

Problem 1 (Cont.)

any input will loop
until 1

It is conjectured that the steps above will terminate (when a 1 is left) for any input value. Despite the simplicity of the process, it is unknown whether this conjecture is true. It has been verified, however, for all integers n such that $0 < n < 1,000,000$ (and, in fact, for many more numbers than this.)

Given an input n , it is possible to determine the number of times this algorithm is repeated before completed. For a given n this is called the iteration-length of n . In the example above, the iteration length for the input n where $n = 7$ is 17.

For any two numbers n_1 and n_2 you are to determine the maximum iteration length over all numbers between n_1 and n_2 .

Input

The input will consist of a series of integer pairs $\overline{n_1}$ and $\overline{n_2}$, one pair of integers per line. All integers will be less than 1,000,000 and greater than 0. Such that:

$$0 < n_1 < n_2 < 1,000,000$$

You should process all pairs of integers and for each pair determine the maximum iteration length over all integers between and including n_1 and n_2 .

You can assume that no operation overflows a 32-bit integer. The input will end with a 0.

binary 32
char
str

Output

For each pair of input integers $N1$ and $N2$ you should output $N1$, $N2$, and the maximum iteration length for integers between and including $N1$ and $N2$. These three numbers should be separated by at least one space with all three numbers on one line and with one line of output for each line of input. The integers $N1$ and $N2$ must appear in the output in the same order in which they appeared in the input and should be followed by the maximum iteration length (on the same line).

while(

Problem 1 (Cont.)

Sample Input

```
1 10
100 200
201 210
900 1000
0
```

Sample Output

```
1 10 20
100 200 125
201 210 89
900 1000 174
```

Problem 2: Lilavati and Bhaskaracharya

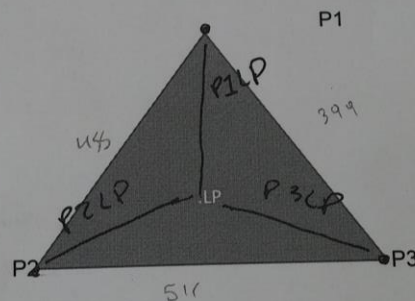
The year is 1150 sometime in the evening an intense debate is going on between the father (Bhaskaracharya) and daughter (Lilavati) – both famous Indian mathematicians. This debate leads to a challenge that Lilavati poses for her father. She also goes on to add that if her father successfully solves the problem then she would allow him to name his thesis *Lilavati* (*Lilavati* is Indian mathematician Bhaskaracharya's thesis on mathematics, written in 1150.). So what is the problem that she poses to her father?

Problem

Three Persons, namely P1, P2 and P3 are standing on the vertexes of a triangular field whose all interior angles are less than 120 degrees (as shown below, figure not to scale). Somewhere inside the field is the point called the Lilavati Point (where she is standing) denoted by LP in the figure. The Lilavati Point is defined as below:-

Lilavati Point (LP): - This is the point where all 3 persons meet (P1, P2 & P3) Lilavati such that the sum of P1LP (the distance between P1LP), P2LP (the distance between P2LP) & P3LP (the distance between P3LP) is the least possible total distance traversed by the three persons.

Point (x, y) ?



Example

If P1P3 (length of side P1P3 of triangle P1P2P3) = 399, P1P2 (length of side P1P2 of triangle P1P2P3) = 455, and P2P3 (length of side P3P2 of triangle P1P2P3) = 511. Then LP is a point where $P1LP + P2LP + P3LP = 784$. Also let's call the triangle P1P2P3 above the Lilavati Triangle.

Now coming back to the challenge: Lilavati asked Bhaskaracharya to find the sum of all distinct values of $P1LP + P2LP + P3LP \leq n$ for Lilavati triangles.

$$P_1, P_2 =$$

$$LP(P_1 + P_2 + P_3) \leq n$$

Problem 2 (Cont.)

Input

The input will consist of a single integer n on each line that will denote the max sum of all distinct Lilavati triangles. The input will end with a 0 to denote the end of the inputs.

