CS 577: Introduction to Algorithms	9/27/2012
Homework 2	
Instructors: Shuchi Chawla and Dieter van Melkebeek	Due: 10/11/2012

This assignment covers greedy algorithms (chapter 4 in the textbook). See the first assignment for guidelines.

Ungraded problems

1. Suppose that you are given a digraph with nonnegative costs on the edges, and two vertices s and t. You want to find a cheapest path from s to t. However, the cost for an edge can grow over time. More precisely, the cost of using edge e as the tth edge of a path is given by a nondecreasing function $c_e(t)$ of t.

   Suppose you are given a subroutine that, on input the graph G, edge e, and time t, returns $c_e(t)$, and that you use the latter quantities in Dijkstra's algorithm. Give an example where this algorithm returns a suboptimal path from s to t.

2. You are given the position of a central hub and of n routers, as well as the number of people p_i served by each router i, $1 \leq i \leq n$. You would like to connect the routers and the hub to each other such that the total amount of cable is minimized. Under this restriction, there may be several optimal networks. In order to further narrow down the options, you would like to minimize the average time required for people to get access to the internet based on the assumption that construction on all cable links in the network begins at the same time. Cable links can be constructed at a rate of one mile of cable per day. As a result, shorter cable links are completed before the longer links. A router will have internet access as soon as it has a path to the central hub along completed cable links. More formally, if t_i is the time when router i gets connected to the central hub, you want to minimize the average connection time $\sum_i p_i t_i / \sum_i p_i$.

   Is the criterion of minimizing the average connection time adequate, i.e., can it narrow down the options further among the lowest cost solutions? Either give an example or prove that no example exists.

3. You want to throw a party and need to decide whom to call. You have n people to choose from, and you have a list of which pairs of these people know each other. You want to pick as many people as possible subject to the following constraint: at the party each person should have at least five other people whom they know *and* five other people whom they don't know.

   Give an algorithm that computes an optimal solution in time $O(n+m)$, where m denotes the number of pairs on the list.

4. Problem 4.6 in the textbook (p. 191).

Graded problems

5. (5 points) Problem "Cola" from the ICPC (see attachment). Your algorithm should run in time polynomial in the bitlength of N, i.e., polynomial in $\log N$.

6. (5 points) Problem 4.13 in the textbook (p. 194–195).
7. (5 points) Problem 4.26 in the textbook (p. 202).

Extra-credit problems

8. Problem I from the 2005 ACM-ICPC World Finals (see attachment), formalized as follows. You are given a list of workshops, each with the same start time but with varying end times and numbers of participants. You are also given a list of rooms, each available from that same start time but until varying end times and with varying capacities. Each room can accommodate at most one workshop. The goal is to schedule as many workshops as possible, and among those assignments pick one that maximizes the total number of people attending those workshops.

   Your algorithm should run in time $O(n \log n)$, where n denotes the sum of the number of workshops and the number of rooms.

9. In some courses you can choose a certain number k of the n assignments that will be dropped in the calculation of your grade. If all the assignments counted equally, the choice would be easy: simply drop the assignments with the lowest scores. However, each assignment may have a different maximum score. Your final homework grade will be the percentage ratio of your total score to the maximum possible score for the retained assignments. This leads to the following problem. You are given a value of k and a list of n assignment results (s_i, m_i), $1 \leq i \leq n$, where s_i denotes your score on the ith assignment and m_i denotes the maximum possible score on that assignment. Your goal is to find a set $I \subseteq \{1, 2, \ldots, n\}$ with $|I| = k$ such that $\sum_{i \notin I} s_i / \sum_{i \notin I} m_i$ is as large as possible.

   Give an algorithm that runs in time $O(n^2 \log n)$. For starters, aim for an algorithm that runs in time polynomial in the number of bits in the input.

Problem C : Cola

Time limit: 10 seconds

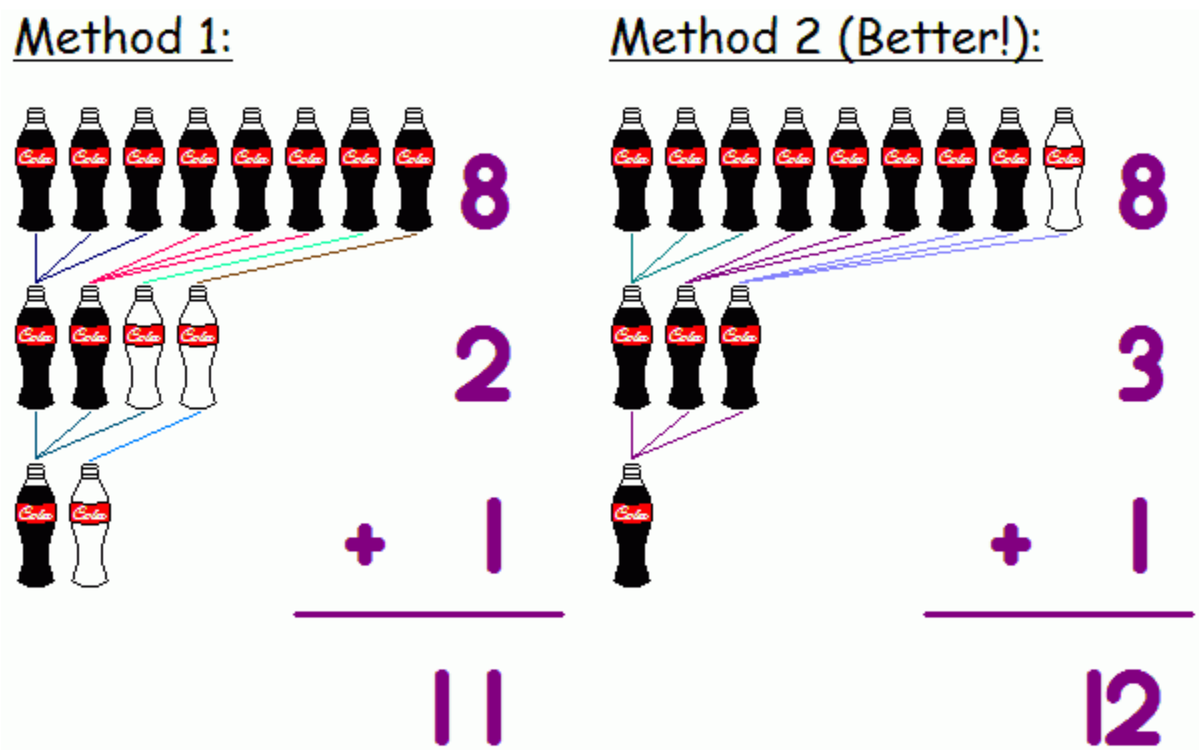
You see the following special offer by the convenience store:

" *A bottle of Choco Cola for every 3 empty bottles returned* "

Now you decide to buy some (say ***N***) bottles of cola from the store. You would like to know how you can get the most cola from them.

The figure below shows the case where ***N* = 8**. *Method 1* is the standard way: after finishing your 8 bottles of cola, you have 8 empty bottles. Take 6 of them and you get 2 new bottles of cola. Now after drinking them you have 4 empty bottles, so you take 3 of them to get yet another new cola. Finally, you have only 2 bottles in hand, so you cannot get new cola any more. Hence, you have enjoyed 8 + 2 + 1 = 11 bottles of cola.

You can actually do better! In *Method 2*, you first borrow an empty bottle from your friend (?! Or the storekeeper??), then you can enjoy 8 + 3 + 1 = 12 bottles of cola! Of course, you will have to return your remaining empty bottle back to your friend.



Input

Input consists of several lines, each containing an integer ***N*** (1 ≤ ***N*** ≤ 200).

Output

For each case, your program should output the maximum number of bottles of cola you can enjoy. You may borrow empty bottles from others, but if you do that, make sure that you have enough bottles afterwards to return to them.

Sample Input

```
8
```

Sample Output

```
12
```

Note: Drinking too much cola is bad for your health, so... don't try this at home!! :-)

Idea from a traditional IQ challenge question.
Special Thanks: Jonathan Mak

Problem I
Workshops
Input File: workshops.in

The first Californian Conference on Holism took place back in 1979 in San Francisco. The term "Californian" was a slight exaggeration, as all 23 participants actually lived in San Francisco. Several years later, in 1987, the conference was truly Californian, with 337 participants from all over the state. Since then, the number of participants has been growing like the size of memory chips. In 1993 the conference was renamed the American Conference on Holism (2549 participants), and a second renaming (World Conference on Holism) followed in 1997, when the number of participants from all over the world had grown to 9973. The conference obtained its present name (Galactic Conference on Holism) in 2003 after some discussion as to whether or not the word Galactic was intended to exclude extragalactic life forms. Still the next year, all registered participants were terrestrial—though a few participants positively reported to have sensed extraterrestrial presence.

The number of workshops grew with the number of participants. For the upcoming conference, the organization has to face some down to earth but very nasty scheduling problems. For the 2005 conference the board has decided to have no more than 1000 workshops concurrently. Nevertheless they had to rent every hall or classroom they could lay their hands on. Some of these rooms are available for a restricted time only.

In the morning of the first day the opening meeting takes place in a football stadium, and in the afternoon the participants attend workshops. Before lunch each participant has to indicate which workshop he or she wants to join that afternoon. The organizing staff then has a list of all workshops, including the duration and the number of participants for each workshop. They also have a list of all available rooms, with the capacity of each room, and the time this specific room must be cleared. With this information the staff must schedule each workshop in a room of sufficient capacity and sufficient availability in time. As this problem is not necessarily solvable, some overflow capacity is supplied by tents in the football stadium. These tents have plenty of capacity, but they are unpleasantly warm and noisy. So the organizing staff wants the schedule to minimize the number of tent workshops—that is, workshops that are not assigned to a room. If there are multiple solutions that minimize the number of tent workshops, the staff wants to minimize the number of participants attending tent workshops.

We ask you to supply such a schedule (preferably before lunch is over).

Input

The input file contains several trials. Each trial consists of two parts: the list of workshops, and the list of rented rooms.

The list of workshops starts with a line containing the number of workshops w ($0 < w \leq 1000$). Each of the next w lines contains two numbers, describing a workshop. The first number is the number p of participants ($0 < p \leq 100$), and the second number is the duration d of the workshop in minutes ($0 < d \leq 300$). For your convenience, other details of the workshops are omitted. All workshops start at 14:00.

The list of rented rooms starts with a line containing the number of rented rooms r ($0 < r \leq 1000$). Each of the following r lines contains the description of a rented room. A line describing a rented room contains the number s of seats in the room ($0 < s \leq 100$), followed by the time when the room must be cleared, in the format *hh:mm* where *hh* represents the hour and *mm* represents the minute, using a 24-hour clock. All the rooms are available starting at 14:00. All times when rooms must be cleared are between 14:01 and 23:59, inclusive.

The input is terminated by a line consisting of the integer zero.

Output

For each trial in the input file the output must contain a line consisting of the trial number, the number of tent workshops, and the number of participants attending tent workshops. Follow the format shown in the sample output.

Sample Input	Output for the Sample Input
1 20 60 1 30 16:00 2 20 60 50 30 1 30 14:50 0	Trial 1: 0 0 Trial 2: 2 70