Theory exercises for the Programming Techniques Course

May 2024

This document collects solved exercises with steps, exercises with solution and exercises in Exam platform mode relating to the following theoretical topics covered during the Course:

- **complexity analysis**
- **union-find algorithms for online connectivity**
  - quick-find
  - quick-union
  - weighted quick-union
- **quadratic iterative sorting algorithms**
  - insertion sort
  - bubble / exchange sort
  - selection sort
  - Shell sort
- **linear iterative sorting algorithms**
  - counting sort
  - radix sort
- **linearithmic iterative sorting algorithms**
  - bottom-up merge sort

Complexity analysis

Given the following function Foo

```
int Foo (int *collector, int n, int k)
{
    int i, j, acc_l;
    for (i=0; i<n-4; i++)
    {
        acc_l = 0;
        for (j=0; j<4; j++)
            acc_l += collector[i+j];
        if (acc_l == k)
            return i;
    }
    return -1;
}
```

- Find the worst-case asymptotic complexity (loose bound) of the function **T(n)** = **O(…)**
  - There are **2** nested loops. The number of iterations of the outer one depends lineraly on **n**. The number of iterations of the inner one is fixed.
    Thus **T(n)** = **O(n)**.
- Find the worst-case asymptotic complexity (strict bound) of the function **T(n)** = **Θ (…)**
  - Unable to define a strict bound because of the non-structured `return 1` statement that forces an exit from the function if the condition `acc_l == k+1` is met. The outer loop is executed at most n-4 times, thus **O(n)**, not exactly n-4 times.
- The function looks for an interval of consecutive elements in the array. What is the distinctive feature of this interval?
  - It is an interval of 4 consecutive elements that sum up to `k`.
- If there are several intervals satisfying the above condition, would the function return the first one, the last one or a random one?
  - The function terminates with a non-structured `return` instruction as soon as an interval is found that satisfies the condition. Thus, it returns the index of the first one.

Union-Find Algorithms for Online Connectivity

Exercise solved with steps: quick-find
Let the following sequence of pairs be given:

3-2 3-4 5-1 7-3 5-7 9-1

where relation **i-j** indicates that vertex **i** is adjacent to vertex **j.** Apply an on-line connectivity algorithm with quick-find, returning as a result the content of the array at each step. The vertices are named with integers between 0 and 9.

Solution
The table below shows array id at each step.

	0	1	2	3	4	5	6	7	8	9
	0	1	2	3	4	5	6	7	8	9
3-2	0	1	2	2	4	5	6	7	8	9
3-4	0	1	4	4	4	5	6	7	8	9
5-1	0	1	4	4	4	1	6	7	8	9
7-3	0	1	4	4	4	1	6	4	8	9
5-7	0	4	4	4	4	4	6	4	8	9
9-1	0	4	4	4	4	4	6	4	8	4

Exercise with solution: quick-union
Let the following sequence of pairs be given:

1-3 3-4 5-1 4-0 7-3 5-7 9-4 2-6

where relation **i-j** indicates that vertex **i** is adjacent to vertex **j**. Apply an on-line connectivity algorithm with quick-union, returning as a result the content of the array at each step. The vertices are named with integers between 0 and 9.

Solution
The table below shows array id at each step.

	0	1	2	3	4	5	6	7	8	9
	0	1	2	3	4	5	6	7	8	9
1-3	0	3	2	3	4	5	6	7	8	9
3-4	0	3	2	4	4	5	6	7	8	9
5-1	0	3	2	4	4	4	6	7	8	9
4-0	0	3	2	4	0	4	6	7	8	9
7-3	0	3	2	4	0	4	6	0	8	9
5-7	0	3	2	4	0	4	6	0	8	9
9-4	0	3	2	4	0	4	6	0	8	0
2-6	0	3	6	4	0	4	6	0	8	0

Exercise solved with steps: weighted quick-union

Let the following sequence of pairs be given:

3-9 4-3 2-5 6-2 3-8 5-6 0-10 3-5 8-9 10-3

where relation **i-j** indicates that vertex **i** is adjacent to vertex **j**. Apply an on-line connectivity algorithm with weighted quick-union, returning as a result the content of the arrays **id** and **sz** at each step. The vertices are named with integers between 0 and 10.