Output

Print out the sum of all Lilavati triangles that are less than the given input n for each n given on a separate line.

Sample Input

12000
6000
5000
0

$$\frac{n}{LP} > (P_1 + P_2 + P_3)$$

Sample Output

30758397
55177
33887

while ((cin >> int) != 0)

{

Problem 3: Hardy Challenge

British mathematician G. H. Hardy and Indian Mathematician Srinivas Ramanujan were great friends and occasionally used to challenge each other to uncover the intricacies and elegance of mathematics.

In 1918, Hardy and Ramanujan studied the partition function $p(n)$ extensively and gave a non-convergent asymptotic series that permits exact computation of the number of partitions of an integer.

In number theory, the partition function $p(n)$ represents the number of possible partitions of a natural number n , which is to say the number of distinct ways of representing n as a sum of natural numbers (order is irrelevant). By convention $p(0) = 1$, $p(n) = 0$ for all negative numbers of n .

During his study, the famous Mathematician Ramanujan was challenged by his Friend Hardy to solve extremely difficult partitions but for simplicity we will only provide simpler cases.

Problem

A natural number, N , that can be written as the sum and product of a given set of at least two natural numbers, $\{a_1, a_2, \dots, a_k\}$ is called a product-sum number: $N = a_1 + a_2 + \dots + a_k = a_1 \times a_2 \times \dots \times a_k$.

For example, $6 = 1 + 2 + 3 = 1 \times 2 \times 3$

For a given set of size, k , we shall call the smallest N with this property a minimal product-sum number.

The minimal product-sum numbers for sets of size, $k = 2, 3, 4, 5$, and 6 are as follows.

$$k=2: 4 = 2 \times 2 = 2 + 2$$

$$k=3: 6 = 1 \times 2 \times 3 = 1 + 2 + 3$$

$$k=4: 8 = 1 \times 1 \times 2 \times 4 = 1 + 1 + 2 + 4$$

$$k=5: 8 = 1 \times 1 \times 2 \times 2 \times 2 = 1 + 1 + 2 + 2 + 2$$

$$k=6: 12 = 1 \times 1 \times 1 \times 1 \times 2 \times 6 = 1 + 1 + 1 + 1 + 2 + 6$$

Hence for $2 \leq k \leq 6$, the sum of all the minimal product-sum numbers is $4+6+8+12 = 30$; note that 8 is only counted once in the sum.

In fact, as the complete set of minimal product-sum numbers for $2 \leq k \leq 12$ is $\{4, 6, 8, 12, 15, 16\}$, the sum is 61 .

What is the sum of all the minimal product-sum numbers for $2 \leq k \leq n$?

$$k=7: 12$$

$$k=8: 1 \times 1 \times 1 \times 1 \times 1 \times 1 \times 2 \times$$

$$k=12: 16 = 1 \cdot 1 \cdot 1 \cdot 1 \cdot 1 \cdot 1 \cdot 1 \cdot 2 \cdot 2 \cdot 2 \cdot 2$$

$$\sum \frac{n(n+1)}{2}$$

$$2^5$$

Problem 3 (Cont.)

Input

The input will start with a number i which will define how many different inputs there will be. The next i lines will be the value n which is the max limit for k .

Output

Output a single line for each different input for the sum of all minimal product-sum numbers.

Sample Input

```
4
1400
986
700
9000
```

Sample Output

```
168287
89993
51245
4476437
```

Problem 4: Cartesian Coordinates

The French mathematician and philosopher René Descartes developed a system in 1637 called the Cartesian coordinate system. A Cartesian coordinate system is a coordinate system that specifies each point uniquely in a plane by a pair of numerical coordinates, which are the signed distances to the point from two fixed perpendicular directed lines, measured in the same unit of length. Each reference line is called a coordinate axis or just axis of the system, and the point where they meet is its origin, usually at ordered pair $(0, 0)$. The coordinates can also be defined as the positions of the perpendicular projections of the point onto the two axes, expressed as signed distances from the origin.

