# PRACTICE QUESTIONS

## PROBLEM DESCRIPTION A: HELLO IEEE

Create a program that prints "Hello IEEE"

## PROBLEM DESCRIPTION B: BABY NAMES

### BABY NAMES

Many years ago, most babies were given traditional names like John, Mary, Paul or Margaret, for example. These days, the choice is much more sophisticated: popular names for babies are Jayden, Kayla, Cameron, and Hailey. Actually, if you want to choose the name for your baby, the Internet is there to help you, with plenty of sites where you can make selections according to many different criteria. I tried that, but the simple criteria I am after was not implemented: I absolutely want a name with three equal letters. More precisely, I want a name in which one of the letters occurs at least three times, as in “Samantha”.

### TASK

Your task is writing a program that, given a list of names, selects those in which one of the letters occurs at least three times.

### INPUT

The input file contains an undetermined number of lines. Each line contains a non-empty string, representing a name.

### OUTPUT

The output contains the list of chosen names, one per line, in alphabetical order, or the message “Not found” if no names match the required criteria.

### CONSTRAINTS

The number of lines in the input file, N: 1 <= N <= 1000;

Each name in the input file is written is capitalized, i.e., the first letter in an uppercase letter and the remaining letters are lowercase. All characters in the name are letters and the length of the name is not greater that 15. The names are unique.

### INPUT SAMPLE

Emily
Madison
Hannah
Emma

Ashley
Abigail
Alexis
Olivia
Samantha
Sarah
Melissa
Margaret
Virginia
Elizabeth
Laura
Vera
Diana
Ruth
Julia
Lillian

## OUTPUT SAMPLE

Lillian
Samantha
Virginia

## SECOND INPUT SAMPLE

Jacob
Michael
Joshua
Matthew
Andrew
Christopher
Joseph
Daniel
Nicholas
Ethan

## SECOND OUTPUT SAMPLE

Not found

# PROBLEM DESCRIPTION C: BMW M3

## BMW M3

Laura decided it’s time to buy a new car. She has a fancy for high-performance convertibles and chose the brand new 2010 BMW M3 Convertible. She browsed the Internet for dealers in her neighbourhood and found plenty. Curiously, the prices vary somewhat. She is willing to visit some dealers, hoping to make a better business, but it is out of the question to visit them all. So, she chose to visit only those whose advertised price for the M3 is less or equal to the average price of that car.

## TASK

Your task is writing a program that, given a list of dealers and their prices for the BMW M3, selects those dealers whose price is less or equal to the average price.

## INPUT

The first line of the input file contains one positive integer, N, the number of dealers. Exactly N lines follow, one for each dealer. Each of these lines contains a string, S, representing the dealer name, and a positive integer, P, representing the price of the BMW M3 at that dealer.

## OUTPUT

The output file shall list the names of dealers whose price is less or equal to the average price, one per line, in alphabetical order. Each line shall contain the name of a dealer and its price rank. The price rank is the position of the dealer in a list of dealers sorted by price. If two or more dealers sell the M3 by the same price, their rank should be the same. In other words, the rank of a dealer *x* is equal to the number of dealers who sell the M3 cheaper than *x* does, plus one.

## CONSTRAINTS

The number of dealers, N: 1 <= N <= 1000;
The name of the dealer, S, is composed of no more that 15 lowercase characters (no diacritics), decimal digits or the underscore character. The names are unique.
The price of the M3, P: 1 <= P <= 1000000

## INPUT SAMPLE

```
super_bm 70000
bm_shop 72000
convert_bm 74000
my_car 73000
lux_bm 78000
bav_cars 72000
```

## OUTPUT SAMPLE

```
bav_cars 2
bm_shop 2
my_car 4
```

super_bm

# COMPETITION QUESTIONS

## PROBLEM DESCRIPTION A: HIMALAYAS

### HIMALAYAS

I am an airline pilot, and I work for Extreme Airways. Today I am flying one of my favourite routes, Shanghai-Munich. I love those two cities, but what I really like best about this route is that it goes over the Himalayas. They are fantastic mountains and they offer one of the best scenarios on earth, especially when looked at from above. I am not above the Himalayas yet, but I am flying towards them, from East to West. This is early morning, the rising sun is behind my plane and there is a wonderful soft light bathing those numerous great hills.

Actually, I cannot distinguish the individual hills, because they are still far, but I can see the skyline of the mountains, and I can pinpoint many peaks. Indeed, there are more than 200 peaks over 6000 meters in the Himalayas and I do not know them all by heart. Still, I have a list of peaks, each with its name, its coordinates and its altitude. Furthermore I know that by one of those mysteries of Nature, the slopes of all the hills in the Himalayas are inclined at exactly 200%, meaning that if you advance 10 meters on the horizontal, you advance 20 meters on the vertical. In other words, the angle of the slope with the horizontal is the arctangent of 2.0.

### TASK

Your task is writing a program that, given a list of all the peaks in the Himalayas, computes the westward skyline, as seen from very far. For each peak we have its name, and three coordinates: the south-north coordinate, the east-west coordinate and the altitude. We assume the earth is flat and that we are watching from an infinite distance along the east-west axis. We also know that all slopes are inclined at 200%. We want to compute the list of names of those hills that form the skyline, i.e., those peaks that are not hidden behind another hill and not sitting in front of other hills. In this later case, please remember that the light is low and we are not able to distinguish a hill that is in front of another taller hill.

### INPUT

The first line of the input file contains one positive integer, N, representing the number of peaks. Exactly N lines follow. In each of these lines there is a string S and three integers, X, Y and Z: S represents the name of the peak; X represents the south-north coordinate (lower values are southernmost); Y represents the east-west coordinate: (lower values are easternmost); Z represents the altitude.

### OUTPUT

The output contains the names of the peaks from the skyline, as described, one name per line, in south-north order (i.e., the first line contains the name of the southernmost peak from the skyline, and so on, northwards.)

## CONSTRAINTS

The number of peaks, N: N > 0, N <= 1024;

The name, S, representing the names of the peaks: S is composed of no more that 15 lowercase characters (no diacritics), decimal digits or the underscore character. The names are unique.

The integers, X, Y and Z, representing the coordinates of each peak: 0 <= X <= 10000; 0 <= Y <= 10000; 0 <= Z <= 10000. All triples <X, Y, Z> are unique.

## INPUT EXAMPLE

```
4
melungtse 10 5 4
lothse 8 2 3
makalu 3 3 5
everest 7 4 8
```

## OUTPUT EXAMPLE

```
makalu
everest
melungtse
```

# PROBLEM DESCRIPTION B: VANGELIS

## VANGELIS, THE BEAR

Vangelis the bear wants his older bear brother Mitsos to give him some of the honey he just brought to the cave but Mitsos is a geek and, instead of just giving it to his brother, he asked him to solve a quiz for him in order to win the honey.

