1 Problems from old MIT 6.046 quizzes

All problems from `http://courses.csail.mit.edu/6.046/`. Detailed solutions can be found at the link as well.

Exercise (**Image Filtering [2005 Spring Quiz 2]**). *Two-dimensional filtering is a common operation in vision and image processing. An image is represented as an $n \times n$ matrix of real values. As shown in Figure 2, the idea is to pass a $k \times k$ window across the matrix, and for each of the possible placements of the window, the filter computes the product of all the values in the window. The product is not typically ordinary multiplication, however. For this problem, we shall assume it is an associative and commutative binary operation $\otimes$ with identity element e, that is, $x \otimes e = e \otimes x = x$. For example, the product could be $+$ with identity element 0, $\times$ with 1, $\min$ with ∞, etc. Importantly, you may not assume that $\otimes$ has an inverse operation, such as $-$ for $+$.*

To be precise, given an $n \times n$ image

$$A = \begin{pmatrix} a_{00} & a_{01} & \cdots & a_{0(n-1)} \\ a_{10} & a_{11} & \cdots & a_{1(n-1)} \\ \vdots & \vdots & \ddots & \vdots \\ a_{(n-1)0} & a_{(n-1)1} & \cdots & a_{(n-1)(n-1)} \end{pmatrix}$$

the $k \times k$-filtered image is the $n \times n$ matrix

$$B = \begin{pmatrix} b_{00} & b_{01} & \cdots & b_{0(n-1)} \\ b_{10} & b_{11} & \cdots & b_{1(n-1)} \\ \vdots & \vdots & \ddots & \vdots \\ b_{(n-1)0} & b_{(n-1)1} & \cdots & b_{(n-1)(n-1)} \end{pmatrix}$$

where for $i, j = 0, 1, \ldots, n-1$,

$$b_{ij} = \bigotimes_{x=i}^{i+k-1} \bigotimes_{j=y}^{j+k-1} a_{xy}$$

(For convenience, if $x \leq n$ or $y \leq n$, we assume that $a_{xy} = e$.) Give an efficient algorithm to compute the (kk)-filter of an input matrix A. While analyzing your algorithm, do not treat k as a constant. That is, express your running time in terms of n and k.

Exercise (**Hacking Skip Lists (2008 Fall Quiz 2)**). *After years of painstaking effort, you launch myferretplanetspace.com, an e-commerce site devoted to selling ferrets online. Your list of active customer orders is implemented as a skip list, with all data about an order stored as an element in the skip list. Every time a customer asks about the status of their order, your site queries the customer order list and displays the data. Every order gets a unique ID number that is used to place it in the customer order list. ID numbers are assigned by your web site and are not necessarily increasing, so the most recent order could go anywhere in the skip list. For a while, your site works well; orders can be managed quickly and customers are happy.*

But the ferret e-commerce industry is a cutthroat one, and one dark and stormy night your competitors try to slow down your site. There are two operations they can do: add a dummy order to your site, and cancel an order that they added.

(a) Your system is storing information about n legitimate orders. Your competitors add m dummy orders to your system. What effect will this have on the runtime of query, insert, and delete operations for legitimate customer orders the morning after your competitors try to slow your site? Assume that $m \gg n$.

(b) Your competitors find a security loophole in your site that lets them see information about your skip list: specifically, which level each order has been promoted to. They again wish to perform m operations, with the goal of slowing down later legitimate customer orders as much as possible. What should they do? What effect will this have on the runtime of query, insert, and delete operations for legitimate customer orders the morning after?

Exercise (Graph Sparsification (2008 Fall Quiz 2)). *You are given a connected graph $G = (V, E)$ such that each edge $e \in E$ has a distinct non-negative value $w(e)$. In order to reduce the cost of storing G, we want to remove edges from G such that the number of remaining edges is exactly equal to the number of nodes. However, there are two constrains on which edges can be removed and which edges can be kept. First, each vertex chooses an edge incident to it to keep. An edge will only be kept when it is chosen by some vertex. Any edge that is not chosen by any vertex will be removed. Second, the set of remaining edges must keep G connected, i.e. for any pair of vertices u and v, there is a path from u to v only using the chosen edges.*

Give an efficient algorithm to find which edge should be chosen by which vertex such that the total value of the remaining edges is maximized.

