empty dataset.
* data\_single\_element.txt: Tests the smallest possible valid dataset for both loading and your search algorithms, ensuring they behave correctly with minimal input.

 **Robust Negative/Error Handling Testing:**

* data\_small\_invalid.txt and data\_mixed\_invalid.txt: These are vital for testing the try-catch blocks within your loadAndSortDatasetFromFile function. They ensure your program gracefully handles non-integer lines, out-of-range numbers, and empty lines within the file, rather than crashing. This demonstrates robust input validation.

 **Positive Range Testing:**

* data\_negative\_numbers.txt: Confirms that your integer handling (loading and searching) works correctly with negative values, which is a common requirement for numerical data.