Alyssa McGaughey

413 Algorithms

30 November 2025

Integer Array Sorting and Inversions

Abstract

This project utilized python to implement an efficient algorithm to count the number of inversions within a list of 10,000 integers, the algorithm then sorts the array and calculates the runtime of the algorithm to output to the user. This project uses an enhances merge sort, modeled after the GeeksForGeeks method to count inversions, to demonstrate a practical use of divide-and-conquer algorithm and python performance.

Problem Description

The goal of this project is to read from a file of integers and determine the total number of inversions within that array. An inversion is formed when two elements a[i] and a[j] are in the array where a[i] > a[j] and i < j, such as within the array {3, 1, 2} where (3, 1) form an inversion. The array location of 3 is zero, which is less than the array location of 1, which is one, however, 3 is greater than 1, therefore an inversion occurs. This program takes that concept and counts any inversion within a 100,000 number array. To improve efficiency, the divide-and-conquer method used, counts these inversions in O(n log n) time.

The Algorithm

```
BEGIN PROGRAM
READ all integers from "IntegerArray.txt" into array
SET N = length of ARR

FUNCTION MergeSort(ARR, N):
     CREATE TEMP array
     RETURN MergeSortHelper(ARR, TEMP, 0, N-1)

FUNCTION MergeSortHelper(ARR, TEMP, LEFT, RIGHT):
     SET INV_COUNT = 0
     IF LEFT < RIGHT:
     MID = (LEFT + RIGHT) / 2
     INV_COUNT += MergeSortHelper(ARR, TEMP, LEFT, MID)
     INV_COUNT += MergeSortHelper(ARR, TEMP, MID+1, RIGHT)
     INV_COUNT += Merge(ARR, TEMP, LEFT, MID, RIGHT)
     RETURN INV_COUNT
```

```
FUNCTION Merge(ARR, TEMP, LEFT, MID, RIGHT):
     INITIALIZE indexes i = LEFT, j = MID+1, k = LEFT
     SET INV_COUNT = 0

     WHILE both subarrays have elements:
     IF left value <= right value:
     COPY left value into TEMP
     MOVE left index
     ELSE:
     COPY right value into TEMP
     ADD number of remaining left elements to INV_COUNT
     MOVE right index

     COPY any remaining left subarray elements into TEMP
     COPY any remaining right subarray elements into TEMP
     COPY TEMP back into ARR for this segmen
     RETURN INV_COUNT

START timer
CALL MergeSort on ARR
STOP timer

PRINT count
PRINT runtime
WRITE sorted ARR to "output.txt"
END PROGRAM
```

Implementation Details

The project was implemented using python and a GeeksForGeeks program as reference. While I am getting better at programming using Python, I still needed a bit of help, however, using their reference I was able to add file input, runtime measurements, and an output file to better fit the needs of my program.

Conclusion

Through this project I learned more about implementing a divide and conquer algorithm and adapting an algorithm to fit the needs of a problem, such as counting inversions. I was also able to gain more experience within python, as it's a language I'm still learning, and was glad to be able to use recursive algorithms within my program. It also showed me more on reading and writing files to add to future projects.

References

"Python Program to Count Inversions in an Array: Set 1 (Using Merge Sort)." *GeeksforGeeks*, GeeksforGeeks, 23 July 2025, www.geeksforgeeks.org/python/python-program-for-count-inversions-in-an-array-set-1-using-merge-sort/.