Hamburg Collegiate Programming Contest

at Hamburg University of Technology

July 2011

Problem Set

Prof. Dr. Volker Turau
Christian Renner
Christoph Weyer

Sponsored by

This page is intentionally left (almost) blank.

Contents

A :. Top Spin

Have you ever been to Heidepark Soltau? If so, you probably remember the Top Spin. That's one of these nasty fun rides that throws you around and lets you hang in the air upside-down. Since you don't like hanging upside-down, you stick to watching, but don't want to get bored. So you find some entertainment: You wonder how many people are going to be on each ride for the people currently in line, hoping that this will keep you busy, until your friends (taking the current ride) will pick you up again.

The whole thing could be very easy, if there were no groups of people. These groups have the silly and annoying wish of not being split up, unless unavoidable, i.e., there are more people in the group than seats per ride. To cope with that problem, you make up the following rules:

1. You split up groups with more persons than available seats into subgroups of sizes equal to the total number of seats and one last subgroup consisting of the remaining persons. However, there is one exception. The last subgroup must not consist of only one person (a person belonging to a group does not want to be on his or her own). To prevent that, you pull over one person from the second-to-last subgroup.
2. Your group-splitting approach leaves you with a new line of groups. For this new line, you apply a very simple seating scheme: Groups are seated with respect to their position in the line until either no seat remains or there are only groups (in the whole line) larger than the remaining number of seats. The latter implies that groups too large for the current ride are skipped in favor of later but sufficiently small groups.

Input Specification

The first line of input contains the number of test cases, at most 100.

Each test case starts with a line holding $10 \leq S \leq 100$, the number of seats in the Top Spin, and $1 \leq G \leq 1\,000$, the number of groups. Then follows one line with G integers (separated by single blanks), the number $1 \leq p_i \leq 200$ of people per group. The left-most group of input is the first one in line.

Output Specification

Per test case one line with the number of people actually seated in the successive rides of the Top Spin.

Sample Data

Input	Output
2 10 6 2 4 4 6 5 3 10 4 21 9 11 8	10 9 5 10 9 4 9 9 8

This page is intentionally left (almost) blank.

B :. Healthy Burglar App

The burglar scene has changed considerably. Nowadays, burglars are very picky about their health and are particularly afraid of an herniated disk. Since you are an inventive salesman, you have decided to create a smartphone app for the modern burglar.

Your app only requires a description (value and weight) of all items worth being stolen and a critical weight. The latter is the maximum weight that the burglar can carry without risking an herniated disk. The worst thing happening is that the burglar cannot leave the place due to too heavy a load of stolen goods. The app therefore calculates the maximum possible profit (summed values of items) with a total weight not exceeding the critical weight. The goal of your application is to enable the burglar to decide whether his burglary will be profitable with a single break-in. As all loose burglars know, never break in the same place twice!

Input Specification

The first line of input contains the number of test cases, at most 100.

Each test case starts with a line holding two integers: $1 \leq N \leq 30$, the number of valueable items, and $1 \leq W \leq 1\,000$, the critical weight. Thereafter follow two lines with N integers each. The first line contains the values v_i of the items, the second line contains the weights w_i of the items ($1 \leq v_i, w_i \leq 100$).

Output Specification

Per test case one line with the maximum profit.

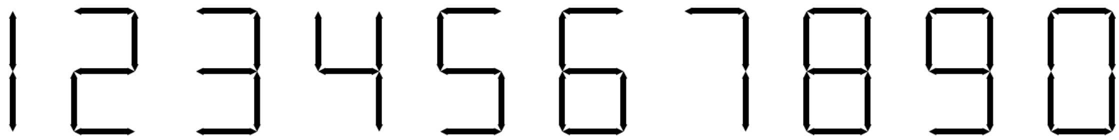
Sample Data

Input	Output
2 5 10 1 4 9 16 25 1 2 3 4 5 5 9 25 16 9 4 1 1 2 3 4 5	42 50

This page is intentionally left (almost) blank.

C :. Matchsticks

On your way home you unfortunately missed the bus and have to wait 20 minutes for the next one to arrive. Since you get bored quickly, you decide to rack your brains a little. While trying to think of something challenging, you spot a box of matchsticks on the ground. You pick it up, open it, and find a few matchsticks. ”So there’s your challenge,” you think: ”How many different numbers could I create using all matchsticks in the box?”.



Example: If you find 5 matchsticks (see sample cases), you can create the numbers 17, 71, 2, 3, and 5. Note that numbers may begin with leading zeros, e.g., ”007” is a valid number.

Input Specification

The first line of input contains the number of test cases, at most 100.

Each test case consists of a single line with an integer $0 \leq N \leq 80$, the number of matchsticks.

Output Specification

Per test case one line with the amount of possible numbers built from the matchstick digits.

Sample Data

Input	Output
4	0
1	1
3	5
5	12
7	

This page is intentionally left (almost) blank.