Exercise (Detecting Compatible Subtrees (2008 Fall Quiz 2)). *Given two rooted trees we say that S is a subtree of T if there is a one-to-one mapping f from the vertices of S to the vertices of T that preserves all parent-child relationships. In this problem, we consider rooted trees with the additional feature that every vertex is assigned a label from some set L (for example, the labels may come from the set of colors $L = \{$Red, Orange, Yellow, Green, Blue, Purple$\}$). Moreover, we are told that only certain pairs of labels are compatible. The set of compatible labels can be represented as an undirected graph H, called a compatibility graph, with vertex set L and edges connecting labels that are compatible.*

Given two rooted trees S and T, both with labels in L, we would like to determine whether S is a subtree of T, but with the additional constraint that each vertex of S must be matched with a compatibly labeled vertex of T. Formally, if there is a function f with all of the properties in the previous paragraph, and the additional property that the labels of v and $f(v)$ are compatible for all $v \in S$, we say S is a compatibly labeled subtree of T.

(a) Suppose you are given two labeled rooted trees, S and T , along with a compatibility graph H. Let $|S| = m$ and $|T| = n$, where $m \leq n$. Further suppose that both S and T have depth 1. Give an efficient algorithm to determine if S is a compatibly labeled subtree of T . You may assume that for vertices $u \in S$ and $v \in T$, it only takes $O(1)$ time to check if the labels of u and v are compatible.

(b) Same as part (a), but now let S and T be arbitrary (not necessarily depth 1) labeled rooted trees.

Exercise (Scheduling [2010 Fall Quiz 2]). *You are advising a high-school science team that is working on a large number of projects simul- taneously. Your team has a set of n projects and a set of m physical tools. Each project specifies a set of tools that it needs, and can only be completed if all those tools are available for use on this project. Each tool might be required by many projects, but naturally can only be*

used for at most a single project. Your job is to distribute tools to projects so as to complete as many projects as possible. More precisely, given a set of projects, a set of tools, a description of the set of tools necessary for each project, and an integer k, you would like to know if it is possible to distribute tools to projects so that at least k projects can be completed.

Consider the following variants and restrictions of the general problem described above. For each one state whether the variant problem is solvable in polynomial time or whether it is NP-hard. Justify your answer by either describing a polynomial-time algorithm or sketching a proof of the problems NP-hardness.

(a) The problem when k is constant.

(b) The special case of the problem when there are two types of tools (say, power tools and computers), and each project requires at most one tool of each type (perhaps one project requires the power drill and the iPad, and another requires the circular saw and the Thinkpad).

(c) The special case (of the original problem) when each tool is needed by at most two projects (but each project may need many tools).

Exercise (**Spy Games [2010 Fall Quiz 2]**). *You are an enemy agent inside the borders of the country of Elbonia.*

Your assignment is to make it as difficult as possible for the Elbonian rulers to send couriers on horseback from their capital city of Anar to its distant outpost of Chy along the countrys network of roads.

Each road r in Elbonia connects two cities, and has an associated time t_r (in minutes) needed to travel it by horseback. A courier will always take the fastest route from Anar to Chy. You have a budget of B dollars to hire workers who can covertly degrade the quality of the roads at night (by digging potholes, etc.). Spending y_r dollars on any road r will increase the time needed to travel it by y_r minutes. Given your complete knowledge of Elbonian roads (including their travel times) and your budget B, how can you efficiently calculate a way to allocate your money so that the fastest route from Anar to Chy will take as much time as possible?

Exercise (**Train Sabotage [2010 Fall Quiz 2]**). *You are a security consultant for a railroad company that manages one very long railroad line. There are n bridges on this line, spaced one mile apart. You are at bridge k_0 and have just received reliable intelligence that a terrorist organization has planted time bombs on all n bridges, scheduled to go off at times $(t_1, \ldots, t_n)$, given in minutes (with the current time being 0). Because of your military background, you can disable a bomb instantaneously (it takes no time). You have a train that travels at one mile per minute.*

(a) Give an efficient dynamic programming algorithm for finding whether there is a travel schedule to all of the bridges that allows you to disable all the bombs before any of them go off (to be precise, assume that if you arrive at a bridge the very instant it is due to go off, it blows up).

(b) Can your method from part (a) be easily modified to handle the variant where the bridges are separated by variable distances? If yes, give a sketch of the required modification. If no, explain why not.

