COMP 2004 Data Structures and Algorithms

Assessment

Total Points: 160 (Regular Parts) + 30 (if Bonus Part in Question 6 is solved correctly)

Submission Instructions

1. For questions that require software code answers, submit your C source code only. Do not submit object code (.o files) or executables (.exe files)
2. For questions that require a written answer, submit your answer in the form of a pdf file.
3. gcc compiler will be used for compiling and testing the code
4. All submissions, code and text answers, must be made via Canvas only. Do not email your submissions. Emailed submissions will not be graded.
5. Name your submitted files as <Your last name>_<Assignment number>_<question number-part-number>.c (or .pdf, for text-based, written answers)
   For example, if your last name is Smith, for assignment 2, question number 1, part b, your answer (C code) should be in a file named Smith_2_1b.c
6. Put all your submission files inside a folder named as <Your last name>_<Assignment number>. For example, if your last name is Smith, for assignment 2, the submission folder should be named Smith_2.
7. Students not using the submission file naming convention and folder structure given in steps 5-6 above will lose points.
8. Late submissions and partial credits will be according to the class policies given in the first lecture slides.
9. All students must adhere to the honor code of the course that they signed on the first day of classes.

Questions 1 through 4 require hand-written answers. You could type these answers electronically and submit as a pdf file or write them down on paper and scan and upload a pdf.

Question 1 (30 points: 15 points for part a), 15 points for part b)

a) Write a program that counts the number of inversions in an input array of integers. Your program should prompt the user to enter the number of elements in the input array and then the array elements. It should then print the number of inversions in the input array. A sample input is shown below:

```
Enter the number of elements in input array: 15
Enter the array elements: -35 11 42 -7 -18 30 -44 23 -25 -6 9 -19
50 -49 38
```

b) This question requires a written answer. Show the result of running Shellsort on the following input with the increment sequence {1, 3, 7}. At each step, you must show the array and record the number of swaps, as shown on the lecture slides.

-46 73 10 -89 55 -32 28 91 -15 -79 64 -3 42 -68 7 -20 86 -95 49 -52

How many inversions are there in the above input? Use your program in part a) to calculate the number of inversions in the array in part b) and write the answer in the same file that you wrote down the shell sort output above.

How many swaps are saved by using shell sort over the total number of inversions? Write the answer in the same file that you wrote down the shell sort output and number of inversions above.

Question 2 (10 points)

This question requires a written answer. Sort the following array in ascending order using Heap sort.

142 543 123 65 453 879 572 434 111 242 811 102

Your answer should start by showing the array represented both as a binary heap following the heap structure and heap order properties, and the array corresponding to the heap. At each step, show the heap and the array including the array's heap and non-heap portions. The final array should show the elements in sorted order.

Question 3 (10 points)

This question requires a written answer. Sort the following array in ascending order using Merge Sort.

-12 45 78 -31 17 -88 54 62 -76 29 -40 83 91 -55 6 -98

Your answer should show the divided arrays at each step of the recursion going down to smaller arrays and the merged arrays at each step of backtracking of the recursion, that ultimately gives the full sorted array.

Question 4 (10 points)

This question requires a written answer. Sort the following array in ascending order using Quick sort with median-of-three partitioning and a cutoff of 3.

63 87 144 118 -2 46 -16 7 -32 99 68 37 128 55 -12 76

Your answer should show the subsets at each step of the recursion going down and the merged sets at at each step of backtracking of the recursion, that ultimately gives the full sorted array.

Question 5 (50 points)

