

Attempt 1 

In Progress

NEXT UP: Submit Assignment

 Add Comment

Unlimited Attempts Allowed

1/5/2025 to 1/23/2025

Details

Part 1: The 4-sum problem

Brute-force algorithm

(20 Points) Develop and unit-test a brute-force solution to the following 4-sum problem:

Given an array of N distinct integers, find all 4 integers a, b, c, d such that $a + b + c + d = 0$.

Input: A file of integers and the number of integers contained in the file read in as command line arguments.

Output: The total number of quadruples that sum to zero.

Tests:

```
FourSum 8ints.txt 8 ---> Result: 4
FourSum 50ints.txt 50 ---> Result: 162
FourSum 100ints.txt 100 ---> Result: 1368
FourSum 200ints.txt 200 ---> Result: 20664
```

(10 Points) Analyze the running time of your algorithm. State the cost model and use the Big-O notation in your analysis. Your answer has to be justified.

Submit: (1) working and properly commented source code for the brute-force program, (2) screenshots of the outputs for each of the test files, and (3) response to analysis question included as a clearly delineated comment block in your program.

Improved algorithm

(25 points) Develop and unit-test an algorithm for the 4-sum problem that runs faster than the brute-force solution above. This program should have identical input, output, and test specifications described above. Confirm that the results of the unit tests for the improved algorithm are the same as the brute-force algorithm.

(10 Points) Analyze the running time of your algorithm. State the cost model and use the Big-O notation in your analysis. Your answer has to be justified.

Submit: (1) working and properly commented source code for the improved program, (2) screenshots of the outputs for each of the test files, and (3) response to analysis question included as a clearly delineated comment block in your program.

Comparison

(15 Points) Run both algorithms using the given **5Hints.txt** and **1Kints.txt** input files and measure their running times using the provided **StopWatch** class (of the **alg_stopwatch.h** file). Your code should print out the recorded running times for both algorithms. Provide your observations on the running time differences and how it relates to your previous algorithm analysis.

Submit: (1) working and properly commented source code for the updated program that instruments the **StopWatch** class, (2) screenshots of the outputs for each of the test files showing clearly labeled running times for both algorithms on the two different input files - four outputs total, and (3) observations about results included as a clearly delineated comment block in your program.

Part 2: Quicksort Best Case Array

(20 Points) Write a program that produces a best-case array (with no duplicates) for the Quicksort algorithm, defined as an array of n distinct items having the property that every partition will produce subarrays that differ in size by at most 1. For the purposes of this

exercise, ignore the initial randomization shuffle. Array keys should be only upper-case letters following traditional alphabetical order, i.e. "ABCDEFGH...."

Input: Array size n read in as a command line argument.

Output: The contents of the best-case array.

Tests:

```
Quickbest 3 ---> Result: BAC
Quickbest 7 ---> Result: DACBFEG
Quickbest 15 ---> Result: HACBFEGDLIKJNMO
Quickbest 22 ---> Result: KACDBHGFIEQMLPNTSRUV
```

Submit: (1) working and properly commented source code for the best-case array program, and (2) screenshots of the outputs for each of the test inputs

NOTE: Up to 10% of the grade will be deducted from uncommented and poorly indented code.

Assignment 1 Files.zip (<https://weber.instructure.com/courses/596800/files/116043316?wrap=1>). [↓](https://weber.instructure.com/courses/596800/files/116043316/download?download_frd=1)
(https://weber.instructure.com/courses/596800/files/116043316/download?download_frd=1)

View Rubric

Programming Assignment 1

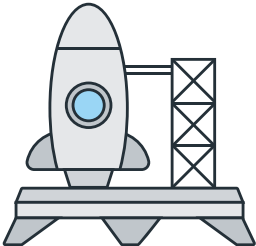
Criteria	Ratings			Pts
Four-Sum: properly commented and formatted code	5 pts Satisfactory commenting and formatting.	3 pts Nominal commenting and formatting.	0 pts No commenting or formatting.	/ 5 pts
Brute-force algorithm: correctness	17.5 to >0 pts Program compiles and Meets Specifications - Output Screenshots submitted.		0 pts Program does not compile and does not meet specifications - Output screenshots not submitted.	/ 17.5 pts
Brute-force algorithm: analysis	10 to >0 pts Accurate analysis, including cost model, big-O notation, and justification..		0 pts Analysis not submitted	/ 10 pts
Improved algorithm: correctness	22.5 to >0 pts Program compiles and Meets Specifications - Output Screenshots submitted.		0 pts Program does not compile and does not meet specifications - Output screenshots not submitted.	/ 22.5 pts
Improved algorithm: analysis	10 to >0 pts Accurate analysis, including cost model, big-O notation, and justification..		0 pts Analysis not submitted	/ 10 pts
Comparison: testing	10 pts Run both programs on input files 5Hints.txt and 1Kints.txt and print running times using Stopwatch class..		0 pts Program running times not generated.	/ 10 pts
Comparison: observations	5 pts Observations submitted on the running time differences and how it relates to your previous algorithm analysis.		0 pts Observations not submitted	/ 5 pts
Quicksort Best-Case: properly commented and formatted code	5 pts Satisfactory commenting and formatting.	3 pts Nominal commenting and formatting.	0 pts No commenting or formatting.	/ 5 pts
Quicksort Best-Case: correctness	15 to >0 pts Program compiles and Meets Specifications - Output Screenshots		0 pts Program does not compile and does not meet specifications - Output screenshots	/ 15 pts

Criteria	Ratings	Pts
	submitted.	not submitted.

Total Points: 0

Choose a submission type





Drag a file here, or

Choose a file to upload

or