The quiz asked him to find out how many multiples of a given number that are greater or equal to a number A, and less or equal than another number B contain only certain digits. Unfortunately Vangelis is not that good at math and cannot figure this quiz out.

## TASK

You task is to help Vangelis by writing a program which, given an integer, an integer interval and a set of decimal digits, computes the number of numbers from the interval that are multiples of the first given number and whose decimal representation uses only digits from the given set.

## INPUT

The input file contains two lines.
The first line contains three integers X, A and B: X represents the number whose multiples we are to consider; A represents the lower bound of the interval; B represents the upper bound of the interval.
The second line contains a single integer, D. The digits of D are those that you are allowed to use for the multiples.

## OUTPUT

The output file contains one line only, with a number. This number represents the number of numbers in the closed interval [A..B] that are multiples of X and that are written using only digits from D.

## CONSTRAINTS

The given number X: $1 \leq X < 10^9$;
The boundaries of the given interval, A, B: $1 \leq A \leq B < 10^9$;
The number representing the digits, D: $1 \leq D < 10^9$

## INPUT SAMPLE

6 1 50
2468

## CORRESPONDING OUTPUT SAMPLE

4

## SECOND INPUT SAMPLE

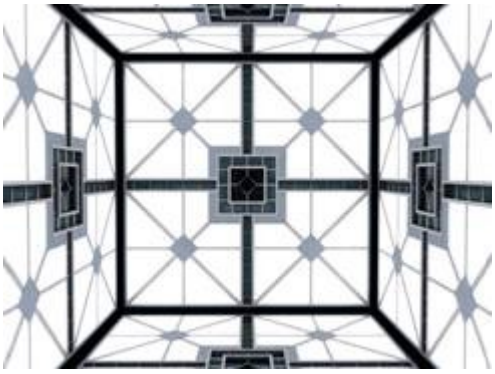
7 35 91
135797531

## CORRESPONDING OUTPUT SAMPLE

3

# PROBLEM DESCRIPTION C: CUBE

## CUBE



You wake up and find yourself inside an empty cubic room.

At first you feel surprised but then you notice that at the centre of each wall, roof, and floor there is a gate that seems to give access to another similar empty cubic room.

You start feeling a kind of fear as you move from room to room and realize that there is no water, there is no food, and the number of rooms seems endless. Unless you find an exit to get out of there, you are going to starve to death! Is this only a bad dream?

## TASK

Compute the number of reachable rooms (i.e., rooms that you may end up in) after K time units, provided that at in one time unit any of the six adjacent rooms to the current room can be chosen (with equal probability) to jump into.

Remember: In all cases, standing still in the current room is not an option. Furthermore, assume that the total number of rooms is unbounded.

## INPUT

The input file contains one line, with a single non negative integer, K, representing the number of time units.

## OUTPUT

The output file shall have only line, displaying a single integer, representing the number of rooms that can be reached after exactly K units of time.

## CONSTRAINTS

· Number of time units, K: 0<K<17576.

## NOTE

The number of rooms can be greater than $2^{31}$-1, but in no case will it be greater than $2^{63}$-1.

## SAMPLE INPUT 1

1

## SAMPLE OUTPUT 1

6

## SAMPLE INPUT 2

4

## SAMPLE OUTPUT 2

```
85
```

# PROBLEM DESCRIPTION D: MAZE KEYS

## MAZE KEYS

Trapped within a maze, you know you have to find a way out as quickly as possible. The maze has many doors and in order to open them you need special maze keys. All the keys have the same shape, and can open any door. However, once used to open a single door, a key becomes stuck in the door lock. That door will therefore always remain open, but the used key will be rendered useless to open any other doors. This means that for example, if you need to open three different doors, you would need three different keys.

A maze is represented by a grid. An empty grid cell is represented by **'.'**, while a cell containing a wall is represented by **'#'**. The starting position is indicated by **'S'**, the exit is indicated by **'E'**. Likewise, **'D'** means a door and **'K'** means a key. Here is an example:

```
##########
#S......K#
#..####D##
#K.#....E#
##########
```

Each time you move from one cell to another, you spend 1 time unit. You are only allowed to move in four different directions: up, down, left and right (no diagonal moves are allowed). You can move in any empty cell, and also in the start and end positions. If you move to a cell containing a key, you automatically grab that particular key and take it with you (there is no limit to the amount of keys you can carry), and that cell becomes empty. If you try to move to a cell with a door, and you still carry an unused key, you open the door using a key that becomes stuck there and from that moment on the door remains open (and you carry one less key). Walking through walls is not possible.

Can you find the quickest route to exit the maze? In the maze given above, the quickest route would take 11 time units. First you spend 7 time units travelling to the right and grabbing a key. Then you spend 2 more time units going to the left and then down, opening the door, and finally you spend two more time units going down and right, reaching the exit.

## TASK

Given a maze with **C** cells in each row and **R** cells in each column, completely enclosed in walls and with **D** doors and **K** keys, your task if to find the minimum distance you have to travel in order to go from your starting position to the exit position.

## INPUT

The first line of input has two integers **R** and **C**, separated by a single space, indicating respectively the number of rows and number of columns of the maze.

Then follow exactly **R** lines, each one of them with **C** chars, indicating the contents of the maze, where each cell is represented by one of the following chars:

**'.'** - an empty cell

**'#'** - a wall (you cannot walk trough)

**'D'** - a door (you can walk trough only if you have an available key)

**'K'** - a key

**'S'** - your starting position (just one per maze)

**'E'** - the maze exit (just one per maze)

All mazes will be completely surrounded by walls (and these walls are part of the input), there will be only one **'S'** and one **'E'** and it will always possible to go from the start to the exit.

## OUTPUT

The output consists of a single line containing an integer indicating the minimum time units that you have to spend in order to reach the exit from your starting position.

## CONSTRAINTS

3 ≤ **R, C** ≤ 50 - Number of rows and columns

0 ≤ **D, K** ≤ 5 - Number of doors and keys

## SAMPLE INPUT 1

```
3 10
##########
#K.S.KDDE#
##########
```

## SAMPLE OUTPUT 1

```
9
```

## SAMPLE INPUT 2

```
5 10
##########
#S......K#
#..####D##
#K.#....E#
##########
```

## SAMPLE OUTPUT 2

```
11
```

# PROBLEM DESCRIPTION E: MOBILE PRIMES

## MOBILE PRIMES

Mike and Lisa met at an important IEEE event. Realizing that they shared points of view on many engineering issues, they decided to exchange e-mail addresses and mobile telephone numbers. “My mobile number is 971 421 139”, Lisa said. “How interesting”, Mike replied, “have you noticed that your number is formed by appending three three-digit prime numbers?”