Solution

The following table shows array **id** at each step:

	0	1	2	3	4	5	6	7	8	9	10
	0	1	2	3	4	5	6	7	8	9	10
3-9	0	1	2	9	4	5	6	7	8	9	10
4-3	0	1	2	9	9	5	6	7	8	9	10
2-5	0	1	5	9	9	5	6	7	8	9	10
6-2	0	1	5	9	9	5	5	7	8	9	10
3-8	0	1	5	9	9	5	5	7	9	9	10
5-6	0	1	5	9	9	5	5	7	9	9	10
0-10	10	1	5	9	9	5	5	7	9	9	10
3-5	10	1	5	9	9	9	5	7	9	9	10
8-9	10	1	5	9	9	9	5	7	9	9	10
10-3	10	1	5	9	9	9	5	7	9	9	9

The following table shows array sz at each step:

	0	1	2	3	4	5	6	7	8	9	10
	1	1	1	1	1	1	1	1	1	1	1
3-9	1	1	1	1	1	1	1	1	1	2	1
4-3	1	1	1	1	1	1	1	1	1	3	1
2-5	1	1	1	1	1	2	1	1	1	3	1
6-2	1	1	1	1	1	3	1	1	1	3	1
3-8	1	1	1	1	1	3	1	1	1	4	1
5-6	1	1	1	1	1	3	1	1	1	4	1
0-10	1	1	1	1	1	3	1	1	1	4	2
3-5	1	1	1	1	1	3	1	1	1	7	2
8-9	1	1	1	1	1	3	1	1	1	7	2
10-3	1	1	1	1	1	3	1	1	1	9	2

Remember that an answer in which the substitution criterion applied does not respect the constraint of the weighted quick-union to merge the tree of minor cardinality into the one of major cardinality is incorrect.

Exercise with solution: quick-find

Let the following sequence of pairs be given:

1-2 3-5 0-1 7-4 5-1 9-0 6-5

where relation **i-j** indicates that vertex **i** is adjacent to vertex **j**. Apply an on-line connectivity algorithm with quick-find, returning as a result the content of the array at the final step. The vertices are named with integers between 0 and 9.

Solution

Content of **id** at the final step:

0	1	2	3	4	5	6	7	8	9
2	2	2	2	4	2	2	4	8	2

Exercise with solution: quick-union

Let the following sequence of pairs be given:

1-2 3-5 0-1 7-4 5-1 9-0 6-5

where relation **i-j** indicates that vertex **i** is adjacent to vertex **j**. Apply an on-line connectivity algorithm with quick-union, returning as a result the content of the array at the final step. The vertices are named with integers between 0 and 9.

Solution

Content of **id** at the final step:

0	1	2	3	4	5	6	7	8	9
2	2	2	5	4	2	2	4	8	2

Exercise in Exam platform mode: quick-union

Text:
given the following sequence of pairs, where relation **i-j** indicates that vertex **i** is adjacent to vertex **j**:

12-5, 1-4, 0-12, 3-9, 3-11, 11-5, 11-12, 3-5, 9-5

Apply an on-line connectivity algorithm with quick-union. Nodes are named with integers between 0 and 12.

Questions and format of answer:

- display array id as a sequence of integers after the step for pair 0-12
  id = 5 4 2 3 4 5 6 7 8 9 10 11 5

- display array id as a sequence of integers after the step for pair 11-5
  id = 5 4 2 9 4 5 6 7 8 11 10 5 5

- display array id as a sequence of integers after the step for pair 3-5
  id = 5 4 2 9 4 5 6 7 8 11 10 5 5

- display array id as a sequence of integers after the step for pair 9-5
  id = 5 4 2 9 4 5 6 7 8 11 10 5 5

- **N** is the number of vertices in the graph and **d** is the number of pairs of vertices (edges in the graph). Is the asymptotic worst-case complexity of the online connectivity algorithm with quick union:

  - O(N)
  - Θ(d)
  - O(dN)
  - Θ(dN)
  - Θ(dlogN)

- Briefly justify your answer:

Suggestion: review the transparencies.

Quadratic Iterative Sorting Algorithms

Insertion sort

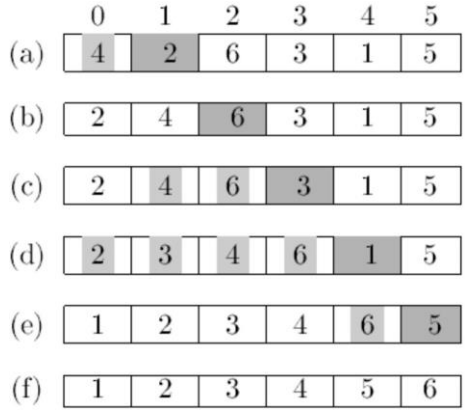
Exercise solved with steps

Sort the following sequence of integers in ascending order using insertion sort, indicating the steps:

4 2 6 3 1 5

Solution

The following figure provides a graphical representation of the steps.