Problem

In order to test the understanding of this new system Cartesian asked one of his students to determine if arbitrary shapes constructed by specifying its co-ordinates overlap each other. Write a program that analyzes shape boundaries and determines whether they overlap. Two shapes are considered to overlap if one contains part of the other, if they share a portion of a common side or if they share a common corner.

Input

The shapes will be designated by their (x, y) co-ordinates. Only the lower left and top right corners of the shape will be provided with origin $(0, 0)$ as the reference. For simplicity we will only consider the positive scenarios.

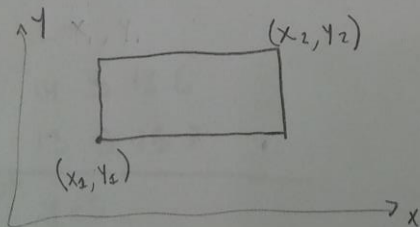
The input will consist of a line containing the number of shapes that we have to consider, followed by one line per Shape in the format x_1, y_1, x_2, y_2 where (x_1, y_1) is the lower left hand corner of the rectangle and (x_2, y_2) is the upper right hand corner of the rectangle.

You may also assume that all shapes within a 20 by 20 area and coordinates range between $(0, 19)$.

Output

Yes or No based on if the shapes overlap. Print YES if they do, otherwise print NO.

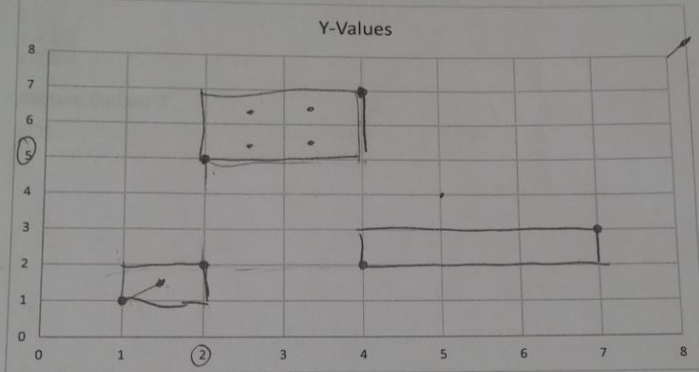
input numShapes,



2-D Array
Double For
loop
for (x)
for (y)
P

Problem 4 (Cont.)

Example



Sample Input 1

3 ← num shapes
 1 1 2 2 (x₁, y₁) (x₂, y₂) ←
 2 5 4 7
 4 2 7 3

Sample Output 1

NO

2

14 3 18 6

13 5 16 7

YES

cin >> numShapes

20 x 20

[][] = NULL

for (x)

for (y):

for (x₁ → x₂)

for (y₁ → y₂)

true

Problem 4 (Cont.)

Sample Input 2

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

Sample Output 2

```
YES
```

Problem 6: Key to Infinity

Vampire numbers first appeared in a 1994 post by Clifford A. Pickover and the article he later wrote was published in chapter 30 of his book *Keys to Infinity*.

Problem

In mathematics, a vampire number (or true vampire number) is a composite natural number v , with an even number of digits n , that can be factored into two integers x and y each with $n/2$ digits. These numbers can't BOTH have trailing zeroes. v contains precisely all the digits from x and from y , in any order, counting multiplicity. x and y are called the fangs.

For example: 1260 is a vampire number, with 21 and 60 as fangs, since $21 \times 60 = 1260$. However, 126000 (which can be expressed as 21×6000 or 210×600) is not, as 21 and 6000 do not have the correct length, and both 210 and 600 have trailing zeroes. Similarly, 1023 (which can be expressed as 31×33) is not, as although 1023 contains all the digits of 31 and 33, the list of digits of the factors does not contain all the digits from the original number.

But for the purposes of this problem we will relax the requirements of this problem and allow x and y to have differing number of digits and v to have any number of digits (both even and odd)

For example: $126 = 6 \times 21$ and $10251 = 51 \times 201$

Given a number N , find the smallest Vampire Number which is greater than or equal to N .

Input

There will be several test cases in the input. Each test case will consist of a single line containing a single integer N ($10 \leq N \leq 1,000,000$). The input will end with a line with single 0.

