COSC2430 Hw3: Sort

v1.1

1. Introduction

You will create a C++ program to sort data. After sorting the data, you need to output the required information to a plain text file.

2. Input and Output

a. Input data file

1) The input data file has multiple records (number >= 1 and <=100000). Your program should read the records one by one from the beginning of the file to the end.
2) Each record is a number which range from 0 to 99999999 (no decimal part), the numbers are separated by space or new line. Each number may appear more than once in one data file, add it even it duplicated.
3) The input data file doesn't have dirty records, like letters and special characters.

b. Input command file

1) The input command file only contains sorting commands which are quick sort, merge sort and shell sort along with the location of the desired data in the sorted results. The location starts from 0. It may out of the range of the data length, you should ignore the location if it happens.

2) For each sorting command, the default sorted results should be “ascending” (smaller number first). But it can also give you “descending”.
3) One file will only have one sorting command.

c. Output file

1) After finishing sorting command, you should output the results to output file.
2) The output file should be clean. Results should be separated by space and in same line, no need ‘\n’ in the end of the line.

3. Algorithm

Your code should use the same algorithms in our text book, the book is “D. S. Malik, Data Structures Using C++, 2e. Course Technology - Cengage Learning, 2010, ISBN-13: 9780324782011”. The algorithms other from the book may lead to execution time error which lead to your code failure.

For quick sort and shell sort, you should use array. For merge sort, you should use linked list.

4. Program specification and Examples

The main C++ program will become the executable to be tested by the TAs. The results should be written to another text file (output file), provided with the command line. Notice the input and output files are specified in the command line, not hard code inside the C++ code. All the necessary parameters will be in

the command line when calling your program, so you don't need to worry about the missing file name problem. When calling your program, the format would be one of the two standard types as below, no mixed calling type will be given. Notice also the quotes in the program call, to avoid Unix/Windows get confused.

The general call to the executable is as follows:

sort "input=input1.txt;output=output1.txt;command=command1.txt"

Call example with another command line type.

sort input=input1.txt output=output1.txt command=command1.txt

The execution time for each group of input files has threshold, if the time out of the range of the desired time, it fails. If the answers were not the same as correct answers, it fails too.

Example 1 of input and output

input31.txt
1 2 3 4 5 6 7 8 9

command31.txt
quick sort ascending 0 5 2

Command line:
sort input=input31.txt output=output31.txt sort=sort31.txt

output31.txt
1 6 3

Example 2 of input and output

input32.txt
1 3 4 2 6 9 5 7 100 20

command32.txt

shell sort descending 2 4 6

Command line:
sort “input=input32.txt;output=output32.txt;command=command32.txt”

output32.txt
9 6 4

Example 3 of input and output

input33.txt
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41

command33.txt
merge sort 1 3 5

Command line:
sort input=input33.txt output=output33.txt command=command33.txt

output33.txt
3 7 11

You may see the time limitation when you run test.sh.

Real test cases differ from above three examples.

5. Requirements summary

- C++:

  Per different sort type, you must create an array or a linked list to store the data.

  Your program will be tested with GNU C++. Therefore, you are encouraged to work on Unix. You can use other C++ compilers, but the TAs cannot provide support or test your programs with other compilers.

- Output:

  Your program should write error messages to the standard output (cout, ptintf).

  Your program should output results within desired milliseconds, if timeout, you fail that test case.

6. Turn in your homework

Homework 3 need to be turned in to our Linux server, follow the link here http://www2.cs.uh.edu/~rizk/homework.html.

Make sure to create a folder under your root directory, name it hw3 (name need to be lower case), only copy your code to this folder, no testcase or other files needed.

ps. This document may have typos, if you think something illogical, please email TAs or Teacher for confirmation.