List of Abbreviations and Symbols

$A[1..n]$	An array indexed from 1 to n of n elements.
$\mathbb{N}$	Set of all natural numbers, i.e., $\{1, 2, 3, \dots\}$.
$\mathbb{R}$	Set of all real numbers.
$\mathbb{Z}$	Set of all integers, i.e., $\{\dots, -2, -1, 0, 1, 2, \dots\}$.

Modifiers

To help you with what problems to try, problems marked with **[K]** are key questions that tests you on the core concepts, please do them first. Problems marked with **[H]** are harder problems that we recommend you to try after you complete all other questions (or perhaps you prefer a challenge). Good luck!!!

Contents

§1 Optimal selection

Exercise 1.1. [K] You have n items for sale, numbered from 1 to n. Alice is willing to pay $a[i] > 0$ dollars for item i, and Bob is willing to pay $b[i] > 0$ dollars for item i. Alice is willing to buy no more than A of your items, and Bob is willing to buy no more than B of your items. Additionally, you must sell each item to either Alice or Bob, but not both, so you may assume that $n \le A + B$. Given $n, A, B, a[1..n]$ and $b[1..n]$, you have to determine the **maximum** total amount of money you can earn in $O(n \log n)$ time.

Exercise 1.2. [K] (Leetcode reference) Given an array A of integers of length n, n vertical lines are drawn such that the two end points of the ith line are $(i, 0)$ and $(i, A[i])$. A container is formed using two of the n vertical lines and the x axis. Design an $O(n)$ algorithm returns the maximum area of water that the container can store.

Exercise 1.3. [K] Assume that you have an unlimited number of \$2, \$1, 50c, 20c, 10c and 5c coins to pay for your lunch. Design an algorithm that, given the cost that is a multiple of 5c, makes that amount using a minimal number of coins.

Exercise 1.4. [K] Assume that the denominations of your $n + 1$ coins are $1, c, c^2, c^3, \dots, c^n$ for some integer $c > 1$. Design a greedy algorithm that runs in linear time complexity for which, given any amount, makes that amount using a minimal number of coins.

Exercise 1.5. [K] Given two sequences of letters A and B, find if B is a sub-sequence of A in the sense that one can delete some letters from A and obtain the sequence B.

Exercise 1.6. [K] There are N towns situated along a straight line. The location of the i'th town is given by the distance of that town from the westernmost town, d_i (so the location of the westernmost town is 0). Police stations are to be built along the line such that the i'th town has a police station within A_i kilometers of it. Design an algorithm to determine the minimum number of police stations that would need to be built.

Exercise 1.7. [H] (LeetCode reference, Patching Array) Given a sorted integer array A and an integer n, add elements to the array such that any number between $[1, n]$ (inclusive) can be formed by the sum of some elements in the array. Design an $O(n)$ algorithm to determine the minimum number of elements to add to A.

Exercise 1.8. [H] Let A be a $2 \times n$ array of positive real numbers such that the elements in each column sum to 1. Design an $O(n \log n)$ algorithm to pick one number in each column such that the sum of the numbers chosen in each row never exceeds $\frac{n+1}{4}$.

For example, consider the following 2×8 array.

0.4	0.7	0.9	0.2	0.6	0.4	0.3	0.1
0.6	0.3	0.1	0.8	0.4	0.6	0.7	0.9

We can choose the following configuration (the configuration is highlighted in red).

0.4	0.7	0.9	**0.2**	**0.6**	**0.4**	**0.3**	**0.1**
0.6	**0.3**	**0.1**	0.8	0.4	0.6	0.7	0.9

The chosen numbers on the first row give us

$$0.4 + 0.2 + 0.6 + 0.4 + 0.3 + 0.1 = 2 < 2.25 = \frac{9}{4},$$

while the chosen numbers on the second row give us

$$0.3 + 0.1 = 0.4 < 2.25 = \frac{9}{4}.$$

Exercise 1.9. [K] (When does Greedy fail?) Consider the following problem.

> *Given a set of denominations and a a value V, find the minimum number of coins that add to give V.*

Consider the following greedy algorithm.

Algorithm: Pick the largest denomination coin which is not greater than the remaining amount. Since we always choose the largest amount on each iteration, we minimise the number of coins required.

Provide a counter example to the algorithm to show that greedy does not always yield an optimal solution.

§2 Optimal ordering

Exercise 2.1. [K] There are N robbers who have stolen N items. You would like to distribute the items among the robbers (one item per robber). You know the precise value of each item. Each robber has a particular range of values they would like their item to be worth (too cheap and they will not have made money, too expensive and they will draw a lot of attention). Devise an algorithm that runs in $O(N^2)$ which either produces the optimal distribution such that every robber is happy or determines that there is no such distribution.