Output

For each test case, output a single integer on its own line, which is the smallest Vampire Number which is greater than or equal to N .

Problem 6 (Cont.)

Sample Input

```
10
126
127
5000
0
```

Sample Output

```
126
126
153
6880
```

Problem 7: The Shape Fascination

Bob, a 4 year old kid, is very fond of certain shapes on certain days and no amount of convincing can change his mind! So one day his mother while teaching him about shapes, gave him some square cards with pictures of some of his favorite cartoon characters on them, but that was the day when he wanted nothing but rectangular cards and his mother did not have rectangular ones with her. So she asked Bob to make rectangular cards with the square ones she gave him but she added that bob has to use all the cards every time he tries to form a rectangular card.

Problem

Your challenge is to help bob to find out the maximum number of unique rectangles he can make with the available square cards.

For example: you can form exactly 2 unique rectangles with 6 1×1 square cards (Remember you have to use all cards) 1×6 and 2×3 (6×1 is considered the same as 1×6 . Likewise, 3×2 is same as 2×3). Similarly you can also form 2 unique rectangular cards with 4 square cards, using all of the cards: 1×4 and 2×2 .

Input

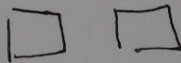
There will be several test cases in the input. Each test case will consist of a single line containing a single Integer N ($1 \leq N \leq 75$) which represents the number of desired rectangles. The input will end with a line with single 0.

Output

For each test case, output a single integer on its own line, representing the smallest number of square cards needed to be able to form exactly N rectangular cards and no more and using all cards. The answer is guaranteed to be at most 10^{18} .

Sample Input

2
16
19
0



Sample Output

4
840
786432

Problem 8: Arithmetic Sequence

Problem

In mathematics, an arithmetic progression (AP) or arithmetic sequence is a sequence of numbers such that the difference between the consecutive terms is constant. For instance, the sequence 5, 7, 9, 11, 13, 15 ... is an arithmetic progression with common difference of 2.

You are to write a program to determine if a give sequence is an arithmetic sequence and if yes then list out the next 5 numbers in the sequence.

Input

The first line in the test data file contains the number of test cases (< 100). After that, each line contains one test case: the first number is the number of entries, n , in the sequence (provided as an integer), and the next n numbers are the sequence itself.

Output

For each test case, you are to either output "not an arithmetic sequence", or you are to output the next 5 numbers in the sequence separated by a single space.

Sample input

3 Num test cases
3 1 2 3
5 4 8 12 16 20
4 1 2 3 5

Sample Output

4 5 6 7 8
24 28 32 36 40
not an arithmetic sequence

```
int numTest;
cin >> numTest;
for (int i = 0; i < numTest; ++i)
{
    cin >> line;
    store in vector
    read first element
    int diff = third element - second element
    for (int j = 4; j < size; ++j)
    {
        if (vec[j] - vec[j-1] != diff)
        {
            raise flag
            break
        }
    }
    if (flag)
        cout << "not an arithmetic sequence" << endl;
    else
    {
        for (j = 0; j < 5; ++j)
            cout << last element + (j * diff) << " ";
    }
}
```


Problem 9: The Magic Saucer

Problem

A sick man is mysteriously healed of all his diseases when rays from a magical alien saucer hit him while he is lying on the ground. He has a shed to cover him but the rays hit him at various parts of the body at various times since the shed is not enough to cover him and the alien saucer is moving.

Given the position of the man's body and the shed, calculate which percentage of the body that is exposed to the alien saucer rays for at least part of the time. For simplicity, we will treat the man's body as rectangular, of size $h \times w$ centimeters. We treat the ground as a two-dimensional plane, with the origin at the center of the man's body. The shed is assumed to be a circle of radius r , mounted horizontally at distance d above the man's body, with the center at the origin as well. The man is lying for a period from α_1 to α_2 , where $1^\circ \leq \alpha_1 < \alpha_2 \leq 179^\circ$ are the angle that the alien machine makes with the ground at the beginning and end of your lying. (Assume that the alien machine is a point infinitely far away from the Earth, and travels straight from the left to the right.)

