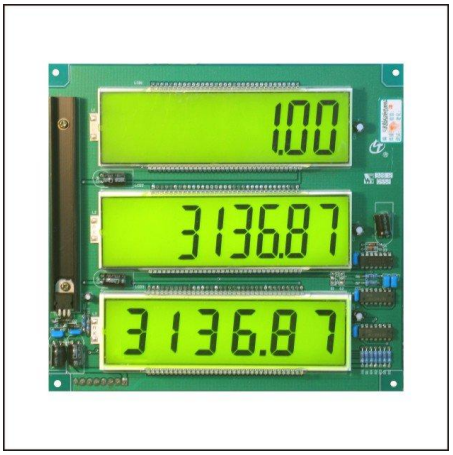
Real Programming final Day:

Note: It is not necessary that you real programmers solve each and every set. If you think one set is easier than other (2nd ofcourse easiest) than you can prefer to write code of that set. But please mention the Day Number when you submit the problems**. Also still if you have not submitted any code you can write code for all the sets.**

1	5
2	5
3	10
4	15
5	25
6	35



1. **Real Programmer's virtual LCD**: Being a real programmer and interest in doing something different with your computer you have decided to emulate a LCD display. Your task here is what to make a program that takes a integer number (say for example we have 1234 as an integer) and displays its LCD numbers as shown here.

2. **Real JUMPS:** A sequence of *n > 0* integers is called *real jumper* if the absolute values of the difference between successive elements take on all the values 1 through *n-1*. For instance,

1 4 2 3
is a real jumper, because the absolutes differences are 3, 2, and 1 respectively. The definition implies that any sequence of a single integer is a real jumper. You are to write a program to determine whether or not each of a number of sequences is a real jumper.

Input

Each line of input contains an integer *n* <= 3000 followed by *n* integers representing the sequence.

Output

For each line of input, generate a line of output saying "Real" or "Not Real".

Sample Input

4 1 4 2 3
5 1 4 2 -1 6

Sample Output

Real
Not Real

3. **Real Programming for Gabbar** : The world-known gangster Gabbar is moving to Ramgarh. He has a very big plan to attack on the rich men of Ramgarh. Since he will visit all of them very often, he is trying to find a house close to them.

   Gabbar wants to minimize the total distance to all of them and has blackmailed you to write a program that solves his problem.

   Input

   The input consists of several test cases. The first line contains the number of test cases.

For each test case you will be given the integer number of relatives r (0 < r < 500) and the street numbers (also integers) where they live (0 < si < 30000). Note that several relatives could live in the same street number.

Output

For each test case your program must write the minimal sum of distances from the optimal Vito's house to each one of his relatives. The distance between two street numbers si and sj is dij= |si-sj|.

Sample Input

2

2 2 4

3 2 4 6

Sample Output

2

4

4. **Real Simulation**: Suppose there are three parties h1, h2, h3 where hi represents hartal (strike) parameter that denotes average number of days between two successive hartals (strikes). You have to start the simulation

from Sunday and we assume that there will be no hartals on Friday and Saturday.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Days														
	Su	Mo	Tu	We	Th	Fr	Sa	Su	Mo	Tu	We	Th	Fr	Sa
Party 1			x			x			x			x		
Party 2				x				x				x		
Party 3								x						
Hartals			1	2				3	4			5		

The simulation above shows that there will be exactly 5 hartals (on days 3, 4, 8, 9 and 12) in 14 days. There will be no hartal on day 6 since it is a Friday. Hence we lose 5 working days in 2 weeks.

In this problem, given the hartal parameters for several political parties and the value of N, your job is to determine the number of working days we lose in those N days.

The first line of the input consists of a single integer T giving the number of test cases to follow.

The first line of each test case contains an integer N ( ) giving the number of days over which the simulation must be run. The next line contains another integer P ( ) representing the number of political parties in this case. The i-th of the next P lines contains a positive integer hi (which will never be a multiple of 7) giving the hartal parameter for party i ( ).

For each test case in the input output the number of working days we lose. Each output must be on a separate line.

Sample Input

2
14
3
3
4
8
100
4
12
15
25
40
Sample Output

5
15

5. **Real Decoding:** Lets decode the string in a real way. You are given a String code and int[]s position and length. code contains an encoded string which you must decode using the following method. Step through the elements of position in order, and for each element i, take the substring of length length[i] at position position[i]. Insert the reverse of that substring before position position[i]+length[i], thereby creating a palindromic substring. All positions are 0-based. Return the decoded String.

Examples

0)

"ab"

{0}

{2}

Returns: "abba"

The decoding step selects the whole string and appends it in reversed form.

1)

"Misip"

{2,3,1,7}

{1,1,2,2}

Returns: "Mississippi"

The decoding steps are: "Misip" -> "Missip" -> "Misssip" -> "Mississip" -> "Mississippi"

2)

"XY"

{0, 0, 0, 0}

{2, 4, 8, 16}

Returns: "XYYXXYYXXYYXXYYXXYYXXYYXXYYXXYYX"

In this example the length of the string doubles in each decoding step.

3)

"TC206"

{1,2,5}

{1,1,1}

Returns: "TCCC2006"

4)

"nodecoding"

{}

{}

Returns: "nodecoding"

Problem Statement for DigitPrime

6. Real Search:

A number is called 2-digit-prime if using each of its digits at most once, we can make a prime number containing exactly 2 digits (with no leading zeros). For example, 153 is 2-digit-prime because we can use its digits to make 13, which is a prime number with 2 digits (note that we can also make 53 and 31). Given ints a and b, return the number of 2-digit-prime numbers between a and b, inclusive. See examples for further clarification.

Examples

0)

11

20

Returns: 6

2-digit-prime numbers are:

11 (note that we can use some digit twice if it appears twice in the number), 13, 14 (using its digits we can make 41), 16 (we can make 61), 17 and 19.

1)

37

98

Returns: 21

2)

9003

9003

Returns: 0

Note that we are looking for 2 digit prime numbers with no leading zeros, so 03 is not considered a 2 digit prime number.

3)

11

11111

Returns: 8777

4)

97463

100000

Returns: 2436

5)

33561

33601

Returns: 40

The only number in this interval that is not 2-digit-prime is 33600.

6)

11000

11999

Returns: 1000

Each number in this interval is 2-digit-prime.