Algorithms and data structures 1

Theoretical exam

2nd night appointment

September 26, 2022

Surname:	
Student number:	

The information is printed on both sides!

universität wien

1	17	29	10	31	19	37	22
+	+	+	+	+	+	+	+

Task 1 [2]

In the table above, enter the digits of your student number in the empty boxes in front of which there is a plus sign.
Do the additions and find the numbers up to .

Task 2 [18]

The following functions are given

```
void done(int n) { for
    (int i=3; i<n; ++i)
        for (int j=0; j<i; ++j)
            for (int k=0; k<n*n; ++k);
}

void todo(int n) { for (int
    i=1; i<n; i=i*5) done(n);

}

void doing(int n, int digit) { if (n==0)
    return; done(n); for
    (int i=0;
    i<digit; ++i) { doing(n/2, digit); done(n);
        doing(n/2, digit)



    }
}
```

Find runtime estimates in theta notation (depending on n). For digit, use (%5)+2.

a. [4] the function done, b.
[4] the function todo, c.
[10] and the doing function

Task 3 [20]

The values up to . (from task 1) are stored in an array in this order from left to right. Sort the values in ascending order

a) [10] Quicksort

b) [6] Counting Sort – each number **modulo 10** (e.g. %10, %10, etc.)

c) [4] Merge sort

Specify all the necessary steps in sufficient detail to make it clear how the algorithm works.

Task 4 [20]

a) [8] Insert the values up to from task 1 (in this order) into an initially empty max heap. (Values may be stored multiple times in the heap.) Sketch the state of the heap after each insertion step.

b) [3] In notation similar to C++, give the definition of the most efficient data structure for a heap.

c) [6] In notation similar to C++, give a definition of a function that is as efficient as possible that determines and returns the depth of the heap in the tree representation. Determine the runtime complexity of your function depending on the number (n) of values stored in the search tree. To do this, use Big O notation.

d) [3] In notation similar to C++, give a definition of a most efficient function that determines whether it is a min-heap or a max-heap. The function should return *true* if it is a max heap, *false* otherwise.

Task 5 [20]

a) [10] Add the numbers 7, 0, 4, 21, 14 and (from Example 1) in that order into an originally empty hash table size 7.

Use the hash function ( ) = % and for collision handling double hashing with ( ) = % + Specify the state of the hash table after each insert operation.

b) [2] Delete the number 21 from the hash table (specify which positions of the hash table you need to visit in order to to find the number to be deleted and the state of the hash table after deletion).

c) [6] Find (after deleting) the numbers 7, 14 and 21 in the hash table (specify which positions for each number the hashtable you need to visit and whether the search is successful or not).

d) [2] What is the fundamental difference between static and dynamic hashing methods?

Task 6 [20]

Given is the following adjacency matrix, which describes the costs of the connections between the nodes of a directed graph (use the numbers up to from problem 1):

0 z4 0 z3 z2
0 0 z5 z8 0
1 0 5 0 1

0 0 z6 0 z7
(41 0 2 0 9)

a) [2] Sketch the graph described by this adjacency matrix.
b) [10] Use Dijkstra's algorithm to determine the shortest paths from node 1 to all other nodes of the Graph (where node 1 corresponds to the node of the first row/column in the adjacency matrix).
c) [8] Remove as small a number of edges as possible so that the graph becomes topologically sortable (state exactly which edges need to be removed) and perform a topological sort (it is enough to specify the sequence of nodes, you do not have to draw the graph for the result).