Algorithms: Assignment Two

Deadline: 27 Feb 2019

1. Let $A[1...m]$ and $B[1...n]$ be two strings (character arrays). A common supersequence of A and B is a string that contains both A and B as subsequences. For example, the string $MATCHES$ is a supersequence of the two strings $MATE$ and $ACHES$. Describe an efficient algorithm to find the shortest common supersequence of A and B.

2. We call a sequence $X_1, X_2, \ldots$ of numbers as "accelerating" if for every $i \geq 2$ we have $X_{i+1} - X_i > X_i - X_{i-1}$. For example, the sequences $1, 3, 6$ and the sequence $1, -4, 0, 5$ are accelerating, whereas the sequences $1, 3, 5$ and $-5, 4, 0, 3$ are not accelerating. A sequence of length less than 3 is considered to be accelerating. Describe an efficient algorithm to find the longest accelerating subsequence of an array $A[1, 2, \ldots, n]$.

3. You have an unlimited supply of coins of denominations $c_1, c_2, \ldots, c_n$, which are positive integers. You wish to make change for a value of V (also a positive integer), using minimum number of coins. You may assume $c_1 = 1$, so that it is always possible to make change for any positive integer value. Describe and analyze an efficient algorithm for this problem.

4. You are given a rectangular sheet of paper filled with a black-and-white image. Your goal is to cut it into smaller sheets of paper each filled completely with black or completely with white. Every cut that you make must completely divide one sheet into two smaller rectangular sheets. The goal is to minimize the number of cuts. The input is represented as a $m \times n$ matrix of 0s and 1s, with 0 standing for Black and 1 for White. An example is shown below, with the input on the left and a legal sequence of 7 cuts on the right, with the first cut being made after the second column.

0	1	0	0	1
0	1	0	0	1
1	1	0	0	1
1	1	1	0	0
0	0	1	0	0

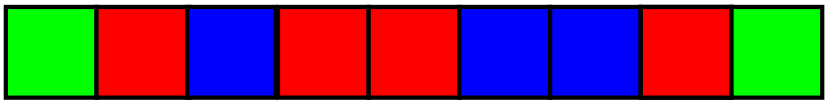
0	1
0	1
1	1
1	1
0	0

0	0	1
0	0	1
0	0	1
1	0	0
1	0	0

Give an efficient algorithm for the problem stated above.

5. **Bonus Problem:** Relevant for those taking a course in Theory of Computation

   In the game of 1-Dimensional Blocks, you see a long sequence of blocks, each colored Red, Blue or Green, as shown in the example below.



   The goal is to remove consecutive blocks of the same color of length 2 or more, in an attempt to eventually clear all the blocks. For the sequence above, removing the two red blocks first leads to a solution. This is illustrated more succintly as: $GRBRRBBRG \rightarrow GRBBBRG \rightarrow GRRG \rightarrow GG \rightarrow \emptyset$, where each letter stands for a color. On the other hand, there are inputs (eg: $RGRG$), for which it is not possible to clear all the blocks. Also note that the order is important - clearing the two blue blocks first does not lead to a solution.

   Give an algorithm that accepts a sequence of colors, and checks if it is possible to clear all the blocks.