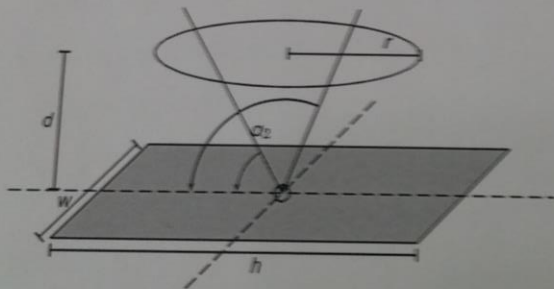
Input

The first line contains a number $K \geq 1$, which is the number of input data sets in the file. This is followed by K data sets of the following form:

Each data set consists of a single line, containing (in this order) the real numbers w , h , r , d , α_1 , α_2 .

Here is a figure illustrating these quantities. Notice that the machine travels "from the feet to the head", and

not "from the left arm to the right arm".



Problem 9 (Cont.)

Output

For each data set, first output "Data Set x:" on a line by itself, where x is its number. Then, output the percentage of the body that will be exposed to machine rays at some time, rounding to two decimals.

Sample Input

```
2
0.5 1.8 5.0 0.2 70.0 120.0
0.5 2.0 0.75 1.0 45.0 90.0
```

Sample Output

```
Data Set 1:
0.00%
```

```
Data Set 2:
76.41%
```

Problem 10: Airline Ticket Rule

Problem:

Airlines (both domestic and international) world over have instituted uniform set of rules to benefit frequent travelers (Hypothetically). The rule is described below:

Suppose a passenger has to fly from City A to City B. Both the cities can be located in any part of the world and the passenger can take any number of intermediate stops and also the flight chosen can be a multiple stopover flight. That is normal but what is unique is let say passenger X has four family members and each of them at different points of time (of course before the final trip) buy four tickets for different routes and the routes selected can be combined in any manner to go from city A to city B. Then according to the new rules the ticket can be transferred to the person originally travelling from city A to city B.

Example

Let's say In order to travel from City A to City B. These are the following options

1. A flight from City A to City D ($A \rightarrow D$), $D \rightarrow C$, $C \rightarrow E$, $E \rightarrow F$ and $F \rightarrow B$ lets name all these connections as 1,2,3,4 and 5 respectively.
2. Now let's say there is a connection from $D \rightarrow E$ also and we name it as name it as 6 And $E \rightarrow B$ as 7

So under this new ticketing scheme any member of the family can purchase a ticket/tickets for any leg /legs and then if all the legs can be combined that successfully allow for travel from A to B then anyone from the family can travel for that leg irrespective of the original person the tickets were bought so in our case if the family (or 1 person in the family) has any of the following tickets then $A \rightarrow B$ can be travelled.

Example legs that make this flight possible:

12345, 1645, 167, and so on.

Here, you will be given a list of all family members. In addition, you will be given, for each person, the details of tickets they bought. The question is whether there is any group of family members whose tickets together could be used to fly from say $A \rightarrow B$. To that end, you will be given details of tickets that together can be used for flying from one city to the other. If, say, tickets {1, 4, 6} together are enough to fly, then of course, if the group together buys {1, 2, 4, 6, 9}, that will also do. You are to decide how many such sets are in your data set that could be used to fly. (Notice that if one person alone

Problem 10 (cont.)

already has all the tickets, then each group containing him also has all the tickets, which means that you may count many such sets. That's correct.)

Input

The first line is the number K of input data sets, followed by the K data sets, each of the following form:

The first line contains four integers' n , m , k , and c , separated by spaces. $1 \leq n \leq 20$ is the number of people in your family. $1 \leq m \leq 50$ is the total number of connections between any 2 cities, and $0 \leq k \leq 100$ is the number of ticket combinations that can be used to fly from one city to the other. $0 \leq c \leq 400$ is the number of people who bought tickets.