“Indeed it is, Mike”, Lisa riposted, “but, in fact, there are 16 different three-digit prime numbers inside my mobile number”.

Mike was baffled!

## TASK

Your task is writing a program that, given a positive integer, computes the number of unique prime numbers with 1, 2, 3, etc., decimal digits existing in the given number. We say that a non negative integer X exists in non negative integer Y if the decimal digits of X are a subsequence of the decimal digits of Y. Recall that “a subsequence is a sequence that can be derived from another sequence by deleting some elements without changing the order of the remaining elements. For example, ABD is a subsequence of ABCDEF”[1].

## INPUT

The input file has one line, in which there is a positive integer, N.

## OUTPUT

The output contains K+1 lines, where K is the number of decimal digits of N. The first line displays the total number of prime numbers existing in N. The (d+1)-th line, for d = 1, 2, ...K, displays the number of prime numbers with exactly d digits existing in N.

## RESTRICTIONS

Size of N: N >= 1, N <= 999999999;

## INPUT EXAMPLE

714139

## OUTPUT EXAMPLE

23
2
8
8
2

2
1

# PROBLEM DESCRIPTION F: OLYMPIC CITY

## OLYMPIC CITY

Many of us are thrilled that the 2016 Olympic Games will be hosted by one of the most beautiful cities (if not *the* most beautiful city) in the world: Rio de Janeiro. This was the result of a recent decision by the International Olympic Committee. As I understand, some 115 members of that committee voted for the four finalist cities: Chicago, Madrid, Rio de Janeiro and Tokyo. I also don't know the exact rules of the vote, but I know there were successive rounds, in each case dropping the least voted city, while an absolute majority was not reached.

This is fine, but I want to suggest another method, which yields results faster (if you have a computer…) and is less prone to unduly political distortion that perhaps can occur between the successive rounds in the current scheme.

It works like this. Each voter ranks all the cities linearly, before the voting starts. Then, he or she is bound to vote always according to his or her ranking. On the first round, all voters vote for the first city on their own ranking. If an absolute majority is formed, the winner will have been found. An absolute majority is the situation where one of the candidates gets strictly more than half of the votes. If an absolute majority is not formed, then one of the candidates will be removed and another election takes place among the remaining candidates (i.e., as if the candidate that was removed had never existed). If there's an absolute majority after this, we're done; if not, we remove another candidate and so on, until an absolute majority is formed.

The question is: which candidate to remove, after an election that did not result in an absolute majority? Well, the one who got the least votes, among those who got at least one vote. But wait, there can be a tie: several candidates with the least number of votes (not counting those who did not get any votes). In this case, we carry an election among those who are tied, and drop the one with least votes in this election. Recall that in all elections, each voter votes according to his or her ranking, considering only the subset of candidates that are running on that election. Still, this election for breaking the tie among those that got least votes may result in yet another tie: more than one candidate with the least number of votes. In this case, the president of the Committee uses his tie-breaking vote, and decides which candidate of those tied in the last election shall be removed.

There is another situation in which the president of the Committee is called to break a tie: all candidates with votes get the same number of votes. This may happen either in a “normal” election or in an election for selecting a candidate to be removed.

The president is a trustworthy person but in order for our procedure to be completely automatic, he or she will also always vote according to his or her own ranking, that he or she will have prepared beforehand.

## TASK

Your task is writing a program that, given a list of rankings of all the candidates in an election creates a global ranking of those candidates, using the described procedure, successively, first for the winner, then for the runner-up, then for the one in third position, etc.

One of those rankings is the president’s ranking, and is only used for tie-breaking. The remaining rankings are the rankings submitted by the voters.

The global ranking is created by first choosing the winner of the election, as explained. This will be the first in the global ranking. Then we repeat the procedure, after having excluded the winner, which has already been ranked in first position. The winner of this second election will be ranked second in the global ranking, and so on.

## INPUT

The first line of the input file contains two positive integers, representing the number of voters, NV, and the number of candidates, NC. Exactly NV+1 lines follow. The first of these lines represents the ranking used for tie-breaking. It contains NC unique strings, separated by spaces. Each of the remaining NV lines represents the ranking of one voter. Each contains a permutation of the strings found in the ranking for tie-breaking, also separated by spaces. All the rankings are ordered from first choice to last choice.

## OUTPUT

The output represents the computed global ranking. It lists the names of the candidates, one in each line, in the order of the ranking.

## RESTRICTIONS

The number of voters, NV: NV >= 2, NV <= 1024;

The number of candidates, NC: NC >= 2, NC <= 32;

The names of the candidates are composed of no more that 15 lowercase characters (no diacritics), decimal digits or the underscore character. The names are unique.

## INPUT EXAMPLE

```
8 4
madrid chicago rio tokyo
madrid chicago tokyo rio
chicago rio tokyo madrid
madrid rio tokyo chicago
rio tokyo chicago madrid
madrid rio tokyo chicago
tokyo rio chicago madrid
rio madrid tokyo chicago
```

rio tokyo chicago madrid

## OUTPUT EXAMPLE

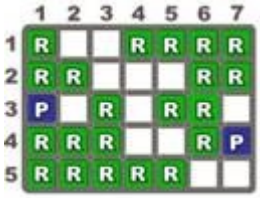
rio
madrid
tokyo
chicago

# PROBLEM DESCRIPTION G: POLICE

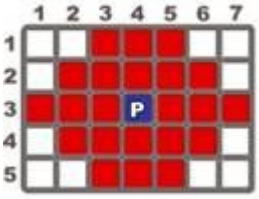
## POLICE STATION

As the mayor of Extremepolis, it is your responsibility to ensure that every citizen feels safe. Your new budget allows you to build a new police station and you wonder
build it.

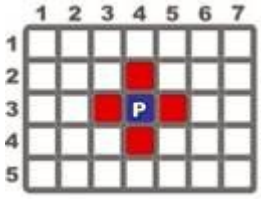
Extremepolis is very well organized in a grid of streets. Each cell represen
city and it is identified by its land use: 'R' means a residential zone and 'P'
station. The following figure illustrates an example city with 5 rows, each c s:



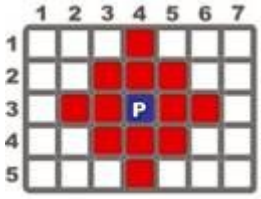

Each police station ensures security for a limited number of residential zones. Given a police range **K**, you know that all residential zones within **K** distance of a police station are covered and therefore safe. Note that since the police cars can only use the streets, the distance is the so called "Manhattan distance": given two cells (y1,x1) and (y2,x2), the distance between them is equal to |y2-y1|+|x2-x1|. The following figures illustrate what cells would be covered by a police station with different ranges:




Range: K = 1

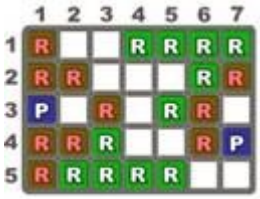



Range: K = 2




Range: K = 3

With this, you can detect which residential zones are already covered by the existing police stations. In the given example, with **K** = 2, they would be these:




What you want is to find the free location (currently empty cell) in which putting a police station would result in the biggest amount of newly covered residential zones (that is, residential zones that were not covered by previously existent police stations and that would become covered by the new police station. In the example, the best position would be (2,5), with 5 new residential zones covered:


Note that there can be several empty cells yielding the same best number of residential zones (in the example, choosing (4,4) would also cover 5 new residential zones). In that case, you should always choose the position with the lowest row number, and in the case of a tie, the one with the lowest column number.