Exercise 2.2. [K] After the success of your latest research project in mythical DNA, you have gained the attention of a most diabolical creature: Medusa. Medusa has snakes instead of hair. Each of her snakes' DNA is represented by an uppercase string of letters. Each letter is one of S, N, A, K or E. Your extensive research shows that a snake's venom level depends on its DNA. A snake has venom level x if its DNA:

- has exactly $5x$ letters
- begins with x copies of S
- then has x copies of N
- then has x copies of A
- then has x copies of K
- ends with x copies of E.

For example, a snake with venom level 1 has DNA SNAKE, while a snake that has venom level 3 has DNA SSSNNNAAAKKKEEE. If a snake's DNA does not fit the format described above, it has a venom level of 0. Medusa would like your help making her snakes venomous, by deleting zero or more letters from their DNA. Given a snake's DNA of length n, design an $O(n \log n)$ algorithm to determine the maximum venom level the snake could have.

> **Hint** — *Combine binary search with greedy.*

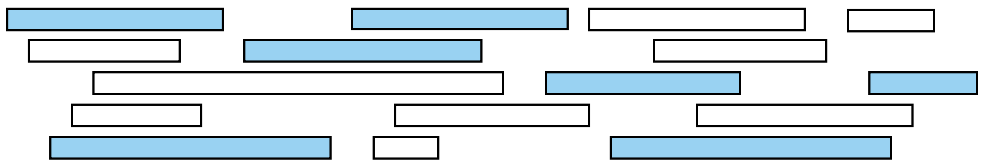
Exercise 2.3. [H] Assume that you got a fabulous job and you wish to repay your student loan as quickly as possible. Unfortunately, the bank *Western Road Robbery* which gave you the loan has the condition that you must start your monthly repayments by paying off \$1 and then each subsequent month you must pay either double the amount you paid the previous month, the same amount as the previous month or half the amount you paid the previous month. On top of these conditions, your last payment must be \$1. Design an algorithm that, given a positive integer S as your loan amount, produces a payment strategy such that minimises the number of months it will take you to repay your loan while satisfying all of the bank's requirements.

Exercise 2.4. [K] There are N people that you need to guide through a difficult mountain pass. Each person has a speed in days that it will take to guide them through the pass. You will guide people in groups of up to K people, but you can only move as fast as the slowest person in each group. Design an algorithm to determine the fewest number of days you will need to guide everyone through the pass.

§3 Scheduling

Exercise 3.1. [K] There are N courses that you can take. Each course has a minimum required IQ, and will raise your IQ by a certain amount when you take it. Design an algorithm to find the minimum number of courses you will need to take to get an IQ of at least K.

Exercise 3.2. [K] Let X be a set of n intervals on the real line. A subset of intervals $Y \subseteq X$ is called a tiling path if the intervals in Y cover the intervals in X, that is, any real value that is contained in some interval in X is also contained in some interval in Y. The size of a tiling cover is just the number of intervals. Describe and analyse an algorithm to compute the smallest tiling path of X as quickly as possible. Assume that your input consists of two arrays $X_L[1..n]$ and $X_R[1..n]$, representing the left and right endpoints of the intervals in X.



A set of intervals. The seven shaded intervals form a tiling path.

Exercise 3.3. [K] A photocopying service with a single large photocopying machine faces the following scheduling problem. Each morning they get a set of jobs from customers. They want to schedule the jobs on their single machine in an order that keeps their customers the happiest. Customer i's job will take t_i time to complete. Given a schedule (i.e., an ordering of the jobs), let C_i denote the finishing time of job i. For example, if job i is the first to be done we would have $C_i = t_i$, and if job j is done right after job i, we would have $C_j = C_i + t_j$. Each customer i also has a given weight w_i which represents his or her importance to the business. The happiness of customer i is expected to be dependent on the finishing time of their job. So the company decides that they want to order the jobs to minimise the weighted sum of the completion times, $\sum_{i=0}^{n} w_i C_i$. Design an efficient algorithm to solve this problem. That is, you are given a set of n jobs with a processing time t_i and a weight w_i for job i. You want to order the jobs so as to minimise the weighted sum of the completion times, $\sum_{i=0}^{n} w_i C_i$.

Exercise 3.4. [K] There is a line of 111 stalls, some of which lack a roof and need to be covered with boards. You can use up to 11 boards, each of which may cover any number of consecutive stalls. Cover all the necessary stalls, while covering as few total stalls as possible.

Exercise 3.5. [K] You have N students with varying skill levels and N jobs with varying skill requirements. You want to assign a different job to each student, but only if the student meets the job's skill requirement. Design an algorithm to determine the maximum number of jobs that you can successfully assign.