Exercise (**Attacking Rooks/Knights [2010 Fall Quiz 2]**). *(a) You are given an $n \times n$ chessboard containing k rooks, where there is at most one rook per square. Give an efficient algorithm to find the largest possible subset of the k rooks such that no rook attacks any other rook. (A rook is said to attack another rook if the two rooks are in the same row or in the same column.)*

(b) Replace rooks with knights in the above problem. (A knight is said to attack another knight if their row numbers differ by 1 and their column numbers differ by 2, or if their row numbers differ by 2 and their column numbers differ by 1.) You may use the following theorem:

Theorem 1.
(Knigs Theorem) In any bipartite graph $G = (V, E)$, the size of a maximum matching in G is equal to the size of a minimum vertex cover of G. Moreover, given a maximum matching, we can find a minimum vertex cover in $O(V + E)$ time.

Hint: maximum independent set reduces to minimum vertex cover on the same graph.

Exercise (**Minimum Spanning Trees and Matchings [2011 Fall Quiz 2]**). *In all four parts of this problem, $G = (V, E, w)$ is a connected, weighted, undirected graph. We define $w(e)$ to be the weight of edge e.*

(a) Suppose that T is a minimum spanning tree of G. If we increase the weight of each edge of G by the same positive amount δ, is T still guaranteed to be a minimum spanning tree? Give an argument for why it will be, or a counterexample showing that it may not be.

(b) Let e be an edge of maximal weight in G. Suppose we add a single new edge e' to G to obtain the graph $G' = (V, E \cup \{e'\}, w)$. The weight function w is extended so that $w(e')$ is defined, and $w(e')$ is less than $w(e)$ - that is, the new edge does not have maximal weight. Show that the total weight of a minimum spanning tree of G' is at least $w(e') - w(e)$ plus the total weight of a minimum spanning tree of G.

(c) We now consider maximum-weight matchings in G. Suppose that we increase the weight of each edge of G by the same positive amount δ. Is it possible that a matching of maximum weight in G will no longer be maximum-weight after the change? Give an argument for why it will be, or a counterexample showing that it may not be.

(d) Finally, we consider maximum-cardinality matchings in G. A maximum-cardinality matching is a matching with the maximum possible number of edges. Suppose we add a single new edge e' to G to obtain the graph G' with edge set $E \cup \{e'\}$. Show that the size of a maximum-cardinality matching in G' is not more than one larger than the size of a maximum-cardinality matching in G.

Exercise (**Gerrymandering in LineLand [2011 Fall Quiz 2]**). *There are n residents of LineLand, residing at the points $1, 2, \ldots, n$ of the Line which is their world. There are two political parties in LineLand: the Cardinals and the Cyans. It is known to everyone if LineLander i is a Cardinal or a Cyan, for each $i, i = 1, 2, \ldots, n$.*

It is time to redistrict i.e. to divide LineLand into m political districts. Here m is significantly smaller than n.

Each district must be a contiguous segment of LineLand: it must include LineLander i up to LineLander j, inclusive, for some i and j, $i \leq j$.

Thus, the first district goes from 1 up to a_1, the second from $a_1 + 1$ up to a_2, and so on, until the last district is from $a_{m-1} + 1$ up to $a_m = n$.

To be fair, each district must have n/m LineLanders, more or less. More precisely, by the LineLand Constitution, a district must contain at least $r = \lceil n/(2m) \rceil$ LineLanders, and may not contain more than $s = \lfloor 3n/(2m) \rfloor$ LineLanders.

Once the districts are drawn, the LineLand Parliament is created by picking one resident uniformly at random from each district. (So the Parliament has size m.)

Explain what algorithm you would use to determine the districts (i.e., to determine $a_1, \ldots, a_m$), given n, the number m of districts desired, and the political affiliation of each LineLander, in such a way that would maximize the expected number of Cardinal members in Parliament. (Also explain how efficient your algorithm is, as a function of n and m.)

Exercise (**The Ivory Tower [2011 Fall Quiz 2]**). *A group of undergraduates, graduate students and professors work in an ivory tower. Each professor p is willing to work with a subset $U(p)$ of the undergrads and a subset $G(p)$ of the graduate students; a research team is comprised of an undergrad, a graduate student and a professor who is willing to work with both. No one can belong to more than one research team.*

Find an efficient algorithm for determining the maximum number of research teams that can be formed given the professors preferences.

