# Algorithm Design and Analysis
# Homework #3

Due: 14:20, November 1, 2012

## Homework submission instructions

- Submit your programming assignment (problem 1) to the Judgegirl System (http://katrina.csie.ntu.edu.tw/judgegirl/). Also, use SVN to commit your report (named "report.pdf") in the same folder of your code.
- Submit the answer of the writing problems and the programming report by one of the following ways:
  1. Submit an electronic copy to the CEIBA system before the deadline; or
  2. Submit a hard copy right before the class on the day of the deadline.

  If you submit by both ways, the TA will randomly select only one of them for grading.
- Please make sure that you write down your name and school ID in the header of your documents. Otherwise, the TA will not grade them.
- If you choose to submit the answers of the writing problems via CEIBA, please combine the answers of all writing problems into only one file in the pdf format, with the file name in the format of "hw3 [ID].pdf" (e.g. "hw2 b99902010.pdf"). Otherwise, the TA will not grade them.
- What should be included in the report:
  1. Explain how your program works in detail.
  2. Derive the time complexity of your program and briefly explain why. No need for formal proof.
  3. **The reference of your report.**
- If you are stuck with any problem, you are welcome to ask the TAs via e-mails or in person during the office hour. The TA may discuss with you and give you some useful hints. :D

# Problem 1 – Slime King

## Description

You are an evil summoner and want to destroy the world with your secret weapons…Slimes.

Figure 1. On the left: a level 10 Slime; on the right, a level 1 Slime.

You have just summoned $N$ Slimes, and are going to merge all of them to create a strong **Slime King**.

$L_i$ represents the level (the size) of Slime $S_i$. You may force Slime $S_i$ to eat another Slime $S_j$, which will take $L_j$ minutes (for $S_i$ to digest). After that, Slime $S_j$ will cease to exist, and the level of Slime $S_i$ will be increased by $L_j$.

The level of any Slime cannot be increased without your magic, and therefore you will never let two or more Slimes eat at the same time. Moreover, after a Slime eats for $K$ times, it will lose the ability to eat another Slime.

You cannot wait to destroy the world, and therefore decide to merge all your Slimes in the fastest way.

## Input

The first line contains the two number $N$ and $Q$.$(1 \leq N, Q \leq 100{,}000)$

The second line contains $N$ integers $L_1, L_2, \ldots, L_N$, where two consecutive integers are separated by a space character ' '. $(1 \leq L_i \leq 1{,}000{,}000{,}000)$

The third line contains $Q$ integers $K_1, K_2, \ldots, K_Q$, where two consecutive integers are separated by a space. $(1 \leq K_i \leq 100{,}000)$

## Output

For each $K_i$ $(1 \leq i \leq Q)$, output a line containing an integer, representing the minimum required time (in minutes) to merge all of the Slimes if $K = K_i$ .

**Sample Input**

```
5 2
2 3 4 1 1
2 3
```

**Sample Output**

```
9
8
```

**Hint**

For sample #1: $S_2$ eat $S_4$(1 min); $S_2$ eat $S_5$(1 min); $S_3$ eat $S_1$(2 min); $S_3$ eat $S_2$(5 min).
For sample #2: $S_3$ eat $S_1$(2 min); $S_3$ eat $S_2$(3 min); $S_4$ eat $S_5$(1 min); $S_3$ eat $S_4$(2 min).
Remember to use **long long** data type.
There are 8 test cases in which $\mathrm{Q} \leq 100$ among the 10 test cases.

**Grade**

Judge result (10%)
Report & code (10%)

## Problem 2 - Gift

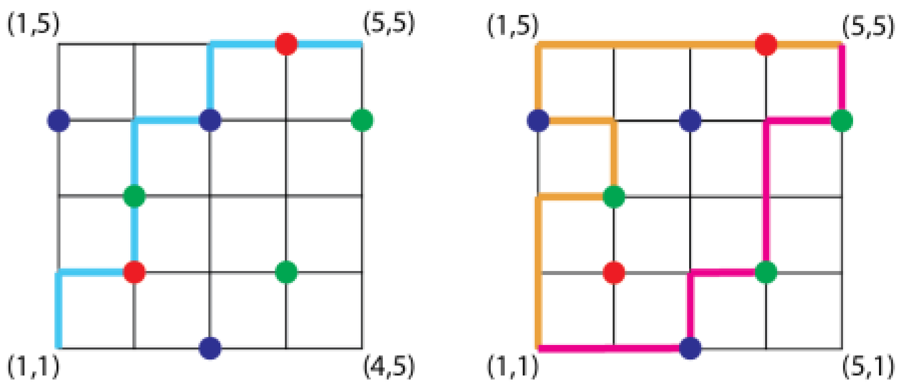
Summer Rift is a modern city. Its roads run either north-south or east-west and therefore the whole city looks like a grid table. As a result, every intersection in Summer Rift can be identified by a pair of integers $(x, y)$, $1 \leq x, y \leq N$.

Garen is a Summer Rift citizen and lives at intersection $(1,1)$. One day, he decides to visit Lux, who lives at intersection $(N, N)$. Garen don't want to be late, so he needs to follow the shortest route from $(1,1)$ to $(N, N)$. In other words, if Garen is at intersection $(x, y)$ on his way to Lux's place, he can only go to $(x+1, y)$ or $(x, y+1)$ in the next step.

Moreover, he decides to buy three different gifts for Lux; that means on the way he must pass through at least a flower shop, a bookstore, and a candy shop, respectively. Also, Garen doesn't care about the order of visiting different shops. The shops can only be located at intersections.

Now, you are given a map of Summer Rift. The map tells you for each intersection whether there is a flower shop, a bookstore, a candy shop, or no shop at all. At most there can only be one shop at each intersection. With the map at your hands, please tell Garen the number of routes he can choose to go to Lux's place and be able to buy all 3 needed gifts.