Exercise 3.6. [K] You have a processor that can operate 24 hours a day, every day. People submit requests to run daily jobs on the processor. Each such job comes with a start time and an end time; if a job is accepted, it must run continuously, every day, for the period between its start and end times. (Note that certain jobs can begin before midnight and end after midnight; this makes for a type of situation different from what we saw in the activity selection problem.)

(a) Given a list of n such jobs, your goal is to accept as many jobs as possible (regardless of their length), subject to the constraint that the processor can run at most one job at any given point in time. Design an $O(n^2)$ algorithm to do this with a running time that is polynomial in n. You may assume for simplicity that no two jobs have the same start or end times.

(b) Suppose that now for each of the n jobs given, an inspector is required to check the operation of the processor at any point during its time of operation at least once, however, you do not know which subset of the jobs will be chosen. Therefore, design an algorithm that finds the minimal and valid list of time stamps of the inspector can come and check the operation. You may assume that each inspection can be done instantaneously.

Exercise 3.7. [K] You are given a set of n jobs where each job i has a deadline $d_i \geq 1$ and profit $p_i > 0$. Only one job can be scheduled at a time. Each job takes 1 unit of time to complete. We earn the profit if and only if the job is completed by its deadline. Design an $O(n^2)$ algorithm to find the subset of jobs that maximises the total profit.

Exercise 3.8. [H] You are given a set X of n disjoint intervals $I_1, I_2, \ldots, I_n$ on the real line and a number $k \leq n$. Design an algorithm that produces a set Y of at most k intervals $J_1, J_2, \ldots, J_k$ which cover all intervals from X, hence

$$\bigcup_{i=1}^{n} I_i \subseteq \bigcup_{j=1}^{k} J_j$$

and such that the total length of all intervals in Y is minimal. Note that we do not require that $Y \subseteq X$, i.e., intervals J_j can be new. For example, if your set X consists of intervals $[1,2]$, $[3,4]$, $[8,9]$, and $[10,12]$ and if $k = 2$, then you should choose intervals $[1,4]$ and $[8,12]$ of total length $3 + 4 = 7$.

§4 Graph algorithms

Exercise 4.1. [K] Assume that you are given a complete graph G with non-negatively weighted edges such that all weights are distinct. We now obtain another complete weighted graph G' by replacing all weights $w(i,j)$ of edges $e(i,j)$ with new weights $w(i,j)^2$.

(a) Assume that T is the minimal spanning tree of G. Does T necessarily remain the minimal spanning tree for the new graph G'?

(b) Assume that p is the shortest path from a vertex u to a vertex v in G. Does p necessarily remain the shortest path from u to v in the new graph G'?

Exercise 4.2. [K] Consider a complete binary tree with $n = 2^k$ leaves. Each edge has an associated positive number that we call the length of the edge (see figure below). The distance from the root to a leaf is the sum of the lengths of all edges from the root to the leaf. The root emits a clock signal and the signal propagates along all edges and reaches each leaf in time proportional to the distance from the root to that leaf. Design an $O(\log n)$ algorithm which increases the lengths of some of the edges in the tree in a way that ensures that the signal reaches all the leaves at the same time while the total sum of the lengths of all edges is minimal.

For example, in the picture below if the tree A is transformed into trees B and C all leaves of B and C have a distance of 5 from the root and thus receive the clock signal at the same time, but the sum of the lengths of the edges in C is 17 while sum of the lengths of the edges in B is only 15.

Exercise 4.3. [K] Assume that you are given a complete weighted graph G with n vertices $v_1, \ldots, v_n$ and with the weights of all edges distinct and positive. Assume that you are also given the minimum spanning tree T for G. You are now given a new vertex v_{n+1} and the weights $w(n+1, j)$ of all new edges $e(n+1, j)$ between the new vertex v_{n+1} and all old vertices $v_j \in G$, $1 \leq j \leq n$. Design an algorithm which produces a minimum spanning tree T' for the new graph containing the additional vertex v_{n+1} and which runs in time $O(n \log n)$.

Exercise 4.4. [K] Assume that you are given a complete undirected graph $G = (V, E)$ with edge weights $\{w(e) : e \in E\}$ and a proper subset of vertices $U \subset V$. Design an algorithm that produces the lightest spanning tree of G in which all nodes of U are leaves (there might be other leaves in this tree as well).

Exercise 4.5. [H] You are given a connected graph with weighted edges with all weights distinct. Prove that such a graph has a unique minimum spanning tree.

Exercise 4.6. [H] Assume that you are given n white and n black dots lying in a random configuration on a straight line, equally spaced. Design a greedy algorithm which connects each black dot with a (different) white dot, so that the total length of wires used to form such connected pairs is minimal. The length of wire used to connect two dots is equal to the straight line distance between them.