## TASK

Given a police range **K** and a city with **R** rows and **C** columns, your task is to find the best empty location for a new police station, that is, the one that would cover the biggest amount of previously uncovered residential zones. In case of a tie, choose the lowest possible row and if a tie still exists, choose the lowest possible column.

## INPUT

The first line of input contains a single integer **K**, indicating the police cover range. The second line contains two integers **R** and **C**, separated by a single space, indicating respectively the number of rows and number of columns of the city.

Then follow exactly **R** lines, each one of them with **C** chars, describing the city, where each cell is represented by one of the following characters:

**'.'** - an empty cell

**'R'** - a residential zone

**'P'** - a police station

There are no restrictions on the number of residential zones and police stations (they can be from zero to the size of the city), except the fact that there will always exist at least an empty cell which is able to cover a previously uncovered residential zone.

## OUTPUT

The output consists of two lines. The first one should contain two integers **Y** and **X**, separated by a single space, indicating the best position for the new police station, as described before. The second line should contain a single integer indicating the number of newly covered residential zones.

## CONSTRAINTS

$1 \leq \mathbf{K} \leq 10$ - Police cover range

$3 \leq \mathbf{R}, \mathbf{C} \leq 100$ - Number of rows and columns of the city

## SAMPLE INPUT 1

```
2
5 7
R..RRRR
RR...RR
P.R.RR.
RRR..RP
RRRRR..
```

## SAMPLE OUTPUT 1

```
2 5
5
```

## SAMPLE INPUT 2

```
1
4 10
.PRR...RRR
..RR.R..R.
....RR.R..
R...P..R.P
```

## SAMPLE OUTPUT 2

```
2 8
3
```

# PROBLEM DESCRIPTION H: DICE

## DICE

John is preparing for his exam in Statistics. For that, he has to solve many of those classical exercises with dice: if you throw two dice, what is the probability that the sum is, say, greater or equal to 9? John knows how to do it: he lists all the possibilities – <1,1>, <1,2>, ..., <6,6> – and simply counts how many of these pairs satisfy the condition that the sum is greater or equal to 9. In this case the count is 10. There are 36 pairs. Hence the probability is 10/36.

But how would John solve these problems if there were more than two dice, making it very awkward to list by hand all the possibilities?

## TASK

Please write a program the given that given a positive integer, D, representing the number of dice, and another non negative integer, N, computes the probability that when throwing D dice the sum of the values obtained is greater or equal to N. The program shall present the result in the form of irreducible fraction, if the result is not zero or one. In these cases, the result shall be represented by 0 or 1, respectively.

## INPUT

The input file contains one or more lines. Each line displays two positive integers, the number of dice, D, and the number N, separated by a space.

## OUTPUT

For each line of the input file there shall be a line in the output file, displaying the value of the probability as an irreducible fraction or simply 0 or 1, when that is the case, as explained. The fraction shall be represented by two decimal integers separated by a forward slash.

## CONSTRAINTS

The number of lines in the input file is not greater than 1000.

The number of dice: 1 <= D <= 11.

The number N: N >= 0.

## SAMPLE INPUT

```
2 9
1 6
2 0
3 18
```

## SAMPLE OUTPUT

```
5/18
1/6
1
```

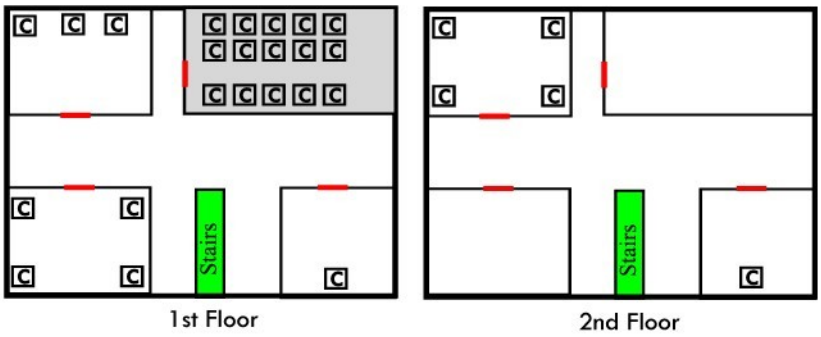
# PROBLEM DESCRIPTION I: PROCTORING

## PROCTORING

This problem is dedicated to the proctors of IEEExtreme. Proctors play an important role in our contest, ensuring all teams follow the rules, no matter where in the world they are competing. Before the actual contest, some proctors also had to organize things at their locations. For example, in some cases, they had to select the computer lab where teams were to work. One proctor, in particular, had a tough time at that. That's the proctor from University of the South Pole.

At USP, the Computer Science building is an unlikely tower, with many floors and no elevators. In each floor, there are some computer labs, with a varying number of computers. Expecting a large number of teams, the proctor wanted the lab with most computers. Searching in person would be very tiresome, but, fortunately the administration has a map of each floor in digital format, identifying individual computers, and so it is possible to just compute the result.

Consider the maps in the figure. The best lab is the one marked in grey. It has 15 computers. Should there be two labs on different floors with the same maximum number of computers, we prefer the one at the lower floor, for it is easier to get there. If two labs on the same floor have the same maximum number of computers, either will do.




## TASK

Given an ASCII description of the map of each floor of the computer science building at the University of the South Pole, find the room with the largest number of computers and report the floor number. In case of a tie, the lowest floor is preferred.

## INPUT

The first line of the input file contains a positive integer, N, representing the number of floors of the building. The following lines contain the ASCII maps of the floors. One map is separated from the next by one blank line. The maps are given in ascending order of floor, i.e., the first map will be from the first floor and so on. There is no blank line after the last line of the last floor.

In each map, an asterisk represents a wall, the letter ‘D’ represents a door, the letter ‘C’ represents a computer, and the letter ‘S’ represents stairs.