Exercise (**Red-Blue Network Flow [2011 Fall Quiz 2]**). *You are given a flow network $G = (V, E)$ with distinguished source vertex s and distinguished sink vertex t. As usual, the edges in E are directed edges. The edges in E are now, however, of two types: blue and red. The blue edges are of the usual type: each blue edge e has a capacity $c(e)$ that is the maximum possible amount of flow that can pass through that edge. The red edges are built out of strange alien technology: each red edge e must have at least a certain amount $d(e)$ of flow passing through that edge. That is, $d(e)$ is the minimum required amount of flow that must pass through that edge.*

Other that the fact that some edges are now red, every thing is as usual (e.g. flow must be conserved at all intermediate vertices, etc.). Each edge is either red or blue. We call such a network problem a red-blue network flow problem. As usual, the goal is to find a flow of maximum possible value.

(a) Show that it is possible to compute a maximum red-blue network flow, or determine that no legal flow exists, in polynomial time.

(b) You now discover how to turn off red edges. If a red edge e is on, at least $d(e)$ units of flow must pass over it; if it is off, no flow can pass over it. You can turn some red edges on and others off. Show that it is now NP-complete to determine the maximum flow possible in a red-blue network. (Use the fact that the subset-sum *problem is NP-hard, where subset-sum is the decision problem that determines if there is a subset in $S = \{x_1, x_2, \ldots, x_n\}$ whose sum equals to k.)*

Exercise (**Recurrences [2010 Fall Quiz 1]**). *Solve the following recurrences by giving tight Θ-notation bounds. You do not need to justify your answers, but any justification that you provide will help when assigning partial credit. As usual, assume that $T(n) = O(1)$ for $n \leq 2$. For part (e), assume that $T(n, k) = O(1)$ for $n \leq 2$ or $k \leq 2$.*

(a) $T(n) = 4T(n/4) + \Theta(n \log n)$

(b) $T(n) = 5T(n/2) + \Theta(n^2 \log n)$

(c) $T(n) = 8T(n/2) + n \log n + 2n^3$

(d) $T(n) = 3T(n/4) + n\sqrt{n}$

(e) $T(n, k) = T(n/2, k) + T(n, k/4) + kn$

Exercise (**True or False, and Justify [2010 Fall Quiz 1]**). *Decide T or F for each of the following statements, and briefly explain why. Your justification is worth more points than your true-or-false designation.*

(a) A Monte Carlo algorithm that uses the sequence 010101010101... (alternating 0 and 1) as its randomness will always run in polynomial time.

(b) It is possible to devise a Las Vegas algorithm to check whether an $n \times n$ matrix M with binary entries has all 0 entries that runs in expected $O(n)$ time for any input.

(c) Searching in a randomized skip list of n elements never takes $\omega(n)$ time.

(d) Suppose we have computed a minimum spanning tree (MST) and its total weight for some graph G. If we make a new graph G' by adding 1 to the weight of every edge in G, we will need to spend $\Omega(|E|)$ time to compute an MST and its total weight for the new graph G'.

Exercise (**Short Answer [2010 Fall Quiz 1]**). *Give brief, but complete, answers to the following questions.*

(a) Consider the following two hash functions h_1 and h_2 from $\{1, 2, 3\}$ to $\{0, 1\}$:

	1	2	3
h_1	0	0	1
h_2	1	1	0

(i.e., $h_1(1) = 0, h_1(2) = 0, h_1(3) = 1, h_2(1) = 1, h_2(2) = 1, h_2(3) = 0$)

1. Is the set $H = \{h_1, h_2\}$ a universal hash family? Why or why not?

2. What if we modify h_1 so that $h_1(2) = 1$. Is $H = \{h_1, h_2\}$ a universal hash family now? Why or why not?

(b) Define a type 0 array as an array where $3/4$ of the elements are 0 and $1/4$ are 1. Define a type 1 array as an array where $1/4$ of the elements are 0 and $3/4$ are 1. You are given two arrays A and B of length n and are told that one is type 0 and one is type 1, but you do not know which is which.

Design a Monte Carlo algorithm that picks one element uniformly at random from each array and determines which array is which. What is the probability that your algorithm outputs the correct answer? (The probability that your algorithm outputs the correct answer should be greater than 1/2.)

Exercise (**Postering the Corridor [2010 Fall Quiz 1]**). *A certain university has a very long corridor with n boards (labeled 1 through n) where people can place posters. Due to university rules, you may put up as many posters advertising an event as you want, provided that:*

- *The same poster can appear only once per board, and*
- *The same poster cannot appear on two adjacent boards. That is, you are prohibited from placing a poster on both board i and board $i + 1$.*

(Assume that there is always space to put up a poster on any board.)

