**Sorting Algorithms Performance Report**

**Overview**

This repository contains C++ code implementing various sorting algorithms and measuring their performance on random, increasing, and decreasing arrays of different sizes. The implemented algorithms include Bubble Sort, Selection Sort, Insertion Sort, and Shell Sort.

**Code Structure**

**File: sortFunctions.cpp**

1. **generateRandomArray**
   * Generates an array with random integer values.
2. **generateIncreasingArray**
   * Generates an array with values in increasing order.
3. **generateDecreasingArray**
   * Generates an array with values in decreasing order.
4. **bubbleSort**
   * Implements the Bubble Sort algorithm.
5. **selectionSort**
   * Implements the Selection Sort algorithm.
6. **insertionSort**
   * Implements the Insertion Sort algorithm.
7. **shellSort**
   * Implements the Shell Sort algorithm.
8. **main**
   * Contains the main execution logic.
   * Measures the execution time of each sorting algorithm on random, increasing, and decreasing arrays of different sizes.
   * Prints the execution times for each scenario.

**Usage**

* The code can be compiled and executed using a C++ compiler (e.g., g++).
* Adjustments to array sizes and the random number range can be made in the main function.

**Execution Results**

The code has been tested, and the following performance results were obtained:

Bubble Sort Random Array (n=100): 0 ms

--------------------------------------------

Bubble Sort Random Array (n=1000): 2 ms

--------------------------------------------

Bubble Sort Random Array (n=10000): 251 ms

--------------------------------------------

Similar results are obtained for other sorting algorithms and array types.

**Recommendations**

1. **Code Documentation**
   * Comments are added at the top of the file, each function, and throughout the code for better readability and understanding.
2. **Input Validation**
   * Consider adding input validation to ensure the provided array sizes are within reasonable limits.

**Conclusion**

The code successfully implements and measures the performance of various sorting algorithms. The provided performance results indicate the time complexity of the Bubble Sort algorithm for random arrays of different sizes. For a more comprehensive assessment, further testing and documentation enhancements