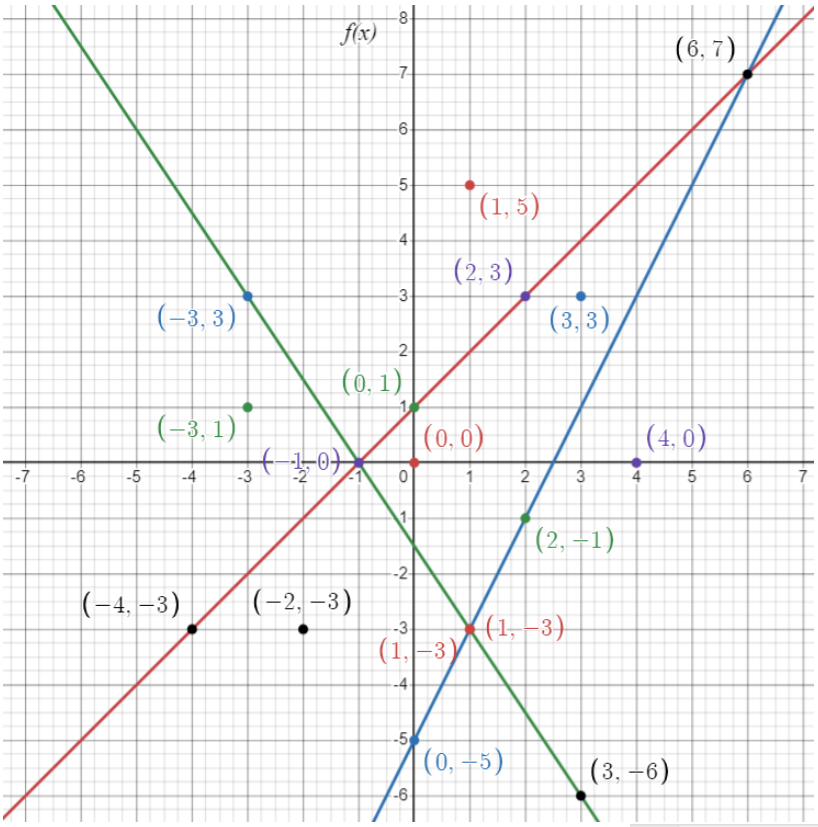
Write a program that reads N points in a plane and outputs any group of four or more collinear points (i.e., points on the same line). The obvious brute-force algorithm requires $O(N^4)$ time. The program you write should use any sorting algorithm and solve the problem in $O(N^2 \log N)$ time. Your program should not print groups of three or less collinear points.

A sample input output is given below. Your programs input and output should use the same format.

```
Enter the number of data points: 16
Enter the data points: (-1, 0) (2, 3) (3, 3) (4, 0) (0, 0) (-3, 3) (2,
-1) (1, -3) (0, 1) (0, -5) (-3, 1) (1, 5) (6, 7) (-2, -3) (3, -6) (-4,
-3)


Collinear points found:
(-1, 0) (-3, 3) (1, -3) (3, -6)
(-1, 0) (0, 1) (2, 3) (6, 7) (-4, -3)
(1, -3) (6, 7) (0, -5) (2, -1)
```

The graph with the above data points and collinear points with equations is shown below for clarity. Your program does not need to find the equations or plot the graphs.



1. $y = x + 1$
2. $y = 2x - 5$
3. $y = -1.5x - 1.5$
4.
5. $(-4, -3)$ Label:
6. $(1, -3)$ Label:
7. $(-3, 3)$ Label:
8. $(0, 1)$ Label:
9. $(2, 3)$ Label:
10. $(6, 7)$ Label:
11. $(0, -5)$ Label:
12. $(2, -1)$ Label:
13. $(3, -6)$ Label:
14. $(1, 5)$ Label:
15. $(3, 3)$ Label:
16. $(-3, 1)$ Label:
17. $(4, 0)$ Label:
18. $(-2, -3)$ Label:
19. $(0, 0)$ Label:

Question 6 (Regular Question 50 points, Bonus Part: 30 points)

We are given an array that contains N integers. We want to determine if there are two integers whose sum equals a given number K. Example input and output are shown below:

```
Enter the number of array elements: 4
Enter the array elements: 8, 4, 1, 6
Enter the target sum (K): 10
Output: Success!
a[1] = 4
a[3] = 6
Sum = 10
```

Bonus points : Solve the above question, for **any** number of integers whose sum equals a given number K. An example input and output are shown below:

```
Enter the number of array elements: 6
Enter the array elements: -4, 13, 8, -2, 1, 0
Enter the target sum (K): 7
Output: Success!
a[0] = -4
a[1] = 13
a[3] = -2
Sum = 7

Enter the number of array elements: 6
Enter the array elements: -4, 13, 8, -2, 1, 0
Enter the target sum (K): 20 26
Output: Fail!
```

For the bonus part, note that if K is an element in the array, it is a valid solution. If there are multiple sequences that yield a sum of K, your program should output all possible sequences.

Using a sorting algorithm to sort the input first gives a quick and elegant solution to the problem. Your program should use this technique.