Each array is obtained through an iteration of the outer cycle. The element in the cell highlighted entirely in dark gray represents the selected and extracted **x** element.

Items in cells partially highlighted in light gray are those shifted to the right by the shift operation performed by the inner loop. Once the shift process is over, the extracted value **x** is inserted in the position left empty, that is the one corresponding to the leftmost light gray element.

For example, during the first iteration (row (a)), value 2 is stored in variable **x**, value 4 is shifted to the right and **x** is inserted in position 0. During the second iteration (row (b)) **x** takes value 6, which however is immediately re-inserted in position 2 without any shift. During the third iteration (row (c)) value 3 is copied into **x**, value 6 and then the value 4 (in that order) are shifted one position to the right and value 3 is re-inserted in position 1. We proceed in this way by copying into **x** values 1 and 5 and finally, we obtain the final configuration of the array at the end of the algorithm in line (f).

Exercise with solution

Sort the following sequence of integers in descending order using insertion sort, indicating the relevant steps:

-3 2 1 10 15 5 -5 35 7 19

Solution

Since a descending ordering is required, let us remark that the shift condition is (j> = 1 && x> A [j]).
The following table provides a graphical representation of the steps.

0	1	2	3	4	5	6	7	8	9
-3	**2**	1	10	15	5	-5	35	7	19
2	-3	**1**	10	15	5	-5	35	7	19
2	1	-3	**10**	15	5	-5	35	7	19
10	2	1	-3	**15**	5	-5	35	7	19
15	10	2	1	-3	**5**	-5	35	7	19
15	10	5	2	1	-3	**-5**	35	7	19
15	10	5	2	1	-3	-5	**35**	7	19
35	15	10	5	2	1	-3	-5	**7**	19
35	15	10	7	5	2	1	-3	-5	**19**
35	19	15	10	7	5	2	1	-3	-5

Each row of the table corresponds to an iteration of the outer loop and the highlighted element represents the inserted one in the correct position of the left subarray.

Exercise in Exam platform mode

Text:
Sort the following sequence of integers in ascending order using insertion sort:

5 2_1 1 3 2_2 6

In the case of duplicate keys, the subscript indicates the position in the input sequence.

Questions and format of answer:

show the array as a sequence of integers after the insertion of 2_2.

Correct answer:
1 2_1 2_2 3 4 6

There may be subquestions about the algorithm. Suggestion: review the transparencies.

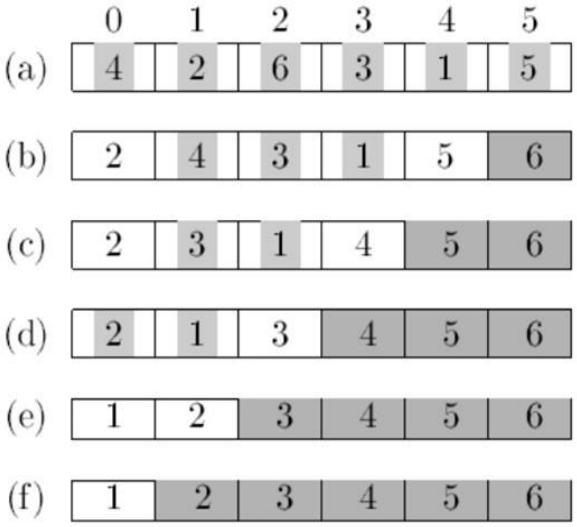
Exchange (Bubble) sort

Exercise solved with steps
Sort the following sequence of integers in ascending order using bubble sort, indicating the steps:

4 2 6 3 1 5

Solution

The following figure shows a graphical representation of how bubble sort proceeds. Each row of the table corresponds to an iteration of the outer loop.



Cells highlighted entirely in dark gray represent stable cells, i.e. those in which the element has reached its final position. These elements constitute the sorted subarray (initially empty). The elements in the cells partially highlighted in light gray are those being exchanged, proceeding from left to right. Such elements are in the subarray yet to be sorted (which initially coincides with the entire array). For example, during the first iteration (row (a)), the elements (4; 2), (6; 3), (6; 1) and (6; 5) are exchanged by moving element 6 to position 5. During the second iteration (line (b)) the elements (4; 3) and (4; 1) are exchanged and the index 4 is confirmed as the final position of the value
5. Line (f) shows the final configuration.