This is followed by k lines, each describing a combination of tickets that can be used to fly. Each such line contains as its first number an integer s_i , $1 \leq s_i \leq m$, the number of items this combination contains. This is followed by s_i integers on the same line, each between 1 and m .

The next n lines, describes the items that the corresponding person bought. Each such line starts with a number t_j , $0 \leq t_j \leq m$, which is the number of tickets person j bought. Then follow t_j integers between 1 and m , each giving a ticket.

Finally, there are c lines, each containing two distinct integers between 1 and n , describing a connection between those two people.

Output

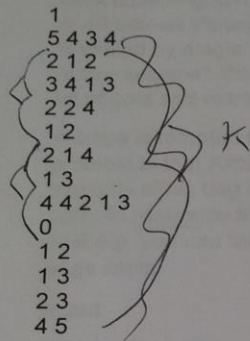
For each data set, output "Data Set x :" on a line by itself, where x is its number. Then, output the number of different group sets (of one or more person) which could be used for flying by pooling their tickets. This should be followed by an empty line.

$m = \# \text{ of connects between } 2 \text{ cities}$

Problem 10 (cont.)

Sample Input

1
5 4 3 4
2 1 2
3 4 1 3
2 2 4
1 2
2 1 4
1 3
4 4 2 1 3
0
1 2
1 3
2 3
4 5



Sample Output

5

Problem 11: Alibaba's Knapsack!

Problem

The tale begins in 18th century. Alibaba is a poor woodcutter. One day, Alibaba is at work collecting and cutting firewood in the forest, and he happens to overhear a group of 40 thieves visiting their treasure stash. The treasure is in a cave, the mouth of which is sealed by magic. It opens on the words "open sesame" and seals itself on the words "close sesame". When the thieves are gone, Alibaba enters the cave himself and sees lot of gold and extravagant things in the cave.

Alibaba is tempted to steal from the thieves and plans to take some from the cave. The problem is that Alibaba has a bag which can carry no more than 30 'weights' and the capacity of his bag is .45 'cubic lengths'. From the items in the cave how many can he carry to maximize the value he is carrying away with him? The number of items has no limit e.g. you can take as many of one item as you can as long as it does not exceed the bags limits.

Input

The first line will contain a single integer p which will define how many different items exist in the treasure stash. The next p lines will contain 4 values per line. These 4 values will go as follows Item_Name Value Weight Volume all separated by a single space.

Output

The output will contain one line which will go as follows "Maximum value achievable is:" followed by the max value without breaking the limits of the "bag".

Item	Value(each) 'mudra'	Weight	Volume
Malachite	3000	2.0	.002
Moonstone	1800	.2	.015
Amber	2500	.3	.020

Sample Input

```
3
Malachite 3000 2.0 0.002
Moonstone 1800 0.2 0.015
Amber 2500 0.3 0.020
```

```
struct bag {
    int weight;
    double volume;
};

int numLoops;
cin >> numLoops;
```

Problem 11 (Cont.)

Sample Output

Maximum value achievable is: 88000

(Comments the sample output is achieved as follows)

This value is achieved by the following:

This is achieved by carrying (one solution): 12 Malachite, 15 MoonStone, 10 Amber
The weight to carry is: 30 and the volume used is: 0.449

Problem 12: The Wedding Gift

Problem

Manny, Sid, and Diego are currently living in a large valley surrounded by an enormously high ice wall on all sides. Manny falls in love with Ellie and they get married. Sid and Diego plan to buy a gift for the wedding. They are short 26 cents for the gift. They go around the whole ice community hoping someone will lend them the money. They come across the sabre toothed squirrel Scrat and ask for help. Scrat asks them how they want it. Sid and Diego consider as to how many ways there are to make 26 cents using pennies, nickels, dimes and quarters. They realize that there are 13 ways to do this:

