

---

**IMPORTANT: Write each problem on a separate sheet of paper. Write your name on each sheet.** (Why? To speed up the grading we will use parallelization—each problem will be graded by a different TA. At the beginning of the class (on the date when the homework is due) there will be a separate pile for each problem.)

---

## 1 Homework - solve and turn in

**1.1 (due Sep. 10)** We are given an  $n \times n$  array  $A$  of zeros and ones. We want to find the size of the largest contiguous all-ones square. Give an  $O(n^2)$ -time algorithm for the problem.

**1.2 (due Sep. 10)** We are given  $n$  positive numbers  $a_1, \dots, a_n$ . The goal is to select a subset of the numbers with maximal sum and such that no three consecutive numbers are selected. Here are three example inputs together with optimal solutions (the numbers in boxes are selected):

$$\begin{array}{cccccc} \boxed{5} & \boxed{5} & 8 & \boxed{5} & \boxed{5} & \\ & \boxed{5} & 5 & \boxed{12} & 5 & \boxed{5} \\ 1 & \boxed{2} & \boxed{2} & 1 & \boxed{2} & \boxed{1} & 2 & \boxed{5} & \boxed{5} \end{array}$$

Give an  $O(n)$ -time algorithm for the problem.

**1.3 (due Sep. 10)** A **shuffle** of two strings  $A, B$  is formed by interspersing the characters into a new string, keeping the characters of  $A$  and  $B$  in the same order (for example, ‘several’ is a shuffle of ‘seal’ and ‘evr’). Given three strings  $A = a_1 \dots a_n$ ,  $B = b_1 \dots b_m$ , and  $C = c_1 \dots c_{m+n}$ , we would like to verify whether  $C$  is a shuffle of  $A$  and  $B$ . Give a dynamic programming algorithm for the problem

## 2 Bonus Homework - solve and turn in

**1.4 (due Sep. 10)** Let  $a_1, \dots, a_n$  and  $b_1, \dots, b_m$  be two sequences of numbers. We would like to find **the number of occurrences** of  $b_1, \dots, b_m$  as a **subsequence**<sup>1</sup> of  $a_1, \dots, a_n$

For example, if  $a_1 = a_2 = \dots = a_n = 1$ ,  $m = 2$ ,  $b_1 = b_2 = 1$  then the answer is  $n(n-1)/2$ . Give a polynomial-time dynamic programming algorithm for the problem.

- 1.5 (due Sep. 10)** We have an  $a \times b$  bar of chocolate (where  $a, b$  are integers). By breaking the bar we can either
- create two bars  $a_1 \times b$  and  $a_2 \times b$  where  $a_1, a_2$  are integers and  $a_1 + a_2 = a$ , or
  - create two bars  $a \times b_1$  and  $a \times b_2$  where  $b_1, b_2$  are integers and  $b_1 + b_2 = b$ .

We can further break the resulting bars. Our goal is to 1) end up with bars that are square (that is, have size  $1 \times 1$ , or  $2 \times 2$ , or  $3 \times 3$ , and so on) and 2) minimize the total number of breaks.

For example, if  $a = 2$  and  $b = 3$  then we use 2 breaks:

$$\begin{array}{l} 2 \times 3 \rightarrow \\ 2 \times 2 \text{ and } 2 \times 1 \rightarrow \\ 2 \times 2 \text{ and } 1 \times 1 \text{ and } 1 \times 1. \end{array}$$

---

<sup>1</sup>What is a subsequence? We say that  $b_1, \dots, b_m$  is a subsequence of  $a_1, \dots, a_n$  if by erasing some of the  $a_i$ 's from  $a_1, \dots, a_n$  we can obtain the sequence  $b_1, \dots, b_m$ ; for example 1, 1 is a subsequence of 1, 2, 1. More formally  $b_1, \dots, b_m$  is a subsequence of  $a_1, \dots, a_n$  if there exist  $1 \leq i_1 < i_2 < \dots < i_m \leq n$  such that  $b_j = a_{i_j}$  for all  $j \in \{1, \dots, m\}$ .

For example, if  $a = 2$  and  $b = 4$  then we use 1 break:

$$2 \times 4 \rightarrow \\ 2 \times 2 \text{ and } 2 \times 2.$$

Give a dynamic programming algorithm which computes a table  $T[1..a, 1..b]$  where  $T[x, y]$  contains the minimal number of breaks to “squareize” an  $x \times y$  bar.

### 3 Additional problems from the book (do not turn in)

Try to solve the following problems. A few of them can be on the quiz.

- 5.14, 5.15, 5.16, 5.20, 5.21, 5.26, 5.32,
- 6.1, 6.2, 6.6, 6.8, 6.9, 6.11, 6.14, 6.20, 6.21, 6.25, 6.26, 6.27, 6.29.

Check Jeff Erickson’s (UIUC) notes for additional problems:

<http://www.cs.uiuc.edu/~jeffe/teaching/algorithms/notes/05-dynprog.pdf>