Exercise with solution
Given the sequence of integers stored in an array:

-1 0 -12 9 -9 16 8 3 1 0

sort it in descending order by applying exchange(bubble) sort. Graphically report the relevant steps of the algorithm and the final result.

Solution
Since a descending order is required, the shift condition is A [j] <A [j + 1]. The table provides a graphical representation of the steps. Each row of the table corresponds to an iteration of the outer loop that takes care of inserting a new element in the correct position (highlighted in gray).

0	1	2	3	4	5	6	7	8	9
-1	0	-12	9	-9	16	8	3	1	0
0	-1	9	-9	16	8	3	1	0	-12
0	9	-1	16	8	3	1	0	-9	-12
9	0	16	8	3	1	0	-1	-9	-12
9	16	8	3	1	0	0	-1	-9	-12
16	9	8	3	1	0	0	-1	-9	-12
16	9	8	3	1	0	0	-1	-9	-12
16	9	8	3	1	0	0	-1	-9	-12
16	9	8	3	1	0	0	-1	-9	-12
16	9	8	3	1	0	0	-1	-9	-12

Exercise in Exam platform mode

Text:

Sort the following sequence of integers in ascending order using optimized bubble sort:

5 3 2 6 4

Questions and format of answer:

After how many steps of the external cycle does the algorithm end?

Correct answer:
3

There may be subquestions about the algorithm. Suggestion: review the transparencies.

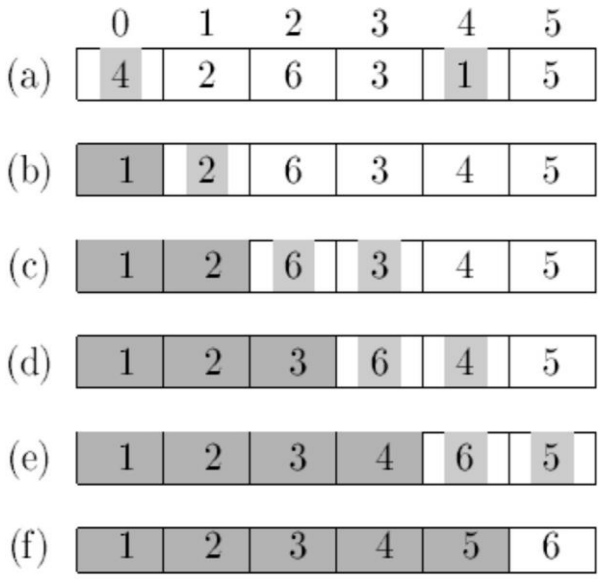
Selection sort

Exercise solved with steps

Sort the following sequence of integers in ascending order using selection sort, indicating the steps:

4 2 6 3 1 5

Solution
The following figure provides a graphical representation of the steps.



Each row of the table corresponds to an iteration of the outer loop. Cells highlighted entirely in dark gray represent stable cells, i.e. those in which the element has reached its final position. These elements constitute the sorted subarray (initially empty). The elements in the cells partially highlighted in light gray are the ones being exchanged, i.e. element A[i] and the smallest element in subarray still to be sorted. This value is selected by the inner loop and exchanged with element A[i]. For example, during the first iteration the smallest element (1) is in position 4. This element is exchanged with the element in position 0 (4) generating the configuration of row (b), in which the element in position 1 is in its final position. In line (b) the smallest element of the array is 2 which is therefore exchanged with "itself". Line (f) illustrates the final array configuration.

Exercise with solution

Given the sequence of integers, supposedly stored in an array:

-3 -2 2 9 -9 6 8 13 11 10

sort it in descending order resorting to selection sort. Graphically report the relevant steps of the algorithm and the final result.

Solution

The next table shows the steps and the final result. Since a descending ordering is required, note that the value sought for the exchange is the largest one in the right subarray.
Each row of the table corresponds to an iteration of the outer loop. In this iteration, the largest element of the right subarray (the white one, still to be sorted) is selected in the iteration of the inner loop and is inserted as the last element of the already sorted (dark colored) array.

