Exam Preparation

Exercise 1 *Integer Multiplication*

1. Describe the recursive school multiplication algorithm for multiplying two n-digit numbers. Develop the recursive formula for the number of primitive operations.
2. Describe the Karatsuba multiplication algorithm for multiplying two n-digit numbers. Develop the recursive formula for the number of primitive operations. Why is the Karatsuba multiplication asymptotically faster than the school multiplication?

Exercise 2 *Master Theorem*

- Revisit the master theorem and its proof.
- Consider example recursive formulas and apply the master theorem to obtain a tight asymptotic bound.

Exercise 3 *Invariants*

- What are invariants, pre-conditions, and post-conditions?
- How do you use them to show that a program is correct?
- Show the correctness of an example program from the course using invariants.

Exercise 4 *Skip Lists*

- Describe the basic properties of a Skip List.
- What is the probability that the Skip Lists exceeds a certain height h after the insertion of n elements?
- Show by example how find, insertion, and deleting works for Skip Lists.

Exercise 5 *Hashing*

- Describe the different hashing mechanisms given in the lecture and compare them.
- Consider an example where hashing has to deal with collisions. How are collisions resolved by the different hashing approaches?
- Prove the upper bounds on the expected execution times (assuming random hash functions) for the different hashing approaches discussed in the lecture.
- What does it mean that a class of hash functions is called c-universal?

- Define a class of 1-universal hash functions.
- Choose a random hash function from a class of 1-universal hash functions efficiently.

Exercise 6 *Graph Algorithms*

- Give the algorithms DFS and BFS and show how they work on an example graph. What are the running times of these algorithms.
- Give an algorithm that computes the strongly connected components for a given directed graph. What is the running time?

Exercise 7 *Shortest Paths*

- Give an algorithm that solves the single-source-shortest path problem for a given weighted graph where the edge weights are positive.
- Show the execution of this algorithm on a example graph.
- Give the Floyd-Warshall algorithm for solving the all-pairs-shortest-path problem with non-negative edge weights.

Exercise 8 *Minimum Spanning Trees*

- Give Kruskal's algorithm for the computation of a minimum spanning tree and show the execution of this algorithm on an example graph.
- Analyze the runtime of Kruskal's algorithm and show how union and find operations can be supported efficiently.
- Give the Jarník-Prim algorithm for the computation of a minimum spanning tree and show the execution of this algorithm on an example graph.

Exercise 9 *Turing machines*

- Define a deterministic and a nondeterministic Turing machine. How do they differ?
- Define the Diagonalization language. Give a proof that it's not decidable.
- What is the Halting problem? What are the implications of it being undecidable?
- Define the classes P, NP, co-NP BPP, RP, co-RP, ZPP, PSPACE via Turing machines. How do these classes differ with respect to the problems they contain? For each subset relationship describe why one complexity class is contained in the other.

Exercise 10 *P and NP*

- Characterize the classes P and NP in an informal way. How do you show that a problem is in P or in NP?

- What does it mean that a problem is NP-hard?
- How do you prove that a problem is NP-complete?
- Show that the Traveling Salesman Problem is NP-complete. You can assume that the Hamiltonian Cycle problem is NP-complete.
- Give three other examples of NP-complete problems.