D :. Caribbean Airlines

Caribbean Airlines (CA) is a small airline. They offer connections between some islands of the Caribbean. There is one flight per connection every day, if the connection is one-way. Some of the connections are offered in both directions, so that there are two flights per day (one in each direction). However, their concept is a little different: They do not own planes, but buy flight contingents from other airlines, so that they don't have to worry about ideal flight routes. They also have a unique one-price policy, i.e., every flight costs a fixed amount of money, because they think this is what costumers want.

Due to an increasing competition on the flight market, CA has decided to reduce the prices. At the same time, they want to keep their flight network intact. At the moment, round-trips are possible starting at every island that CA serves (possibly taking a few intermediate stops and with different routes on the flights there and back). Since this is their second marketing strategy—they call it the every-island round-trip guarantee—there is no way to cancel contingents violating this property.

The plan therefore is to identify the minimum ticket price while preserving the one-price policy and every-island round-trip guarantee. To achieve this goal, the managers of CA have compiled the charter costs for each flight provided by other airlines. They only considered the cheapest possibility for each possible connection. In addition, they have calculated the minimum fee that must be added to the charter costs enabling CA to operate profitably. The final ticket price will be the largest charter cost of all selected connections plus the fee.

Since managers only plan but do not execute, they have hired a skilled computer scientist to tell them the lowest possible ticket price. And as you may have guessed, that computer scientist is you!

Input Specification

The first line of input contains the number of test cases, at most 100.

Each test case starts with a line holding three integers: $2 \leq I \leq 200$, the number of islands CA currently serves; $I \leq C \leq 2 \cdot I \cdot (I - 1)$, the number of available connections (charter flights), and $50 \leq F \leq 1\,000$, the fee per ticket (in cents, the currency at headquarters in Martinique is Euro). Thereafter follow I lines with three integers o_i, d_i, c_i each, where o_i is the origin of a flight, d_i is the destination, and $1\,000 \leq c_i \leq 100\,000$ is the charter cost (in cent). Islands are numbered from 1 to I.

CA always has the same contingent of seats per flight, so you don't have to worry about flight capacities and passenger volumes. It is also safe to assume that the given connections meet the every-island round-trip guarantee.

Output Specification

Per test case one line with the minimum ticket price (including the fee), which is neither violating the one-price policy nor the every-island round-trip guarantee.

Sample Data

Input	Output
2 3 6 100 1 2 1000 2 1 2000 2 3 3000 3 2 1500 1 3 2500 3 1 3500 4 7 200 1 2 1000 2 3 1000 3 4 1000 4 1 5000 4 3 2000 1 3 2500 3 1 3000	2600 3200

E :. Hope for Konrad's Team?

Konrad is a fascinated handball fan. His team and last year's champion is unfortunately not really doing well this year. At some point of the season Konrad is wondering, if there is any hope left for his team to defend their title. After Konrad has worked on the problem himself for quite a while, he has realized that it's not really that simple. He therefore asks you to help and tell him, if there is hope left for his team.

Surely, you want to help him! However, you're no handball fan at all, so that Konrad informs you about the rules:

- The winner of a game receives two points, and the looser receives no point. If the game ends in a tie, both teams split the points, i.e., each team receives one point.
- There are exactly two games between the same two teams per season
- Usually, teams have the same amount of games played. However, you must not count on that, because there are quite a few games postponed this year.
- At the end of the season, the team with the most points is the new champion. If there is more than one team with the same amount of points, the goal difference is used as a tie breaker. However, no one can really foresee the precise results of upcoming games, so that Konrad decides that there is hope for his team, if his team can end the season with either more or the same amount of points as the second-best team.

Input Specification

The first line of input contains the number of test cases, at most 100.

Each test case starts with one line containing the name of Konrad's team (a team name does not contain blanks). Next is a line holding $2 \leq T \leq 20$, the number of teams. Then follow T lines with the current ranking. Each line contains the current rank of the team ($1 \leq R \leq T$), the name of the team (a string without blanks), and the number of points P of that team ($0 \leq P \leq 4(T-1)$). After the ranking follows a single line with an integer G ($0 \leq G \leq T \cdot (T-1)$), the number of remaining games of the season, and thereafter G lines describing one game: each line contains the names of the two playing teams (the team names are separated by a single blank).

Output Specification

Output "`YES`", if there is hope for Konrad's team to win the championship, or "`NO HOPE`", if it is not possible to win.

Sample Data

Input	Output
4 Your_Team 3 1 Team_B 4 2 Team_C 2 3 Your_Team 2 2 Team_C Team_B Team_C Your_Team Your_Team 3 1 Team_B 4 2 Team_C 4 3 Your_Team 0 2 Team_B Team_C Team_B Your_Team Your_Team 3 1 Team_B 5 2 Your_Team 4 3 Team_C 1 1 Your_Team Team_C Your_Team 3 1 Team_B 5 2 Team_C 3 3 Your_Team 2 1 Your_Team Team_C	YES NO HOPE YES NO HOPE

F :. Fair-Trade Coffee

Bernd likes Coffee. He is really into a special, quite extra-ordinary and extra-expensive flavor from a single producer on Java. Bernd is convinced that the producer's original price of the coffee cannot be that enormous, so that a large share of the final price stays within the distribution chain. Every distributor is adding some percentage to his purchase price (per pound), making the coffee more expensive. Bernd can only buy the coffee from vendors (vendors are the distributors directly selling to costumers), that all offer his favorite coffee at the same price.