0	1	2	3	4	5	6	7	8	9
-3	-2	2	9	-9	6	8	13	11	10
13	-2	2	9	-9	6	8	-3	11	10
13	11	2	9	-9	6	8	-3	-2	10
13	11	10	9	-9	6	8	-3	-2	2
13	11	10	9	-9	6	8	-3	-2	2
13	11	10	9	8	6	-9	-3	-2	2
13	11	10	9	8	6	-9	-3	-2	2
13	11	10	9	8	6	2	-3	-2	-9
13	11	10	9	8	6	2	-2	-3	-9
13	11	10	9	8	6	2	-2	-3	-9

Exercise in Exam platform mode

Text:

Sort the following sequence of integers in ascending order using selection sort:

$$A = (10_1, 5_1, 4_1, 4_2, 9, 5_2, 10_2, -1)$$

Subscripts denote instances of duplicate keys.

Questions and format of answer:

- Sorted array

$$A = (-1, 4_1, 4_2, 5_1, 5_2, 9, 10_2, 10_1)$$

- Asymptotic worst-case complexity of selection sort on N integers:
  - O(logN)
  - $\Omega(N)$
  - $\Theta(N^2)$
  - $\Omega(N^2)$

  Brief explanation of the answer: review the transparencies.

- Number of swaps in the worst case using selection sort to sort N integers :
  - exactly N
  - at most N
  - exactly N^2
  - at most N^2

  Brief explanation of the answer: review the transparencies.

When is a sorting algorithm stable? review the transparencies.

- Is selection sort stable?
  - Yes
  - No

  Brief explanation of the answer: review the transparencies.

Shell sort

Exercise solved with steps

Sort the following integer array in ascending order using Shell sort with Knuth's sequence:

7 6 8 9 8 6 2 1 8 7 0 4 5 3 0 1 0 0

Show the main steps.

Solution

Given the size of the array, the last value of Knuth's sequence is h = 13. By applying insertion sort to the subarrays whose successive elements are 13 apart from each other, we obtain 13-sorted subsequences:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
3	0	1	0	0	6	2	1	8	7	0	4	5	7	6	8	9	8

Continuing with h = 4, applying insertion sort to the subarrays whose successive elements are 4 apart from each other, we obtain 4-sorted subsequences:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
0	0	0	0	3	6	1	1	5	7	2	4	8	7	6	8	9	8

Finally with h = 1, applying the insertion sort to the subarrays whose successive elements are distant from each other by one position, the sorted array is obtained:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
0	0	0	0	1	1	2	3	4	5	6	6	7	7	8	8	8	9

Exercise with solution

Sort the following integer array in ascending order using Shell sort with Knuth's sequence:

17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

Show the main steps.

Solution

Given the size of the array, the last value of Knuth's sequence h = 13.

13-sorted sub-sequences:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
4	3	2	1	0	12	11	10	9	8	7	6	5	17	16	15	14	13

4-sorted sub-sequences:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
0	3	2	1	4	8	7	6	5	12	11	10	9	13	16	15	14	17

sorted array:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17

Exercise in Exam platform mode

Text:

Sort array A of integers in ascending order using Shell sort with Knuth's sequence:

A = 35 13 22 47 14 92 15 0 9 0 27 55 33 42 61 44

Questions and format of answer:

- Among the following sequences, which one is Knuth's sequence?
  - 1 2 4 8 16 …
  - 1 2 3 4 6 8 9 12 …
  - 1 4 13 40 121 …
  - 1, 5, 19, 41, 109, 209 …

  review the transparencies

- Using Knuth's sequence, given the size of array A, what is the starting value of h? Brief explanation of the answer: review the transparencies.

- Briefly explain the notion of h-sorted array: review the transparencies.

- Display as a sequence of integers the 4-sorted array after performing the previous steps:

  9 0 15 0 14 13 22 44 33 42 27 47 35 92 61 55

- Is 13 14 92 15 0 27 55 44 a subsequence or a subarray of array A ? Brief explanation of the answer: review the transparencies.

Linear Iterative Sorting Algorithms

Counting sort

Exercise solved with steps
Sort the following integer array A in ascending order by counting sort:

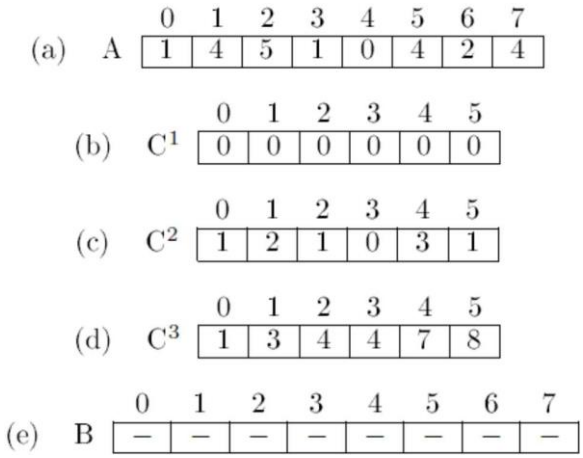
1 4 5 1 0 4 2 4

Show the data structures used in the intermediate steps.

