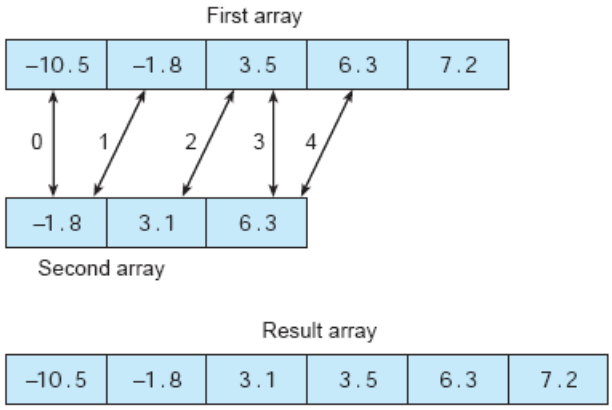
EC1003 程式設計

Spring 2018

Homework 4

Due: 23:55PM, June. 4

1. Write a function that will merge the contents of two sorted (ascending order) arrays of type double values, storing the result in an array output parameter (still in ascending order). The function should not assume that both its input parameter arrays are the same length but can assume that one array does not contain two copies of the same value. The result array should also contain no duplicate values.



Assume the max length of the first and second array is 20.

Hint: When one of the input arrays has been exhausted, do not forget to copy the remaining data in the other array into the result array. Test your function with cases in which (1) the first array is exhausted first, (2) the second array is exhausted first, and (3) the two arrays are exhausted at the same time (i.e., they end with the same value). Remember that the arrays input to this function *must already be sorted.*

Write a main program and test your function by the following cases:

a) 1st array: -12 -7.65 -2.4 0 1.18 7.92 12.888 24.23
   2nd array: -13.5 -1.423 0.023 1.0112 2.5

b) 1st array: -256.44 -101.393 -33.2356 -1.123 6.8
   2nd array: -4.543 -1.123 64.8 140.231 236.22 484.93 5893.231 32233.2

c) 1st array: -23.2 -14.8 34.69 39.88 57.45
   2nd arrary: -16.83 -6.34 8.893 15.223 60.54

2. (Ch. 7, Prob. 7.16) ***(Card Shuffling and Dealing Modification)*** In the card shuffling and dealing program of Fig. 7.24, we intentionally used an inefficient shuffling algorithm that introduced the possibility of indefinite postponement. In this problem, you'll create a high-performance shuffling algorithm that avoids indefinite postponement. Modify the program of Fig. 7.24 as follows. Begin by initializing the deck array as shown in Fig. 7.29. Modify the shuffle function to loop row-by-

row and column-by-column through the array, touching every element once. Each element should be swapped with a randomly selected element of the array.

Print the resulting array to determine if the deck is satisfactorily shuffled (as in Fig. 7.30, for example). You may want your program to call the shuffle function several times to ensure a satisfactory shuffle.

Note that although the approach in this problem improves the shuffling algorithm, the dealing algorithm still requires searching the deck array for card 1, then card 2, then card 3, and so on. Worse yet, even after the dealing algorithm locates and deals the card, the algorithm continues searching through the remainder of the deck. Modify the program of Fig. 7.24 so that once a card is dealt, no further attempts are made to match that card number, and the program immediately proceeds with dealing the next card.

Unshuffled deck array

	0	1	2	3	4	5	6	7	8	9	10	11	12
0	1	2	3	4	5	6	7	8	9	10	11	12	13
1	14	15	16	17	18	19	20	21	22	23	24	25	26
2	27	28	29	30	31	32	33	34	35	36	37	38	39
3	40	41	42	43	44	45	46	47	48	49	50	51	52

Fig. 7.29 | Unshuffled deck array.

Sample shuffled deck array

	0	1	2	3	4	5	6	7	8	9	10	11	12
0	19	40	27	25	36	46	10	34	35	41	18	2	44
1	13	28	14	16	21	30	8	11	31	17	24	7	1
2	12	33	15	42	43	23	45	3	29	32	4	47	26
3	50	38	52	39	48	51	9	5	37	49	22	6	20

Fig. 7.30 | Sample shuffled deck array.

3. (Ch. 8, Prob. 8.14) ***(Tokenizing Telephone Number)*** Write a program that inputs a telephone number as a string in the form (02)2733-3141. The program should use function strtok to extract the area code as a token, the first four digits of the phone number as a token and the last four digits of the phone number as a token. The eight digits of the phone number should be concatenated into one string. The program should convert the area code string to int and convert the phone-number string to long. Both the area code and the phone number should be printed.
(Put a 0 in front of the conversion specifier will make the program to print the leading zeros in front of a number. For example: printf("%08d", num);)

4. (Ch. 8, Prob. 8.20) ***(Counting the Number of Words in a String)*** Write a program that inputs 4 lines of text and uses strtok to count the total number of words. Assume that the words are separated by either spaces or newline characters.

   Test your program by the following input:
   This line of text has seven words
   This line has five words
   There are three words on the next line
   I am happy

Hand in:

1. Source code (You must write comments to explain your codes.)
2. Running script of your results