Due to fortunate circumstances (you better don't ask!), Bernd has obtained a pretty good picture of the distribution chains and the profit margins of each distributor. However, he does not know at what prices the producer can sell his coffee to his distributors. Unfortunately, that's precisely what Bernd wants to know: Since he loves his coffee so much, he wants to make sure that he buys his coffee from the vendor with the cheapest distribution chain, i.e., the profit for the producer is the highest among all chains. Could you assist Bernd in finding out the maximum margin of the producer for the cheapest distribution chain? It is safe to assume that all distributors buy their coffee at the lowest price and that there is no limitation on the amount of coffee per distributor.

Input Specification

The first line of input contains the number of test cases, at most 100.

Each test case starts with a line holding an integer $1 \leq D \leq 1\,000$, the number of distributors. Then follows one line with D integers $0 \leq M_i \leq 100$ ($1 \leq i \leq D$), the profit margin of the distributor with respect to his purchase price in percent (e.g., if $M_i = 50$ the i'th distributor sells the coffee for 150% of his purchase price). The third line per test case contains the integer $0 \leq C \leq 10\,000$, the number of sales channels. Thereafter follow C lines with three integers b, s, and m ($1 \leq b, s \leq D$), stating that distributor b buys coffee from distributor s, where s may offer a special price (margin) $0 \leq m \leq M_s$. Some distributors don't offer special prices ($m = -1$), so that M_s is the margin on that channel.

The final four lines of a test case describe the distributors buying directly from the producer and the vendors selling coffee to Bernd. The first line holds the integer $1 \leq A \leq 50$, the number of distributors acquiring from the producer. The next line contains these A producers, represented by integers $1 \leq a \leq D$. The third line holds the number $1 \leq V \leq 50$ of vendors selling to Bernd. The fourth line contains these V vendors, represented by integers $1 \leq v \leq D$.

Output Specification

Per test case one line with the vendor guaranteeing the best purchase price for the producer and the percentage of the final sales price that the producer receives (an absolute or relative error of 10^{-6} is accepted). If there is more than one vendor allowing for the same percentage, output the smaller vendor number.

Sample Data

Input	Output
2 5 10 20 30 40 50 6 2 1 -1 3 2 -1 4 3 -1 5 4 -1 4 1 -1 5 3 5 1 1 2 4 5 5 50 20 30 40 10 7 2 1 -1 3 2 -1 4 3 10 5 4 10 4 1 -1 5 3 -1 3 1 30 1 1 2 4 5	4 64.9350649351 5 57.7934462232

G :. Car Wash

Your friend Carmen is owning the well-known company *Carmen's Car Wash*. Since the equipment, i.e., the individual stations of the car wash, are considerably old, Carmen wants to invest into a completely new refurbishment. She has therefore talked to the bank, who is willing to give her a credit. Looking into the catalog of available machines, Carmen has figured that choosing an optimal configuration is going to be quite difficult. For each station in the car wash, Carmen can choose from a variety of machines coming at different costs and processing times.

Carmen has thought about her options for a long time: Most of her costumers complained about long waiting times before entering the car wash. On busy days, the waiting time depends on the slowest station. She therefore believes that it would be best to equip her car wash with stations, so that the slowest station has the lowest possible processing time.

Carmen is very concerned about finding the optimal solution, but she doesn't really like those mathematical problems. Would you be so kind and help her out?

Input Specification

The first line of input contains the number of test cases, at most 100.

Each test case starts with a line holding $1 \leq M \leq 1\,000$, the number of machines Carmen takes into consideration, and $0 \leq C \leq 1\,000\,000\,000$, the credit Carmen received from the bank. Thereafter follow M lines with 4 entries separated by a single blank:

- station: A string telling what kind of station the machine is
- product ID: A string with the unique product ID in the catalog
- price: The price of the machine, $0 \leq \text{price} \leq 1\,000\,000$
- time: The processing time of the machine for one car, $0 \leq \text{time} \leq 1\,000\,000\,000$

Output Specification

Per test case one line with the time needed by the slowest station in the new car wash

Sample Data

Input	Output
1 18 800 prewash Prewash_Light 66 19 prewash Prewash_Star 103 17 prewash Prewash_Ultimate 156 15 prewash Prewash_Ultimate_II 219 12 foaming FoamParty_2000 36 14 mainwash MW3k 35 21 mainwash MW4k 88 18 mainwash MW5k 170 12 mirror_wash Starlet 52 14 underfloor_wash DirtKiller 54 14 underfloor_wash CornerMob 99 12 dryer WindStille 157 19 dryer LauesLueftchen 175 17 dryer SteifeBriese 210 15 dryer Orkan 293 12 polisher Shiny 18 12 polisher MoreShiny 30 15 waxing WaxProtector 4 14	15

LAST PAGE