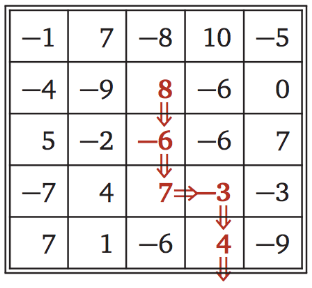
CS325H: Analysis of Algorithms, Fall 2020

Practice Problems 2

Problem 1. Vankin's Mile is an American solitaire game played on an n ? n square grid. The player starts by placing a token on any square of the grid. Then on each turn, the player moves the token either one square to the right or one square down. The game ends when player moves the token o the edge of the board. Each square of the grid has a numerical value, which could be positive, negative, or zero. The player starts with a score of zero; whenever the token lands on a square, the player adds its value to his score. The object of the game is to score as many points as possible. For example, given the grid below, the player can score $8 - 6 + 7 - 3 + 4 = 10$ points by placing the initial token on the in the second row, and then moving down, down, right, down, down. (This is not the best possible score for this grid of numbers.)



−1	7	−8	10	−5
−4	−9	8	−6	0
5	−2	−6	−6	7
−7	4	7	−3	−3
7	1	−6	4	−9

(a) Describe and analyze an ecient algorithm to compute the maximum possible score for a game of Vankin's Mile, given the $n \times n$ array of values as input.

(b) In the European version of this game, appropriately called Vankin?s Kilometer, the player can move the token either one square down, one square right, or one square left in each turn. However, to prevent infinite scores, the token cannot land on the same square more than once. Describe and analyze an ecient algorithm to compute the maximum possible score for a game of Vankin's Kilometer, given the $n \times n$ array of values as input.

Problem 2. A shuffle of two strings X and Y is formed by interspersing the characters into a new string, keeping the characters of X and Y in the same order. For example, the string BANANAANANAS is a shuffle of the strings BANANA and ANANAS in several different ways.

$$\mathrm{BANANA}_{\mathrm{ANANAS}} \qquad \mathrm{BAN}_{\mathrm{ANA}}{}^{\mathrm{ANA}}{}_{\mathrm{NAS}} \qquad \mathrm{B}_{\mathrm{AN}}{}^{\mathrm{AN}}{}_{\mathrm{A}}{}^{\mathrm{A}}{}_{\mathrm{NA}}{}^{\mathrm{NA}}{}_{\mathrm{S}}$$

Similarly, the strings PRODGYRNAMAMMIINCG and DYPRONGARMAMMICING are both shuffles of DYNAMIC and PROGRAMMING:

$\mathtt{PRO^{D}G^{Y}R^{NAM}AMMI^{I}N^{C}G}$ $\mathtt{^{DY}PRO^{N}G^{A}R^{M}AMM^{IC}ING}$

Given three strings $A[1..m]$, $B[1..n]$, and $C[1..m+n]$, describe an algorithm to determine whether C is a shuffle of A and B. Prove your algorithm is correct and analyze its running time.

Problem 3. Call a sequence $X[1 \cdots n]$ of numbers oscillating if $X[i] < X[i+1]$ for all even i, and $X[i] > X[i+1]$ for all odd i. Describe an efficient algorithm to compute the length of the longest oscillating subsequence of an arbitrary array A of integers.

Problem 4.

(a) Suppose we are given a set L of n line segments in the plane, where each segment has one endpoint on the line $y = 0$ and one endpoint on the line $y = 1$, and all $2n$ endpoints are distinct. Describe and analyze an algorithm to compute the largest subset of L in which no pair of segments intersects.

(b) Suppose we are given a set L of n line segments in the plane, where each segment has one endpoint on the line $y = 0$ and one endpoint on the line $y = 1$, and all $2n$ endpoints are distinct. Describe and analyze an algorithm to compute the largest subset of L in which every pair of segments intersects.

Problem 5. Suppose you are given an $m \times n$ bitmap as an array $M[1 \ldots n, 1 \ldots n]$ of 0's and 1's. A solid block in M is a subarray of the form $M[i \ldots i', j \ldots j']$ in which all bits are equal. A solid block is square if it has the same number of rows and columns.

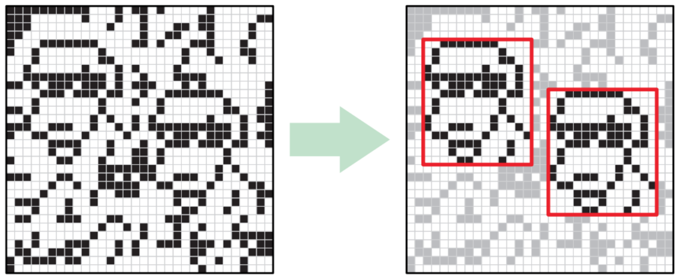
(a) Describe an algorithm to find the maximum area of a solid square block in M in $O(n^2)$ time.

(b) Describe an algorithm to find the maximum area of a solid block in M in $O(n^3)$ time.

(c) Describe an algorithm to find the maximum area of a solid block in M in $O(n^2 \log n)$ time. [Hint: Divide and conquer.]

(d) Describe an algorithm to find the maximum area of a solid block in M in $O(n^2)$ time.

Problem 6. Describe and analyze an algorithm that finds the maximum-area rectangular pattern that appears more than once in a given bitmap. Specifically, given a two-dimensional array $M[1 \ldots n, 1 \ldots n]$ of bits as input, your algorithm should output the area of the largest repeated rectangular pattern in M. For example, given the bitmap shown on the left in the figure below, your algorithm should return the integer 195, which is the area of the 15×13 doggo. (Although it doesn?t happen in this example, the two copies of the repeated pattern might overlap.)



(a) For full credit, describe an algorithm that runs in $O(n^5)$ time.

(b) For extra credit, describe an algorithm that runs in $O(n^4)$ time.

(c) For extra extra credit, describe an algorithm that runs in $O(n^3 polylogn)$ time.