Through extensive spy-camera research, you know that each person only looks at one of the boards. Furthermore, you can predict that exactly b_i people will look at board i. Imagine you have an event you wish to publicize, and you want to select which boards to put posters on in order to maximize the number of viewers while obeying university rules. Thus, you want to design an algorithm that, given $\{b_1, b_2, \ldots, b_n\}$, outputs a set of board numbers S that maximizes $\sum_{i \in S} b_i$, subject to the constraint that if $i \in S$, then $(i + 1) \notin S$ and $(i - 1) \notin S$.

(a) Consider the following greedy algorithm for placing posters to advertise an event:

- *Define a still-available board as a board that you have not put a poster on and that is not adjacent to one you have put a poster on.*
- *While there are any still-available boards remaining: Select the still-available board with the greatest number of viewers, and put a poster on it.*

Give an example (i.e. a configuration of b_i values) that shows that this algorithm does not always give the arrangement that maximizes the number of viewers.

(b) Give an efficient algorithm that uses dynamic programming to find the maximum possible number of viewers of a poster that obeys the rules.

(c) In a few sentences, explain how you would modify your algorithm from part (b) to return not only the maximum number of viewers itself, but a configuration of boards obeying the rules that gives that maximum.

2 Problems from TopCoder

You need a (free) account to access certain parts of the TopCoder website. You can obtain one by clicking "Register" at the top right corner of `http://community.topcoder.com/tc`. TopCoder hosts frequent programming competitions with algorithmic problems of a similar flavor to the homework programming problems. For all their problems, they provide written solution descriptions in their match editorials. You can also view code from competitors who solved the problem correctly. To do so, go to Match Overviews: `http://community.topcoder.com/stat?c=round_overview`. Then under "Please select a round", select the round that the problem is from. Then you will see the highest-scoring individuals; click on the golden circle to the left of their name to see their code for that round. Then under "Problems" click the appropriate "Class Name" to see their solution for that problem (the Status in the second to last column will say "Passed System Test" if their solution was judged as correct and sufficiently fast).

Generally in TopCoder, the time limit per test cases is two seconds. Thus, when you see the maximum input size given in problem statements, you should take this consideration as a hint to which solutions would be fast enough and which wouldn't.

In general: problems on TopCoder are labeled as either "Easy", "Medium", or "Hard". They are also "Div 1" or "Div 2"; Div 1 is harder than Div 2. (In general, a Div 2 Hard problem is typically between a Div 1 Easy and Div 1 Medium.) Div 1 hard is much harder than would ever be on a CS 124 exam unless there were multiple steps laid out as hints (but some are listed below, and you can take a look if you want to try them out). A challenging CS 124 exam problem could be a Div 1 medium.

A list of all TopCoder problems ever, labeled by technique(s) required to solve them, can be found here: `http://community.topcoder.com/tc?module=ProblemArchive`.

Note: TopCoder problems essentially never require randomization. (Thus the problems below wouldn't test your knowledge of e.g. hashing, skip lists, etc.)

1. **Problem:** TCCC '03 Semifinals 3 Div 1 Easy — ZigZag

   Description: Problem description: `http://community.topcoder.com/stat?c=problem_statement&pm=1259&rd=4493`

   Match Editorial: `http://community.topcoder.com/tc?module=Static&d1=tournaments&d2=tccc03&d3=semiprob3`

2. SRM 210 Div 2 Hard — Topographical Image

   Description: `http://community.topcoder.com/stat?c=problem_statement&pm=2932&rd=5856`

Match Editorial: http://community.topcoder.com/tc?module=Static&d1=match_editorials&d2=srm210 (also see http://tinyurl.com/kxguchf)

3. **Problem:** TCCC '04 Round 4 Div 1 Easy — BadNeighbors

   Description: http://community.topcoder.com/stat?c=problem_statement&pm=2402&rd=5009

   Match Editorial: http://community.topcoder.com/tc?module=Static&d1=match_editorials&d2=tccc04_online_rd_4

4. **Problem:** TCCC '04 Round 1 Div 1 Medium — FlowerGarden

   Description: http://community.topcoder.com/stat?c=problem_statement&pm=1918&rd=5006

   Match Editorial: http://community.topcoder.com/tc?module=Static&d1=match_editorials&d2=tccc04_online_rd_1

5. TCO '04 Round 4 Div 1 Medium — HeatDeath

   Description: http://community.topcoder.com/stat?c=problem_statement&pm=2982&rd=5881

