

# CS 473U: Undergraduate Algorithms, Fall 2006

## Homework 2

Due Tuesday, September 19, 2006 in 3229 Siebel Center

Remember to turn in in separate, individually stapled solutions to each of the problems.

---

1. You are given an  $m \times n$  matrix  $M$  in which each entry is a 0 or 1. A *solid block* is a rectangular subset of  $M$  in which each entry is 1. Give a correct efficient algorithm to find a solid block in  $M$  with maximum area.

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

An algorithm that runs in  $\Theta(n^c)$  time will earn  $19 - 3c$  points.

2. You are a bus driver with a soda fountain machine in the back and a bus full of very hyper students, who are drinking more soda as they ride along the highway. Your goal is to drop the students off as quickly as possible. More specifically, every minute that a student is on your bus, he drinks another ounce of soda. Your goal is to drop the students off quickly, so that in total they drink as little soda as possible.

You know how many students will get off of the bus at each exit. Your bus begins partway along the highway (probably not at either end), and moves at a constant rate. You must drive the bus along the highway- however you may drive forward to one exit then backward to an exit in the other direction, switching as often as you like (you can stop the bus, drop off students, and turn around instantaneously).

Give an efficient algorithm to drop the students off so that they drink as little soda as possible. The input to the algorithm should be: the bus route (a list of the exits, together with the travel time between successive exits), the number of students you will drop off at each exit, and the current location of your bus (you may assume it is at an exit).

3. Suppose we want to display a paragraph of text on a computer screen. The text consists of  $n$  words, where the  $i$ th word is  $p_i$  pixels wide. We want to break the paragraph into several lines, each exactly  $P$  pixels long. Depending on which words we put on each line, we will need to insert different amounts of white space between the words. The paragraph should be fully justified, meaning that the first word on each line starts at its leftmost pixel, and *except for the last line*, the last character on each line ends at its rightmost pixel. There must be at least one pixel of whitespace between any two words on the same line.

Define the *slop* of a paragraph layout as the sum over all lines, *except the last*, of the cube of the number of extra white-space pixels in each line (not counting the one pixel required between every adjacent pair of words). Specifically, if a line contains words  $i$  through  $j$ , then the amount of extra white space on that line is  $P - j + i - \sum_{k=i}^j P_k$ . Describe a dynamic programming algorithm to print the paragraph with minimum slop.