Hint: How many dimensions should you use when creating a Dynamic-Programming table? Well, you can think about how many "things" we should remember, such as where you are, how many televisions that have not been placed, etc.




Picture 1. examples of valid(left) and invalid routes(right).

1) (5%) We know that there is $C_{N-1}^{2N-2}$ shortest routes from $(1,1)$ to $(N, N)$. Design an $O(N^2)$-time algorithm to compute $C_{N-1}^{2N-2}$. (**Hint:** Pascal's Triangle)
2) (10%) Design an $O(N^2)$-time algorithm to compute the number of routes Garen can choose. You need to briefly explain how and why your algorithms work.

## Problem 3 – Go Go Go Greedy Girl!

— *It's a long story. Those you've defeated have come back.*

Here comes Gabrielle. She trusts that all of the problems in the world can be solved by greedy algorithms. Therefore, while taking the ADA course, she tries to use a greedy algorithm on every problem in her homework! The following is her answer sheet. However, Gabrielle doesn't give any proof for her algorithms. You, as a TA of the ADA course, need to determine whether her algorithms are correct or not. For each problem, if her algorithm is correct, please give a formal proof to her greedy algorithm. Otherwise, give a counterexample that makes her algorithm fail.

1) (10%) **Dividing and Matching by Segments**

On a **line**, there're $n$ amber points $a_1, a_2, \dots, a_n$ and $n$ blue points $b_1, b_2, \dots b_n$. Your task is to match each amber point to a blue point by a line segment, and each blue point is connected to exactly one amber point. The total length of those n segments should be minimized.

**Gabrielle:** Sort both $a_i$ and $b_i$ and re-index them. So that $a_1 < a_2 < \cdots < a_n$ and $b_1 < b_2 < \cdots < b_n$. Directly connect $a_k$ to $b_k$ for all $k = 1,2 \dots n$.

2) (10%) 連串矩陣相乘問題

You are given $n$ matrices $(A_1, A_2, \dots, A_n)$, where $A_i$'s size is $p_{\mathrm{i}-1} \times p_i$. Multiplying a $a \times b$ matrix with a $b \times c$ matrix needs $a \times b \times c$ operations, and results in a $a \times c$ matrix. Determine an order to multiply $A_1 A_2 \dots A_n$ so that the number of operations is minimized.

**Gabrielle:** Find the maximum value of $p_1, \dots p_{n-1}$, says, $p_k$. Then, multiply $A_k$ with $A_{k+1}$ first. Repeat the process (find the maximal $p_k$ then multiply $A_k$ with $A_{k+1}$) to the remaining matrices until there is only one matrix left. If there are multiple $p_i$ which are maximal, choose the one whose index is the smallest.

3) (10%) **Open!!**

There are $n$ days in this semester. Every day you receive a coupon. You will also need to buy a few items every day. The coupon allows you to buy one item for free. You can either use a coupon immediately on the day it was issued, or save it and use it later. Assume you already know that you will buy $n_d \geq 1$ items with the prices $P_d = \{p_{d,1}, \dots p_{d,n_d}\}$ on day $d$, $1 \leq \mathrm{d} \leq \mathrm{n}$. Determine the minimum cost you will pay to purchase the items after these $n$ days.

**Gabrielle:** First, assume you have to pay for everything. Then, consider day $d = n, n-1, \dots, 1$ in the given order, and for each day choose an item that you still have to pay in $P_d \cup P_{d+1} \cup \dots \cup P_n$ with the maximal price. Use a coupon to pay for this item.

4) (10%) **Breaking a String**

A programmer wants to combine $m$ strings $(S_1, S_2, \dots, S_m)$ into a string $S_1S_2 \dots S_m$. A certain string-processing language can combine two strings $S$ and $T$ into a string $ST$ of $n$ characters, which takes $n$ unit of time. Compute the lowest cost of a sequence combination.

**Gabrielle:** Always choose two consecutive strings $S_iS_{i+1}$ that contain the lowest number of characters and combine them. If there is more than one possible choice, choose the one whose $i$ is the smallest.

Although her greedy algorithm(s) is/are wrong for some problem(s), she still believes that these problem(s) may be solved by other smarter greedy strategies. Thus, she doesn't feel disappointed to greedy algorithms. Moreover, she is trying to open her XiGui using a greedy algorithm now! Keep going, greedy Gabrielle!

## Problem 4 – Coin Changing

Assume that there are $n$ kinds of available coins. The denominations are $1 = a_1 < a_2 < \cdots < a_n$ dollars, respectively, and each $a_i$ is an integer. Consider you are making change for $m$ dollars using the fewest number of coins.

1) (10%) Describe a greedy algorithm to make change when $n = 4$ and the denominations are 1, 5, 10 and 50. Prove that your algorithm yields an optimal solution.

   More generally, suppose that the set of denomination satisfies $a_{k-1}$ divides $a_k$ for every $k = 2,3 \dots n$. Show that the greedy algorithm always yields an optimal solution.
2) (5%) Give a set of denomination for which the greedy algorithm does not yield an optimal solution to some $m$.
3) (10%) Give an $O(nm)$-time algorithm that makes change for $m$ dollars for any set of $n$ different coin denomination.
4) (Bonus 5%) Now just consider $n = 3$ case. Let $r = \lceil a_3/a_2 \rceil$ and $M = ra_2$. Prove that the greedy algorithm works for every $m$ if and only if it works for $M$.

* $\lceil s \rceil$ means the smallest integer that is not smaller than $s$. For example, $\lceil 2.78 \rceil = 3, \lceil 7.01 \rceil = 8, \lceil 5 \rceil = 5$. In addition, note that $\lceil s \rceil$ is an integer.