- 1) $26 = 25 + 1$
- 2) $26 = 10 + 10 + 5 + 1$
- 3) $26 = 10 + 10 + 1 + 1 + 1 + 1 + 1 + 1$
- 4) $26 = 10 + 5 + 5 + 5 + 1$
- 5) $26 = 10 + 5 + 5 + 1 + 1 + 1 + 1 + 1 + 1$
- 6) $26 = 10 + 5 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1$
- 7) $26 = 10 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1$
- 8) $26 = 5 + 5 + 5 + 5 + 5 + 1$
- 9) $26 = 5 + 5 + 5 + 5 + 1 + 1 + 1 + 1 + 1$
- 10) $26 = 5 + 5 + 5 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1$
- 11) $26 = 5 + 5 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1$
- 12) $26 = 5 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1$
- 13) $26 = 1 + 1$

Sid and Diego have to go to a Water Park to get the gift but they realize that the combination might not be right anymore since they use a different currency. However Sid and Diego don't know what currency is used in the Water Park.

Write a program to solve the following problem. Given a currency system ($c_1, c_2, c_3, \dots, c_k$) (that is the value of the coins: c_1 cents, c_2 cents, \dots, c_k cents) and a given number N , output the number of ways to obtain N cents using the currency system.

Input

The input will contain two lines. The first line will have a number n that states how many different problems to solve. The next n lines will contain the problems starting with p the number of different coin values that exist at the water park. The next p numbers will be the different coin sizes from smallest to largest. The last number will be the value needed by Sid and Diego.

Problem 12 (cont.)

Output

The output will contain one line per answer that will just output the number of combinations that can be reached with the given input.

Sample Input

1
4 1 5 10 25 26

read in int

Sample Output

13

store first one

read first int of next line

make array of coin values

store total

~~function~~ int i = size of array for coins

while (total added < total)

~~if~~

if (array[i] + total) <

Problem 13: Lockscreen

Our smartphones carry a lot of personal information. All of your text messages, emails, notes, apps, app data, music, pictures, and so much more are all on there. While it's a very great convenience to have all of these on your phone, it's also a major security risk if all of this data is easily accessible. The best way to prevent simple unauthorized access is by setting some sort of lock on your phone.

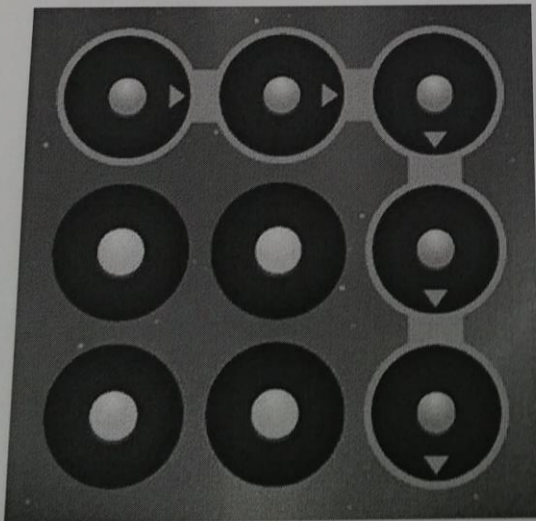
Problem

A lot of phones nowadays are moving away from the standard number lock to using pattern locks. Given a n -by- n grid of dots, a pattern is defined to be a directed path among the n^2 points of length at least 1. The below picture depicts one possible pattern for a 3 by 3 grid.

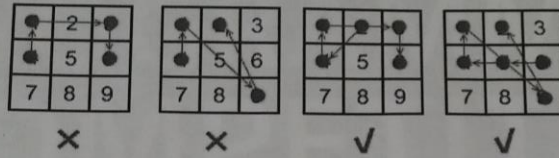
The rules for creating a pattern are as follows.

- We must use a minimum of p nodes or more to make a pattern.
- Once a node is visited, then the node can't be visited anymore.
- You can start at any node.
- A pattern has to be connected.
- Cycle is not allowed.

How many distinct patterns are possible?



Problem 13 (cont.)



Input

The first line will contain a number n which is the total number of inputs you will be given. The next n lines will have a single number, p , for each line which will define the number of nodes you must use for a pattern. (If the number is 4 you must use exactly nodes to make a pattern)

Output

For each line of input output the total number of different patterns that can be achieved for given input.

Sample Input

3
1
2

Sample Output

9
56