All labs have walls (and doors) surrounding them and the entire floor is closed (by walls). However, the labs and the floor may have any shape, not necessarily rectangular. There is at least one lab in each floor.

## OUTPUT

The output file contains a single line, displaying two numbers separated by a space: the lowest floor number where a lab with the maximum number of computers exists, and the number of computers in that lab.

## CONSTRAINTS

Number of floors, N: 1 <= N <= 10.

Maximum number of characters per line in the map: 500.

Maximum number of lines in a map: 500.

## SAMPLE INPUT

```
2
************************
*C C C *   * C C C C C  *
*     *   * C C C C C  *
*     *   D          *
*     *   * C C C C C  *
***DD***   **************
*                  *
***DD*** S *****DD*******
* C  C * S *          *
*     * S *          *
* C  C * S *    C     *
************************

************************
* C  C *   *          *
*     *   *          *
* C  C *   D          *
*     *   *          *
***DD***   **************
*                  *
***DD*** S *****DD*******
*     * S *          *
*     * S *          *
*     * S *    C     *
************************
```

## SAMPLE OUTPUT

```
1 15
```

# PROBLEM DESCRIPTION J: XTREBBLE

## XTREBBLE

I have just invented a new game. It's called Xtrebble. I am sure nobody has ever thought of anything similar. The rules are simple, but I suspect the game is very enticing, and I wouldn't be surprised if it becomes a huge success and I become very famous for having invented it.

Xtrebble is played with tiles. Each tile displays a letter. Each letter has a value. With tiles, players compose words (what else can we do with letters?) The value of a word is the sum of the values of its letters.

All players are given some tiles. The goal of each player is to form a valid word which is as valuable as possible, using only his tiles. I anticipate that there will be a lot of discussion on whether a word is valid or not and decided to provide an official dictionary: a valid word is a word that is present in the dictionary. All other words are not valid.

To make the game even more interesting and challenging, I invented the concept of a joker. This is a special tile which can be used to represent any letter, but its value is zero. That is, jokers can help forming words, but add no value to the value of the other letters.

## TASK

Write a program that given the value of each letter, given the tiles a player has and given the official dictionary, finds the most valuable word the player can form.

## INPUT

The first two lines of the input are used for setting the value of each letter. They are organized as follows: the first line contains all 26 Latin lowercase letters, in alphabetical order, each letter separated from the next by one space. On the second line there are 26 positive one-digit integers placed exactly below each letter. Each number represents the value of the letter right above it.

The third line of the input contains one positive integer, P, the number of players.

Each of the next P lines contains a string formed by lower-case Latin letters and the uppercase letter 'J'. The i-th string represents the tiles of the i-th player. The uppercase 'J' represents the joker tile.

The line following those P lines contains a positive integer, W, representing the number of words in the dictionary. Each of the next W lines contains one string, entirely composed of lowercase Latin letters, representing a valid word. These words are in any order.

## OUTPUT

The output file contains P lines, one for each player. The i-th line of the output displays the most valuable word the i-th player can produce and its value. The value of the word comes first and is separated from the string by one space. If it is impossible to form any word, the line will contain the word “pass” (without the quotes). If two or more words have the same maximum value, we prefer the one that comes first in alphabetical order.

## CONSTRAINTS

Number of players, P: 1 <= P <= 100.

Number of words in dictionary, W: 1 <= W <= 30000.

Maximum length of any word in dictionary: 25.

Maximum number of tiles held by a player: 25.

## SAMPLE INPUT

```
a b c d e f g h i j k l m n o p q r s t u v w x y z
1 2 2 2 1 2 2 2 1 2 2 2 2 2 1 2 2 2 2 2 1 2 2 2 2 2
3
tac
butterJJJ
xyz
8
dog
cat
lion
butterfly
whale
shark
eagle
pig
```

## SAMPLE OUTPUT

```
5 cat
10 butterfly
pass
```

# PROBLEM DESCRIPTION K: FOCUS LIST

## THE RETURN OF THE FOCUS LIST

Our company has asked a consultancy team to develop a new “Focus List” functionality: the focus list solution deployed last year was so efficient in its processing speed that MRM (Market Risk Management) is interested to exploit the functionality for its historical analysis. FYI[1], the request made last year was:

“The focus list is a list of financial instruments (shares and indexes) the trader must pay special attention to: the trader has to mainly monitor the volatility and dividend table changes.

Years ago, we developed under Windows a small C++ command prompt application that lists the focus instruments. At that time, the number of focus elements was less than 30. Currently the list is more than 2,000 elements and the pressure from the trader to improve the application is everyday more insistent.

After talking with the traders, we extracted the following function to implement: the trader would like to dynamically filter the list on the instrument name, the Reuters and/or Bloomberg code with characters and some wildcards: ‘*’ for multiple characters and ‘?’ for single character). Furthermore, the trader wants to use three other special characters, as follows:

- ‘~’ to display the previous page;
- ‘#’ to display the next page;
- ‘^’ to reset the filter string.”

This year, the MRM stated their request as follows:

“We would like your team to start from the focus list solution to filter historical market data: the market data are the instrument spot prices provided by Reuters and Bloomberg; these are generally daily figures. The market data will be downloaded from the live database and extracted in a text file comma separated. As we are dealing with the Bloomberg and Reuters code MRM believe the right starting point is to use the previous focus list and adapt it to meet some new input requirements.”

## TASK

Your task, should you decide to accept it, is to take the embedded solution of last year and adapt it to fit the MRM requirements. You will be provided with the last year’s project.

## INPUT FORMAT

Last year the input format was encoded in scheme list as showed below:

(10(Name "CAC 40 INDEX")(Reuter ".FCHI")(Bloomberg "CAC Index")(LastDivUpdate 20080118)(LastVolUpdate 20080121))

The requested MRM file format for this year is as follows:

<Sicovam>,<JulianDate>,<Name>,<BloombergCode>,<BloombergSpot>,<ReutersCode>,<ReutersSpot>

## OUTPUT FORMAT

The file output file format for the filtered elements is as follows:

<Sicovam>,<CR/LF>

,<BloombergCode>,<ReutersCode><CR/LF>

[<Date format YYYY-MM-DD>,<BloombergSpot>,<ReutersSpot> <CR/LF>]$^{1...n}$

This means that the output file is composed of several groups of lines, one for each filtered element. The first line of the group has the “sicovam” number and the name, separated by a comma; the second line has the two codes, separated by a comma, but notice that, for some unknown reason, the line also starts by a comma; an undetermined number of lines follow, with a date in the format YYYY-MM-DD and two numeric values, all three fields separated by commas. The numeric values may be missing (but not the commas).

There shall be no space characters in the lines of the output file, other than those embedded in the Bloomberg code or in the Reuters code.