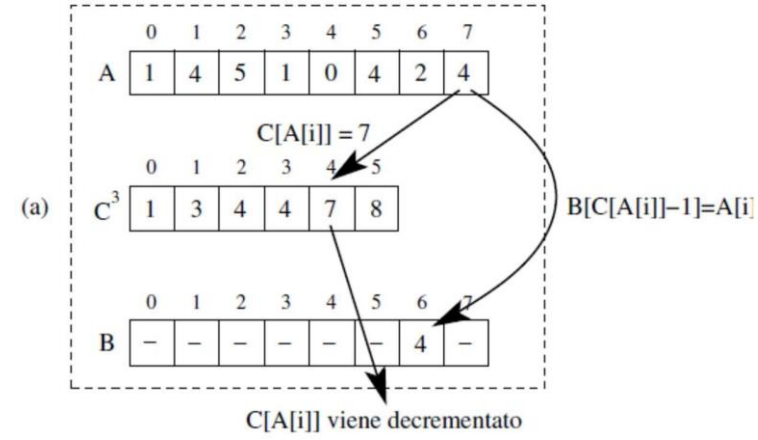
Solution

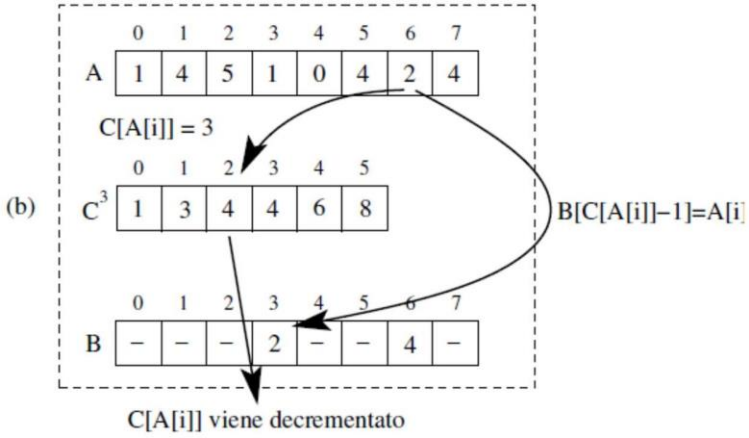
In the exercise we have n = 8 and k = 6. The following table shows in:

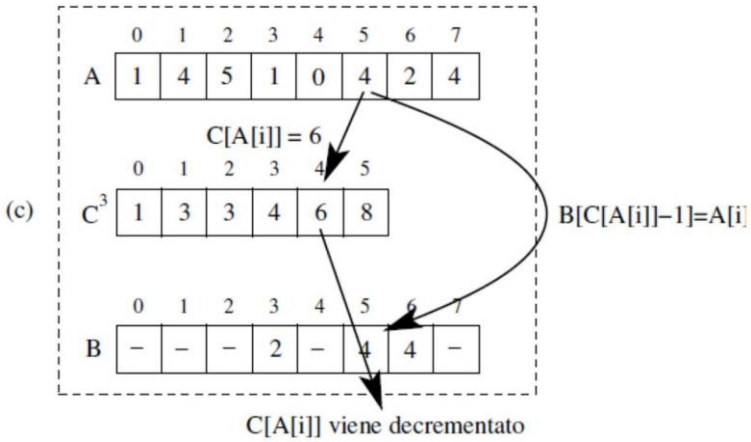
a) the original array A
b) array C after initialization (C1)
c) array C (simple occurrences, C2)
d) array C (multiple occurrences C3)
e) the result array B at the beginning.



The single elements of array A are then analyzed one after the other, starting from the rightmost one and proceeding towards the left. The following figure shows the first three steps of the procedure.







The first value considered is 4, using which as an index we obtain from the array of multiple occurrences that the number of elements of A with a value less than or equal to it equals 7. This means that 4 can be stored directly in the seventh element (at index 6) of the sorted array, precisely because it is already known that all the elements of the sorted array with index less than 6 will eventually contain values less than (or at most equal to) 4.

