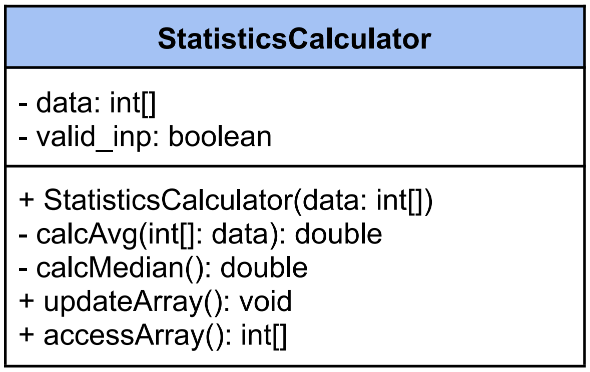
Assignment01-Q1:



StatisticsCalculator

- data: int[]
- valid_inp: boolean

+ StatisticsCalculator(data: int[])
- calcAvg(int[]: data): double
- calcMedian(): double
+ updateArray(): void
+ accessArray(): int[]

Attributes:
data: An array of integers to hold the data for which statistics will be calculated.
Valid_inp: A boolean flag indicating whether the input data is valid.

Methods:
StatisticsCalculator: A public constructor that initializes the StatisticsCalculator object with a given array of integers (data).

- calcAvg: A private method that takes an integer array (data) and calculates the average of its elements, returning a double value.
- calcMedian: A private method that calculates the median of the data stored in the object and returns it as a double.
- updateArray: A public method that updates or modifies the data array.
- accessArray: A public method that returns the data array.

Assignment01-Q6:
In q3:
String cleaning was divided into smaller methods, each handling a specific subtask: removing punctuation, converting to lowercase, and removing extra spaces. This way it was easier to maintain and modify individual aspects of the process.

q4:
The functionality was divided into separate methods, such as calculating word frequencies, finding the longest word, and checking for palindromes. I applied it here because by dividing the problem into smaller components, the logic became more modular and easier to maintain.

q5:
Following the same pattern from q4, breaking down the task into separate methods for calculating the average word length, finding the most frequent letter, and sorting the words alphabetically helped in making a cleaner code and easy management of the various functionalities, ensuring each part can be improved without affecting the rest of the system.

Assignment01-Q7:
Open Closed principle was mainly applied in q5 where we extended the StringAnalyzer to the TextAnalyzer.

Benefits of Applying the Open-Closed Principle:

- Maintainability: New functionality can be added without the risk of breaking existing code. For example, extending the StringAnalyzer class into the TextAnalyzer class.
- The system is easy to extend by adding new features (e.g., new TextAnalysis functions while keeping the previous code intact and not needing to modify it.) This makes coding easier and more efficient.
- Reusability: The existing classes and methods were reused in new contexts.
- Lesser risk of errors and bugs.