## IMPORTANT NOTICE

Unlike the project from last year, this task is to be implemented by a program that gets all its data from the standard input and writes all the results to the standard input. This implies that the filter string comes in the input file, before the actual data, and not in the command line, as before. Also, the component that deals with displaying results on the screen is not required this year.

## IMPORTANT UPDATE FROM THE JUDGE:

Make sure you check the new sample input and output (the last one was wrong). Also, please make sure that all real numbers in your output (BloombergSpot and ReutersSpot) have exactly 6 decimal digits.

## SAMPLE INPUT

See example http://w3.ualg.pt/~pjguerreiro/IEEExtreme/fl_in_01.txt.

## SAMPLE OUTPUT

See example at http://w3.ualg.pt/~pjguerreiro/IEEExtreme/fl_out_01.txt.

## NOTES

Please download the project from last year
from http://w3.ualg.pt/~pjguerreiro/IEEExtreme/IeeExtreme_2009_Proposal.zip.

The provided solution, from last year, may contain bugs. You are supposed to correct them, as part of the task, if they interfere with the assignment.

The provided projects are NOT Mooshak-friendly programs: they get data from named files (not from the standard input) and take arguments from the command line.

The sample input file is quite large. Your program may take a few seconds to process it. For testing inside Mooshak, assume the size of input file is typically around 5% the size of the sample input.

Your submission must consist of a single source file, as in the other problems. You can include code from the provided solution, at your will, but you must copy it to the submitted file.

The provided project is a Microsoft Visual C++ project, and the source files may contain languages features that are Microsoft specific. Beware!

Although the provided project is a Microsoft Visual C++ project, you may use any of the authorized languages, but, if you do, you will not be able to reuse the code directly, of course.

You may assume all test cases yield a non-empty output file.

[1] FYI = for your information.

# PROBLEM DESCRIPTION L: QUALITY

## QUALITY OF WATER

Have you checked problem Mineral Water? This is a “special problem”, whose grading can only be made after the contest in over, since the score of each submission depends on its result as well as on the results of all the others.

Scoring is problem Mineral Water is explained as follows:

For each team that sent the solution for a test, the sum of qualities of fields that are in range of some well will be calculated. Note that this sum can be negative. Each field will be counted only once even if it is in range of several wells. Then teams will be sorted according to this sum. Let t denote the number of teams. The best team will receive 10 points for that test. Next (t-1) div 9 teams will receive 9 points for that test. Next (t-1) div 9 teams will receive 8 points, etc. In the end (t-1) div 9 + (t-1) mod 9 teams will receive 1 point.

We can understand this, but some clarifications are required. First, the description concerns one test case only. The score of a team in the problem is the sum of points for all tests cases to which the team submitted a solution.

Then, the description assumes that there are no ties, i.e., that no two teams reach exactly the same sum of qualities in the test case. This is a simplification, because ties do happen. Suppose two teams, A and B, achieve the same sum, and, because the sorting procedure, A was the last team with 8 points and B was the first with 7 points. This would be tremendously unfair for B. Therefore, we stipulate that teams with the same sum of qualities will get the same number of points as the first team with that sum of qualities. A limit case is when all teams have the same sum of qualities: then they all should get 10 points.

Another clarification is needed when the number of teams submitting to some test case is less than 10. Say there are 4 teams submitting solutions to a given test case. Then, applying the above rule strictly, the best team would get 10 points and all the others 1 point. We do not want that. Instead, the second best team should get 9 points, the third should get 8 points and the forth should get 7 points. So, we can add the rule that when there are less than 10 teams, the points are awarded sequentially, from 10 downwards, in decreasing order of the sum of qualities. Still, if there is a tie, the previous rule still applies.

The examples included below (with the sample inputs) further clarify these issues.

## TASK

Write a program that given the sums of qualities obtained by the teams in the tests cases to which they submitted solutions computes the number of points for each team, using the procedure described.

## INPUT

The first line of the input file contains one positive integer, N, representing the number of tests. An indeterminate number of lines follow, one for each submission. Each of these line contains three items of information, separated by spaces: a string, S, representing the team name, a positive integer, X, representing the test number, and an integer, Y, representing the sum of qualities, for that team in that test.

## OUTPUT

The output file shall have a line for each team, displaying the team name and the points obtained. The file shall be sorted by decreasing number of points and then by team name.

## CONSTRAINTS

Maximum length S, the team name: 127.

Number of tests, N: 1 <= N <= 10.

Value of X, the test number: 1 <= X <= N.

Maximum number of teams is 512.

The sum of qualities, Y: any integer number, positive or negative.

Uniqueness: no team submits more than once to any test case.

## SAMPLE INPUT 1

```
1
john 1 22
peter 1 26
mary 1 22
ann 1 18
paul 1 27
julia 1 20
charles 1 28
bob 1 23
cameron 1 26
kyle 1 20
michele 1 19
george 1 18
james 1 17
bill 1 31
ronald 1 23
fred 1 31
catherin 1 22
caroline 1 18
sophia 1 20
victor 1 19
lucy 1 26
lise 1 24
laura 1 27
richard 1 22
```

hector 1 26
louis 1 20
mike 1 16
david 1 29
lynn 1 24
mercedes 1 28

## SAMPLE OUTPUT 1

bill 10
fred 10
charles 9
david 9
mercedes 9
laura 8
paul 8
cameron 7
hector 7
lucy 7
peter 7
lise 6
lynn 6
bob 5
catherin 5
john 5
mary 5
richard 5
ronald 5
julia 3
kyle 3
louis 3
sophia 3
michele 2
victor 2
ann 1
caroline 1
george 1
james 1
mike 1

## SAMPLE INPUT 2

2
cameron 1 40
kyle 1 23
michele 2 14
james 2 15
michele 1 38
george 1 23
james 1 40
mary 2 10

## SAMPLE OUTPUT 2

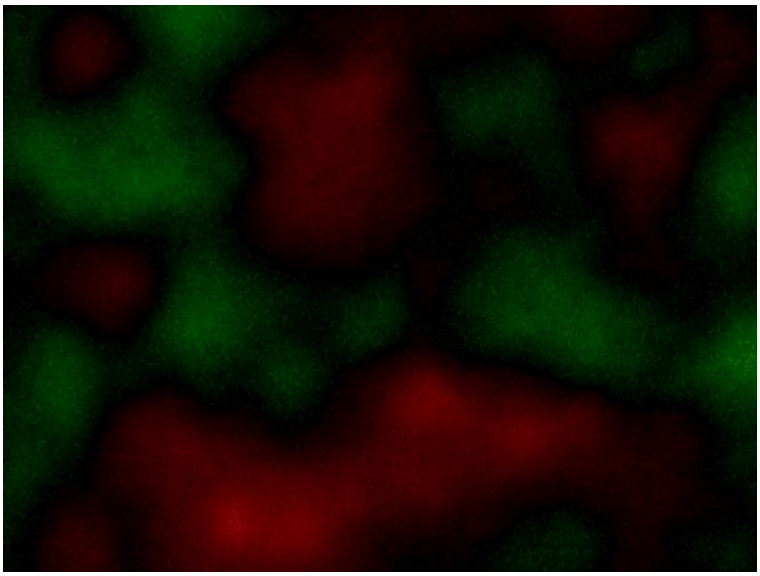
james 20
michele 17
cameron 10
mary 8
george 7