   Match Editorial: http://community.topcoder.com/tc?module=Static&d1=match_editorials&d2=tco04_online_rd_4

6. TCO '03 Semifinals 4 Div 1 Easy — AvoidRoads

   Description: http://community.topcoder.com/stat?c=problem_statement&pm=1889&rd=4709

   Match Editorial: http://community.topcoder.com/tc?module=Static&d1=tournaments&d2=tco03&d3=summary&d4=room4_analysis

7. SRM 356 Div 2 Hard — RoadReconstruction

   Description: http://community.topcoder.com/stat?c=problem_statement&pm=7921&rd=10765

   Match Editorial: http://community.topcoder.com/tc?module=Static&d1=match_editorials&d2=srm356 (also see http://tinyurl.com/kxguchf)

8. TCCC '03 Round 4 Div 1 Easy — ChessMetric

   Description: http://community.topcoder.com/stat?c=problem_statement&pm=1592&rd=4482

   Match Editorial: http://community.topcoder.com/tc?module=Static&d1=match_editorials&d2=tccc03_regfinal

9. TCO '03 Round 4 Div 1 Medium — Jewelry

   Description: http://community.topcoder.com/stat?c=problem_statement&pm=1166&rd=4705

   Match Editorial: http://community.topcoder.com/tc?module=Static&d1=match_editorials&d2=tco03_online_rd_4

10. SRM 150 Div 1 Medium — StripePainter

    Description: http://community.topcoder.com/stat?c=problem_statement&pm=1215&rd=4555

    Match Editorial: http://community.topcoder.com/tc?module=Static&d1=match_editorials&d2=srm150

11. SRM 197 Div 2 Hard — QuickSums

    Description: http://community.topcoder.com/stat?c=problem_statement&pm=2829&rd=5072

Match Editorial: http://community.topcoder.com/tc?module=Static&d1=match_editorials&d2=srm197

12. SRM 156 Div 1 Hard — PathFinding

13. **Description:** http://community.topcoder.com/stat?c=problem_statement&pm=1110

    Match Editorial: http://community.topcoder.com/tc?module=Static&d1=match_editorials&d2=srm156 (also see http://tinyurl.com/kxguchf)

14. SRM 165 Div 2 Hard — ShortPalindromes

    Description: http://community.topcoder.com/stat?c=problem_statement&pm=1861&rd=4630

    Match Editorial: http://community.topcoder.com/tc?module=Static&d1=match_editorials&d2=srm165

15. SRM 207 Div 1 Medium — TCSocks

    Description: http://community.topcoder.com/stat?c=problem_statement&pm=2894&rd=5853

    Match Editorial: http://community.topcoder.com/tc?module=Static&d1=match_editorials&d2=srm207

16. SRM 208 Div 1 Hard — StarAdventure

    Description: http://community.topcoder.com/stat?c=problem_statement&pm=2940&rd=5854

    Match Editorial: http://community.topcoder.com/tc?module=Static&d1=match_editorials&d2=srm208

17. SRM 178 Div 1 Hard — MiniPaint

    Description: http://community.topcoder.com/stat?c=problem_statement&pm=1996&rd=4710

    Match Editorial: http://community.topcoder.com/tc?module=Static&d1=match_editorials&d2=srm178

18. SRM 211 Div 1 Medium — GrafixMask

    Description: http://community.topcoder.com/stat?c=problem_statement&pm=2998&rd=5857

    Match Editorial: http://community.topcoder.com/tc?module=Static&d1=match_editorials&d2=srm211 (also see http://tinyurl.com/kxguchf)

19. SRM 224 Div 1 Medium — Rationalization

    Description: http://community.topcoder.com/stat?c=problem_statement&pm=2347&rd=5870

    Match Editorial: http://community.topcoder.com/tc?module=Static&d1=match_editorials&d2=srm224

3 Other resources

- Jeff Erickson has a free book on algorithms covering many topics from this course. He also posts all his old exam and homework problems online (caveat: he doesn't post the solutions!). One nice thing about his book, compared to say CLRS, is that it also covers some of the randomized algorithms from CS 124 (like skip lists, treaps, Karger's algorithm, and QuickSort). URL: http://web.engr.illinois.edu/~jeffe/teaching/algorithms/

- Codeforces is similar to TopCoder and also has many problems you can try out. As far as I can tell, match editorials aren't in any one single place, but you can google for match editorials for a particular round and usually find it. URL: `http://codeforces.ru/?locale=en` (again, you need an account for some features, but registration is free).