Array B will therefore take on the appearance of figure (a). At the same time, the element at index 4 of the array of multiple occurrences is decreased, the content of C being shown in figure (b). The penultimate element of A is then examined, that is 2. In this case, the number of elements of A with a value less than or equal to 2 equals 4, so 2 can be copied into the fourth element (at index 3) of B; the element of C of index 2 is also decremented. The corresponding arrays B and C are shown in figures (b) and (c), respectively.

At this point, we move on to consider the third last element of A, which is again 4. This time we deduce from the array of multiple occurrences that the number of "remaining" elements in A with a value less than or equal to 4 equals 6, so 4 is copied into the sixth element of B, and the corresponding element of C is again decremented: array B is shown in figure (c). We proceed in this way until all the elements of A have been considered. The final situation we reach is:

	0	1	2	3	4	5	6	7
B	0	1	1	2	4	4	4	5

where array B corresponds exactly to sorted array A. The final step in which array B is copied into array A is omitted, as it is trivial.

Exercise with solution

Sort the following integer array in ascending order by counting sort:

5 6 5 3 3 7 4 4 4 5 3 8 8

Show the data structures used in the intermediate steps.

Solution

The following table shows in order: the initial array A, the array C after the initialization (1), the computation of simple (2) and multiple (3) occurrences and the final array B. The decrements of the elements of C during the various phases of insertion of the original elements in the final array are also shown.

	0	1	2	3	4	5	6	7	8	9	10	11	12
A	5	6	5	3	3	7	4	4	4	5	3	8	8

	0	1	2	3	4	5	6	7	8
C^1	0	0	0	0	0	0	0	0	0

	0	1	2	3	4	5	6	7	8
C^2	0	0	0	3	3	3	1	1	2

	0	1	2	3	4	5	6	7	8
C^3	0	0	0	3	6	9	10	11	13
				2	5	8	9	10	12
				1	4	7			11
				0	3	6			

	0	1	2	3	4	5	6	7	8	9	10	11	12
B	3	3	3	4	4	4	5	5	5	6	7	8	8

Exercise in Exam platform mode

Text:
Sort the following integer array in ascending order by counting sort:

$2_1\ 4_1\ 3_1\ 1\ 3_2\ 5_1\ 4_2\ 2_2\ 4_3\ 5_2\ 3_3\ 9\ 8$

Questions and format of answer:

- Given the array to sort, what is the value of k? Briefly justify the answer.
  k =
- show as a sequence of integers the content of array C of simple occurrences
- at what index in the result array B will key 42 be stored?

Correct answer:

- k = 10 review the transparencies.
- C: 0 1 2 3 3 2 0 0 1 1
- at index 7

Radix sort

Exercise solved with steps

Sort the following array A of 2-digit base-8 numbers in ascending order using radix sort:

11 42 35 10 70 34 26 54

Show the data structures used in the intermediate steps.

Solution
Two counting sort steps are performed on data expressed on 1 octal digit, starting from the column containing the least significant digits.

Step 1: array D of the column of the least significant digits contains 1, 2, 5, 0, 0, 4, 6, 4. As the base is 8, the array of the simple/multiple occurrences consists of 8 elements. The array of simple occurrences contains 2 1 1 0 2 1 1 0, the array of multiple occurrences contains 2 3 4 4 6 7 8 8. After applying counting sort A contains 10 70 11 42 34 54 35 26.

Step 2: given the array A computed in the previous step, array D of the column of the most significant digits contains 1, 7, 1, 4, 3, 5, 3, 2. As the base is 8, the array of simple/multiple occurrences consists of 8 elements. The array of simple occurrences contains 0 2 1 2 1 1 0 1, the array of multiple occurrences contains 0 2 3 5 6 7 7 8. After applying counting sort A contains 10 11 26 34 35 42 54 70.