# BONUS QUESTIONS

## PROBLEM DESCRIPTION A: MINERAL WATER

### MINERAL WATER

I would like to found a company that will extract mineral water from underground sources, pour it out into bottles and sell it. To determine the best location for wells from which I will take the water I created a geological map of the land that I have. I divided it into many square fields and using advanced technology I determined the quality of the underground water that is below each square. The quality is described by the number from the range from -255 to 255inclusive. I saved this information in the PNG image where each pixel represents one field. Fields with positive quality *q* are represented by green color with RGB values (0, q, 0) and fields with negative quality *q* are represented by red color with RGB values (-q, 0, 0). The example map is presented below.



I can build several types of wells. Each well has its range radius *r*. The radius *r* means that when I build a well on the field with coordinates (x, y) then it will gather water from all fields that have Euclidean distance from this field less or equal to *r*. The profit of my company will be proportional to the sum of qualities of fields that are in range of some well.

### TASK

Please help me determine the best location for wells to maximize my profit.

### INPUT

With this task you received ten images saved in the PNG format and ten text files. Names of the files are mwaterXX.png and mwaterXX.txt where XX is the test number. The PNG files contain geological maps presented as described above. The (0, 0) coordinates are located on the first pixel in the left, top corner. Text file in the first line contains integer number $3 <= n <= 50$ - the number of wells that I want to build. In the second line there are exactly *n* integer numbers denoting ranges of wells that I want to build. Example text file is following:

3

10 30 40

## OUTPUT

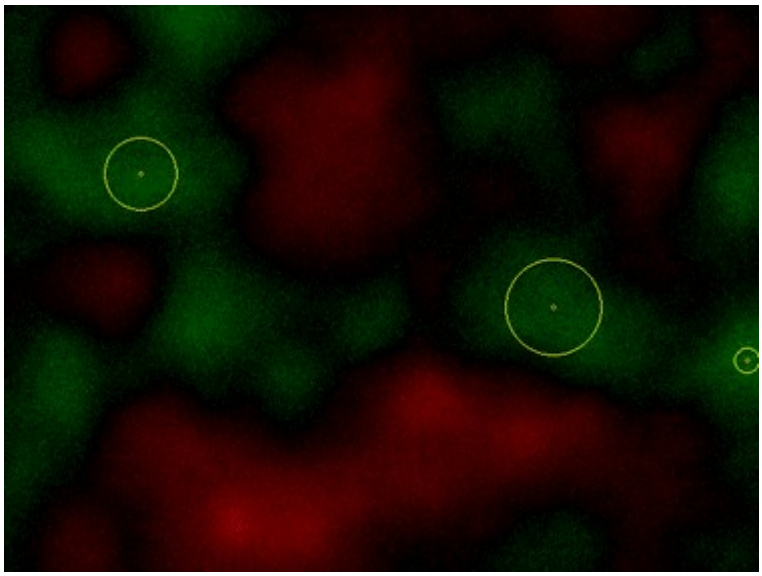
For each pair of input files generate one output file named mwaterresXX.txt. It should contain *n* lines. *i*-th line should consist of two numbers - *x*, *y*coordinates of the *i*-th well. Example output (not necessarily the best one) for the above input file is following:

629 300

115 142

465 255

The visualization of the above output is presented below. Small circles represent location of wells and yellow circles theirs ranges.



## SCORING

For each test the following scoring procedure will be used. For each team that sent the solution for that test the sum of qualities of fields that are in range of some well will be calculated. Note that this sum can be negative. Each field will be counted only once even if it is in range of several wells.

Then teams will be sorted according to this sum. Let *t* denotes the number of teams. The best team will receive 10 points for that test. Next *(t-1) div 9* teams will receive 9 points for that test. Next *(t-1) div 9* teams will receive 8 points, etc. In the end *(t-1) div 9 + (t-1) mod 9* teams will receive 1 point.

## SUBMISSION

Your results files should be submitted together in a single zip file, named mwaterres.zip. If you submit a file with another name, the scoring procedure may not find it. You are allowed to make more than one submission: only the more recent counts, the others are ignored.

# PROBLEM DESCRIPTION B: COOKIES RECIPE

## NEIMAN-MARCUS $250 COOKIES RECIPE

Instead of paying “two fifty” for the secret recipe, my collaborating cashier agreed to give me a secret nod when all the five different ingredients needed for making the cookies are on the conveyor belt.
There are ten different possible ingredients: apple, butter, chocolate, decaf, egg, flour, gelatine, honey, ice, jam.
So all I have to do is arrange them on the conveyor belt – which is exactly five items long – and make sure that all possible combinations are covered in the shortest possible sequence.

## TASK

Please write a program that computes the shortest sequence of ingredients you can find which guarantees an affirmative nod from our cashier.

## INPUT

None.

## OUTPUT

A single line with a list of lowercase letters: each letter is the first letter from the ingredient it represents.

## SCORING

The following C program returns the score of your program (the higher the better). The program with highest score will get 150 points and the program with the lowest score larger than zero will get 50 points. All others get a proportional number of points.

```
#include <stdio.h>

int hash(char vec[6]) {
        int sorted[6];
        int i,j,k,s;

        for (i=0; i<6; sorted[k] = vec[i++]-'a')
                for (k=j=0; j<6; ++j)
                        if (i!=j && vec[i]==vec[j]) return 0;
                        else    k += vec[i]<vec[j];
        for (s=i=0; i<5; ++i) s = 10*s+sorted[i];

        return s;
}

main() {
        char    c;
        char    last5[6]={'@','@','@','@','@','@'};
        int     num_appear=0, hash_appear[100000];
        int     i,i5=0;

        for (i=0; i<100000; ++i) hash_appear[i]=!i;
        for (i=0; i<500 && (c=getchar()) <= 'j' && c>='a'; ++i) {
                last5[i5] = c;
                i5 = (i5+1)%5;
                num_appear += !hash_appear[hash(last5)]++;

        }
```

```
        return (num_appear<252) ? 0 : 500-i;
}
```

# PROBLEM DESCRIPTION C: MATH EXERCISE

## MATH EXERCISE

There is a nice game one can play with phone numbers: trying to make an exercise from it whose result is 100. The rules of the games are that you can add the four basic math operations (+, -, *, /) and parentheses between the digits such that the result is a valid exercise whose result is 100.

