Exercises for Algorithms and Data Structures Lecture 1

Question 1. Given a chess board of size 8×8, of which a random white square and a random black square are removed. Can this board be filled with dominoes? If yes, sketch a procedure how. If not, give a counter example.

Question 2. Locker doors (Levitin: exercise 1.1.12)

There are n lockers in a hallway, numbered sequentially from 1 to n. Initially, all the locker doors are closed. You make n passes by the lockers, each time starting with locker #1. On the ith pass, $i = 1, 2, \ldots, n$, you toggle the door of every $i$ith locker: if the door is closed, you open it; if it is open, you close it.

For example, on the first pass, you toggle the doors of all lockers. On the second pass, you toggle the doors of lockers 2, 4, 6, 8, etc. On the third pass, you toggle the doors of lockers 3, 6, 9, 12, etc.

After the last pass, which locker doors are open and which are closed? How many of them are open?

Also proof that your answer (which lockers are open at the end?) is correct.

Question 3. Worst/best/average case (Levitin: exercise 2.1.4)

a. Glove selection

There are 22 gloves in a drawer: 5 pairs of red gloves, 4 pairs of yellow, and 2 pairs of green. You select the gloves in the dark and can check them only after a selection has been made. What is the smallest number of gloves you need to select to have at least one matching pair in the best case? In the worst case?

A matching pair is one left glove and one right glove of the same colour. Note that left gloves and right gloves are different.

b. Missing socks.

Imagine that after washing 5 distinct pairs of socks, you discover that two socks are missing. Of course, you would like to have the largest number of complete pairs remaining. Thus, you are left with 4 complete pairs in the best-case scenario and with 3 complete pairs in the worst case. Assuming that the probability of disappearance for each of the 10 socks is the same, find the probability of the best-case scenario; the probability of the worst-case scenario; the number of pairs you should expect in the average case.

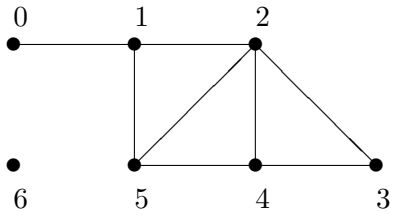
Question 4. Given n coins, of which one is counterfeit. They all look exactly the same. All coins have the same weight, except the counterfeit coin. We want to detect the counterfeit coin, and in order to do so, we have a balance scale (see figure).

a. One possible solution is to weight the coins one by one (one on the left scale, one on the right scale) until we found the counterfeit coin. What is the minimal number of weighting operations? And what is the maximal number of weighting operations?

b. Assume for part **b.** and **c.** that we know that the counterfeit coin weights more than the others. Describe a more efficient way to detect the counterfeit coin than the procedure in **a.** What is the minimal/maximal weighting operations neccessary? How much is that for $n = 24$?

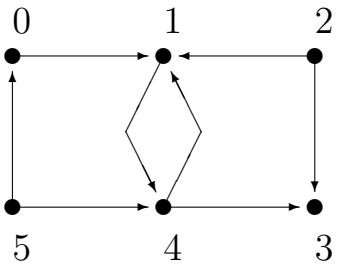
c. Show for $n = 24$ that the counterfeit coin can be found in at most 3 weighting operations.

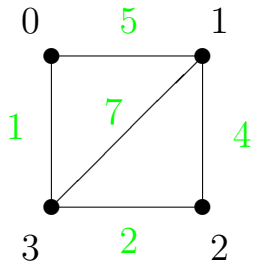
Question 5. Determine the adjacency matrix and the adjacency list for the following undirected graph.



For the following excersises, we use the adjacency matrix and adjacency list representation of graphs from the lecture (see slides). The adjacency matrix can be represented as a 2D numpy array, and the adjacency list can be represented as a dictionary, mapping from a key (in this case an integer) to a list (in this case a list of integers).

Question 6. Show how these data structures need to be used, initiated or adapted in order to facilitate weighted grapghs and directed grapghs. Write down the adjacency matrix and adjacency list for the example graphs **2.** and **3.** from the lecture (below).





Question 7. Describe an algorithm that determines for a given undirected graph whether it contains at most two vertices with an odd number of neighbours. Use the adjacency list representation.

Question 8. **a.** Describe an algorithm that determines the total number of edges in a given undirected graph. Use the adjacency list representation.
b. Idem **a.**, but use adjacency matrix representation.
c. Which representation is most appropriate (efficiënt)?

Question 9. Given a directed grapgh, with an edge from vertex i to vertex j, but not the other way around.
a. Give an algorithm that inverts this edge. Use the adjacency matrix.
b. Idem, using the adjacency list.

Question 10. Optional (gcd and common graph puzzles): excersise 1.1.6, 1.3.4, 1.3.5 and 1.3.8 (Levitin)