Exercise with solution

Sort the following array A of base-10 integers in ascending order using radix sort:

101 942 85 1023 70 4 26 154

Solution
d = 4
Step 1
- D = 1, 2, 5, 3, 0, 4, 6, 4
- C (simple occ.) = 1, 1, 1, 1, 2, 1, 1, 0, 0, 0
- C (multiple occ.) = 1, 2, 3, 4, 6, 7, 8, 8, 8, 8
- A = 70, 101, 942, 1023, 4, 154, 85, 26

Step 2
- D = 7, 0, 4, 2, 0, 5, 8, 2
- C (simple occ.) = 2, 0, 2, 0, 1, 1, 0, 1, 1, 0
- C (multiple occ.) = 2, 2, 4, 4, 5, 6, 6, 7, 8, 8
- A = 101, 4, 1023, 26, 942, 154, 70, 85

Step 3
- D = 1, 0, 0, 0, 9, 1, 0, 0
- C (simple occ.) = 5, 2, 0, 0, 0, 0, 0, 0, 0, 1
- C (multiple occ.) = 5, 7, 7, 7, 7, 7, 7, 7, 7, 8
- A = 4, 1023, 26, 70, 85, 101, 154, 942

Step 4

- D = 0, 1, 0, 0, 0, 0, 0, 0
- C (simple occ.) = 7, 1, 0, 0, 0, 0, 0, 0, 0, 0
- C (multiple occ.) = 7, 8, 8, 8 8, 8, 8, 8, 8, 8
- A = 4, 26, 70, 85, 101, 154, 942, 1023

Exercise in Exam platform mode

Text:

Sort the following array A of integers on 3 digits in base 10 in ascending order using radix sort:

200 643 387 820 272 607 743 294

Questions and format of answer:

- What condition makes radix sort stable? review the transparencies.
- step 1 unit column: show the initial content of array C of simple and multiple occurrences as a sequence of integers

  simple occurrences: C = 2, 0, 1, 2, 1, 0, 0, 2, 0, 0

  multiple occurrences: C = 2, 2, 3, 4, 6, 6, 6, 8, 8, 8

- step 1 tens column: display as a sequence of integers the contents of array A after the counting sort step

  200, 607, 820, 643, 743, 272, 387, 294

- the worst-case asymptotic complexity of radix sort for 3-digit N decimal integers is:
  - O(3N)
  - O(N)
  - O(NlogN)
  - $O(N^2)$

  Brief explanation of the answer: review the transparencies.

  What would the worst-case asymptotic complexity of radix sort for 3-digit N decimal integers be if insertion sort were used instead of counting sort to sort columns? Brief explanation of the answer: review the transparencies.

Linearithmic iterative sorting algorithms

Bottom-up merge sort

Exercise solved with steps
Sort the following array A of integers in ascending order by bottom-up merge sort:

121 42 58 13 7 41 161 314 15 9 27

Solution
Observe that n = 11, not a power of 2. Arrays formed by a single element are by definition sorted. At each step contiguous pairs of sorted arrays are merged to obtain a sorted array of size equal to the sum of the sizes of the 2 arrays.

Step 1: 121 42 58 13 7 41 161 314 15 9 27
Step 2: (42, 121) (13, 58) (7, 41) (161,314) (9, 15) (27)
Step 3: (13, 42, 58, 121) (7, 41, 161,314) (9, 15, 27)
Step 4: (7, 13, 41, 42, 58, 121, 161, 314) (9, 15, 27)
Step 5: (7, 9, 13, 15, 27, 41, 42, 58, 121, 16

Exercise with solution

Assuming that at a certain step of the bottom-up merge sort algorithm, the following 2 ordered sub-vectors have been obtained

(2, 6_1, 19, 35, 67_1, 89_1, 110, 230) and (1, 6_2, 6_3, 45, 67_2, 89_2, 89_3, 89_4, 95)

merge them.

Solution

(1, 2, 6_1, 6_2, 6_3, 19, 35, 45, 67_1, 67_2, 89_1, 89_2, 89_3, 89_4, 95,110, 230)

Exercise in Exam platform mode

Text:
Sort the following vector A of integers in ascending order by bottom-up merge sort:

10 54_1 28 92 17 55 54_2 19

Questions and format of answer

- display as a sequence of integers the contents of the sorted subarrays of size 2 (assume the condition in Merge is <=)

  (10 54_1) (28 92) (17 55) (19 54_2)

- display as a sequence of integers the contents of the sorted subarrays of size 4 (assume the condition in Merge is <=)

  (10 28 54_1 92) (17 19 54_2 55)

- display as a sequence of integers the final sorted array **assuming the condition in Merge is <**

  (10 17 19 28 54_2 54 $_1$ 55 92)

- The asymptotic worst-case complexity of the **Merge** function on N data is:
  - O(N)
  - O(NlogN)
  - $O(N^2)$

  Brief explanation of the answer: review the transparencies.