Note that you must use all the digits, keep the order of the digits and you are not allowed to combine several digits into a number. Thus the solutions “5*5*4”, “5*5*4+0*1*5*5” and “5*(5+5+1+0)+45” are not a valid ones for the number 5551045, while the solution “5*(5+5+1+0+4+5)” is.

## TASK

Write a program that given a seven digit decimal number, finds an exercise whose result is 100. (Leading zeros are acceptable.)

## INPUT

The input file contains 100 exercises, one per line. Each line consists of exactly seven decimal digits.

## OUTPUT

For each line in the input, there shall be a line in the output, featuring a valid exercise using the digits of the input line, and characters from the set [‘+’, ‘-‘, ‘*’, ‘/’, ‘(’, ‘)’], as explained.

## INPUT SAMPLE

The following is an example of the first three lines of an input file:

```
5750004
8682395
0277770
```

## OUTPUT SAMPLE

The following output example solves two out of the three examples in the input sample:

```
5*/*7*/5*(0+0+0+4)
(-8+6+8+2+3+9)*5
0-(2/7+7+7)*7+0
```

## NOTE

You must keep the original digits in order and add characters from the following six long alphabet: “+-*/()”. If you cannot solve the exercise, leave the output the same as the input; otherwise you may lose points.

## SCORING

One point will be given for each valid solution.

The validity would be checked using the following script:

```
#/bin/csh
# Make sure we are using only the input digits and allowed operators
tr -d '+\-*/()' < output | diff input -
if ($? == 1) exit 1
# Make sue no two digits are left without an operator between them
grep '[0-9][0-9]' output
if ($? == 0) exit 1
# Once we passed the preconditions above,
# Score according to number of exercises solved correctly
sed 's/$/-99.99999999/' output | bc -l | grep -c '^.00000'
```

A zero exit code is considered a valid solution.

## TRICKY NOTE

Pay close attention to the scoring script – can you foul it? Remember: your score is determined by the script, so if it accepts it (even though it is not what we meant) – you win the points.

# PROBLEM DESCRIPTION D: VHDL

## MAXIMUM SUB SEQUENCE SUM

The first 10 teams uploading a working solution for this problem will receive extra points. :-)

In this problem, you will be given a sequence of 10 signed integer numbers. Your task is to find a (not necessarily unique) sub sequence of consecutive numbers producing a maximal sum if added together.

Example:

Sequence: 1, 2, -5, 6, -1, 7, -10, -1, 2, 4

Sub sequence with maximal sum: 6 + (-1) + 7 = 12

Sounds easy? Well, in this problem you will not be using C, C++ or Java – actually you will not be using a programming language intended for a normal CPU. Instead of this, you will be using VHDL. VHDL is an ADA based language commonly used to implement software running on field-programmable gate arrays (FPGAs). Lots of information about this language can be found on the internet. As a starting point you can use:

http://en.wikipedia.org/wiki/VHDL

Your task is to create a VHDL module receiving 10 numbers. After some (limited) computation time, the module returns the maximum sub sequence sum as described above (the location of the sub sequence does not need to be returned).

As a VHDL tool we recommend GHDL which is available free of charge for Windows as well as Linux at http://ghdl.free.fr/.

Your solution will be evaluated after the competition using an automated test bench. For testing your solution locally, you can download a copy of this test bench from:

http://w3.ualg.pt/~pjguerreiro/IEEExtreme/MaxSeqSumTestBench.txt

Please do not modify this file and do not upload it to the Mooshak server, but please change the extension from .txt to .vhd before compiling it, as explained below. Submitted solutions will be evaluated using a fresh copy of this file. Modifications to the test bench will likely render your solution incompatible to the original test bench used for evaluation.

In case you are using GHDL, you can run a simulation of your module (i.e. execution of the module in software) using the following commands (assuming your module is named MaxSeqSum.vhd):

ghdl -a MaxSeqSumTestBench.vhd

ghdl -a MaxSeqSum.vhd

ghdl -r MaxSeqSumTestBench --stop-time=10000ns

## MODULE INTERFACE

To be compatible to the test bench you must stick exactly to the following interface:

```
 entity MaxSeqSum is
port (
    clock : in STD_LOGIC;
    sreset : in STD_LOGIC;
    input_valid : in STD_LOGIC;
    value_1 : in STD_LOGIC_VECTOR (31 downto 0);
    value_2 : in STD_LOGIC_VECTOR (31 downto 0);
    value_3 : in STD_LOGIC_VECTOR (31 downto 0);
    value_4 : in STD_LOGIC_VECTOR (31 downto 0);
    value_5 : in STD_LOGIC_VECTOR (31 downto 0);
    value_6 : in STD_LOGIC_VECTOR (31 downto 0);
    value_7 : in STD_LOGIC_VECTOR (31 downto 0);
    value_8 : in STD_LOGIC_VECTOR (31 downto 0);
    value_9 : in STD_LOGIC_VECTOR (31 downto 0);
    value_10 : in STD_LOGIC_VECTOR (31 downto 0);
    idle : out STD_LOGIC;
    output_valid : out STD_LOGIC;
max_sum : out STD_LOGIC_VECTOR (31 downto 0)
);
end MaxSeqSum;
```

## REQUIRED BEHAVIOR

The clock signal is active at the rising edge of the signal. All other signals are synchronized and considered to be valid at the rising edge of the clock signal only. There must be no combinational paths through the module. When signal sreset is asserted (= 1), the module should clear its internal state and return to an idle state regardless of all other inputs and its current state. The module needs to signal its idle state (i.e. the module can accept a new set of numbers at the next rising clock edge) by asserting the idle signal.

While the module is processing a sequence, the idle signal must be cleared (the module processes only one set of numbers at a given time). When processing of a sequence finished, the output_valid signal has to be asserted for exactly one clock cycle and the computed maximum sum being provided on the signal bus max_sum during the same clock cycle.

Whenever the module is in idle state, a new sequence can be sent to it by asserting the input_value signal and providing the numbers on the value_* buses.

All numbers are encoded in two’s complement representation. You can assume that the input numbers provided are out of the range -1,000,000 to 1,000,000.

Computation must finish after a **maximum of 100 clock cycles**. The minimum allowed computation time is 1 clock cycle (i.e. the module must be registered – a combinational module is not allowed). After computation finished, the module must **return to idle state after at most 10 clock cycles**.

Your module must only use standardized VHDL packages as defined in IEEE standards 1076.3 and 1164 (namely packages IEEE.NUMERIC_STD and IEEE.STD_LOGIC_1164; these are readily included in the GHDL distribution).