CSC 413 (Only) Course Programming Project

Inversion Count by Divide-and-Conquer

Note: This project is only for CSC 413 students. If you are a CSC 513 student, please look at the other project.

Inversion Count for an array indicates – how far (or close) the array is from being sorted. If array is already sorted then inversion count is 0. If array is sorted in reverse order then inversion count is the maximum. Formally speaking, two elements $a[i]$ and $a[j]$ form an inversion if $a[i] > a[j]$ and $i < j$. **Example:** The sequence 2, 4, 1, 3, 5 has three inversions (2, 1), (4, 1), (4, 3).

Project Description: the file "IntegerArray.txt" included in this project folder contains all the 100,000 integers between 1 and 100,000 (inclusive) in some order, with no integer repeated. Your task is to compute the number of inversions in the file given, where the *i-th* row of the file indicates the *i-th* entry of an array. Because of the large size of this array, you should implement a **divide-and-conquer** algorithm.

Grade: 100 marks in total, while it accounts for **25%** of the final grade.

Due time: November 30, 2025, 11:59 PM

Type: Independent work

Programming language: C/C++, Java, Python, or any other advanced programming languages

Submission:

1. **Venue: Canvas**.
2. **Contents:**
   (1) **Code**. Please add necessary comments to the code to help me understand and grade your program.
   (2) **Running instruction and results**. Please provide a Readme.txt file to list the steps to compile and run your program, including the software environment. It is also REQUIRED to provide your running time and running result (a sorted array saved in a file) by including the relevant **screenshots/files** in a **"Result"** folder.
   (3) **Project report**. Each student is required to submit a project report including at least the following items: **abstract**, **problem description**, **algorithm (pseudocode)**, **implementation details**, **running results and analysis**, **conclusions on what you have learned in the project**, and **references** (if applicable, also explicitly cite them in your project report).

Please **zip** all the above documents into one package and name it as "**Your_Last_Name_Your_First_Name.zip**". The upload page will be closed after the due date, so please upload your assignment before the deadline. You may submit your assignments several times before the deadline. No email submission is accepted.

CSC 413 Course Programming Project – Q&A

Inversion Count by Divide-and-Conquer

Question 1:

Good afternoon Dr. Li,

I am having some trouble determining the best place to count the inversions with the project. With it being a divide and conquer algorithm, would it be best to count inversions during the merge steps? Or during the sort? or both?

In my mind, it makes the most sense on the merge steps, due to the fact that you can just test for inversions at the lowest merge point for each branch and obtain an exact count. But, I have tried a few different ways, but none of them have been 100% accurate for every test case. If you have any direction, I would greatly appreciate it.

Answer:

The idea is to add the inversion counting during sorting. What you need to do is adding the inversion counting part to the MergeSort algorithm.

Below are two helpful observations for you to develop the algorithm.

Observation 1: *Given two subarrays, the inversions between the first and second subarrays remains the same regardless of the order of elements in each subarray.*

Observation 2: *Given two sorted subarrays A1 and A2, if A1[i] > A2[j], A1[i] and A2[j] are inverted, and every element in A1 after A1[i] is an inversion of A[j] as well.*

Question 2:

I am working on the general outline of my report, and I noticed you including a potential references section. I was just wondering, what kind of tools could we use for this project? Specifically what kind of reference could we have used?

>>Here references mainly means books and online resources (webpages, blogs, etc). You can refer to whatever you can find, but the bottom line is that you cannot copy others' idea, let alone code.

Question 3:

Also, for the psuedocode portion, does that have to be perfect? I simply included the merge functions since the whole thing is rather long.

>>For completeness, please provide the complete version of the pseudocode for your algorithm. Please avoid pasting the source code. Instead, please follow the conventions we learned to write the pseudocode. Thanks!

Question 4:

Final question, in the results and analysis section, obviously you want the overall results of the project, but can you elaborate as to what you expect to be in the analysis portion?

>>You are expected to show the results/outputs (including #of inversions, as well as the total running time of your program) of your program and provide some analysis on the correctness and . Also, if you want to earn some extra points, you may perform a theoretical complexity analysis on your algorithm, and compare with that of a brute-force approach, both theoretically and practically (i.e., compare with the total running time of a brute-force implementation).

Question 5:

I am working on the project now, and I was wondering if the text file integers can be read in as a vector instead of an array or would that take points off? I know this is likely a dumb question, but after my midterm I want to make sure I get full marks for everything.

>> Yes, you have the freedom to choose whatever data structure you like.

Question 6:

Is there anyway to know for sure if the value of inversions that I have is accurate?

>>Yes: you can use a brute-force method (embedded for loops) to count the number of inversions in the sequence.

CSC 413 Course Project Grading Details

1. Correctness (60):

2. Efficiency (20):

3. Report (20):

-abstract (2):

-problem description (2):

- algorithm (pseudocode) (2):

- implementation details (8):

- running results and analysis (4):

- conclusions on what you have learned in the project (2):

- references ():

4. Extra Credit (10):

Total: