Computer University (Magway)

Programming Contest

Problem Set

Please check you have 6 problems.

Contest Problem Set

Problem A	Finding Number in Sequence	1 page
Problem B	IMEI	1 page
Problem C	Prime Digital Roots	2 pages
Problem D	Number Pattern	1 page
Problem E	Message Encryption	1 page
Problem F	Octal Code	2 pages

Note. The input and output for all problems are standard input and output.

January 3, 2017

Problem A: Finding Number in Sequence

Kyaw Kyaw and Maw Maw, two friends play a test which finds a number in sequence (fibonanci sequence). The Fibonacci Sequence is the series of numbers: 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144 …, and so on. The first two numbers in the series is '0' and '1' and every next number is found by adding up the two numbers before it. The 2 is found by adding the two numbers before it (1+1). Similarly, the 3 is found by adding the two numbers before it (1+2), and the 5 is (2+3), and so on! Suppose, the number is "21" which is in fibonanci sequence and the number is "35" which is not in this sequence.

Input

The first line of the input has a single integer number, P, (1<= P <=1000), which is the number of data sets. The integer in each line indicates the input integer in the range 0 to 75025 inclusive.

Output

The output must be **1** for that number which is in fibonanci sequence. If not, the output must be **0**.

Sample Input	Sample Output
4	1
55	0
289	1
377	1
46368	

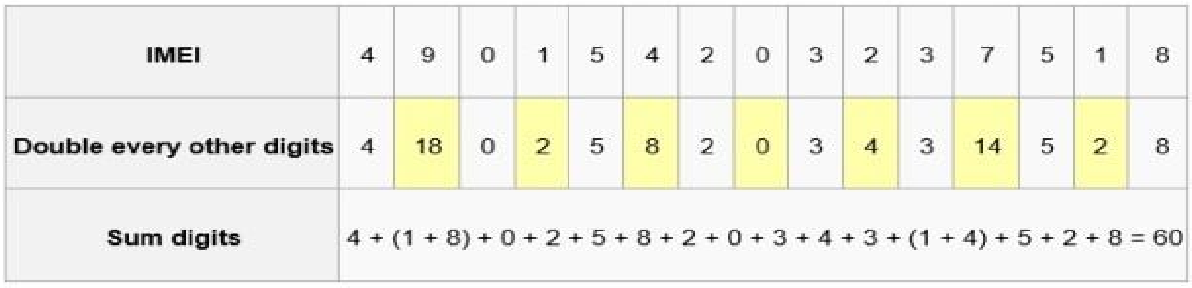
Problem B: IMEI

The International Mobile Station Equipment Identity or IMEI is a number, usually unique, to identify mobile phones, as well as some satellite phones. It is usually found printed inside the battery compartment of the phone. Every mobile phone, GSM modem or device with a built-in phone / mode has a unique 15 digital IMEI number. The IMEI number is used by a GSM network to identify valid devices and therefore can be used for stopping a stolen phone from accessing that network. The IMEI (15 decimal digits: 14 digits plus a check digit) includes information that consist the first eight digits, known as the Type Allocator Code (TAC) and the next six digits are the serial number specified by the equipment manufacturer. The last digit is the check digit.

The IMEI is validated in three steps:

1. Starting from the right (excluding the check digit), double the digit, then skip next digit, double the digit and so on (e.g., 7 becomes 14).
2. Sum the digits (e.g., 14 becomes 5)
3. Check if the sum is divisible by 10.

For example, the input IMEI number is 490154203237518.



IMEI	4	9	0	1	5	4	2	0	3	2	3	7	5	1	8
Double every other digits	4	18	0	2	5	8	2	0	3	4	3	14	5	2	8
Sum digits	4 + (1 + 8) + 0 + 2 + 5 + 8 + 2 + 0 + 3 + 4 + 3 + (1 + 4) + 5 + 2 + 8 = 60														

Since, 60 is divisible by 10, hence the given IMEI number is Valid.

Input

The first line of the input has a single positive integer number, T, $1 \leq T \leq 100$, which is the number of data sets. Each of data sets consists of a single line of a fifteen digit IMEI number.

Output

For each data set, produce one line of output. If the IMEI number input is valid, print "**Valid**", otherwise print "**Invalid**".

Sample Input	**Sample Output**
3	Valid
490154203237518	Valid
353538057997225	Invalid
353051013170287	

Problem C: Prime Digital Roots

The *digital root* a number is found by adding together the digits that make up the number. If the resulting number has more than one digit, the process is repeated until a single digit remains.

Your task in this problem is to calculate a variation on the digital root – a *prime digital root.* The addition process described above stops when there is only one digit left, but will also stop if the original number or any of the intermediate numbers (formed by addition) are prime numbers. If the process continues and results in a single digit that is not a prime number, then the original number has no prime digital root.

An integer greater than one is called a prime number if its only positive divisors (factors) are one and itself.

- For example, the first six primes are 2, 3, 5, 7, 11, and 13.
- Number 6 has four positive divisors: 6, 3, 2 and 1. Thus number 6 is not a prime.
- Caveat: number 1 is not a prime.

Examples of Prime Digital Roots

1	This is not a prime number, so 1 has no prime digital root.
3	This is a prime number, so the prime digital root of 3 is 3.
4	This is not a prime number, so 4 has no prime digital root.
642	This is not a prime number, so adding its digits gives 6 + 4 + 2 = 12. This is not a prime number, so adding again gives 1 +2 = 3. This is a prime number, so the prime digital root of 642 is 3.
128	This is not a prime number, so adding its digits gives 1 + 2 + 8 = 11. This is a prime number, so the prime digital root of 128 is 11.
886	This is not a prime number, so adding its digits gives 8 + 8 + 62 = 22. This is not a prime number, so adding again gives 2 +2 = 4. This is not a prime number, so 886 has no prime digital root.

Input

The input will contain a single integer on each line in the range 0 to 999999 inclusive. The end of the input will be indicated by the value 0.

Output

If the input number has a prime digital root, then the input number which must be followed by a single space, and then the calculated prime digital root.

If the input number has no prime digital root, then the input number which must be followed by a single space, and then print "none" as messages string for no prime digital root.

Sample Input	Sample Output
1	1 none
3	3 3
4	4 none
11	11 11
642	642 3
128	128 11
0	

Problem D: Number Pattern

For this problem, you will write a program that takes an integer number *n* (number of pattern) which is only odd number (not even number) and creates new patterns for that integer *n*. The first pattern must start with 1, each pattern consists of *n+1* digits and the last digit is greater than 1 of the previous digits. The second pattern starts with a number which is greater than 1 of the last digit of the first pattern. Every odd row pattern, all digits are equal to the first digit except the last digit. Every even row pattern, all digits are equal to a number which is less than 1 of the start digits in this pattern.

Input :

The input contains a single odd integer N, ($1 \leq N < 10$) which is the number of patterns. The end of the input will be indicated by the value 0.

Output:

Output will be the corresponding input number.

Sample Input	**Sample Output**
3	1112 3222 3334
5	111112 322222 333334 544444 555556
0	

Problem E: Message Encryption

John wants to send messages to Smith. John has decided to encrypt his message so that nobody else could understand them, except Smith. The code is based on shifting approach where each letter is shifted to one right through the alphabet. In this context, the alphabet is treated as being circular so that the first letter follows after the last letter, and the last letter precedes the first letter. The encryption message should be the characters between 'A' to 'Z', 'a' to 'z', punctuation, and any other symbols are not affected in this scheme. For example, with a shift of 1, 'A' becomes 'B', 'Z' becomes 'A', 'a' becomes 'b', 'z' becomes 'a'. Then, John wants to more secure his message, so reverse the message. Suppose that the message is "cumgy" to shift right to 1 shift, so the first message is "dvnhz" and then reverse the message, finally the encrypted message is as "zhnvd".

Input

The first line of the input has a single positive integer number, T, 1<= T <=100, which is the number of data sets. Each line consists of a string of 1 to 200 arbitrary characters and represents a fragment of the text to be encrypted.

Output

The output should display the encrypted message for each input.

Sample Input	Sample Output
4 computer PROGRAMMING zoology ACM-ICPC	sfuvqnpd HOJNNBSHPSQ zhpmppa DQDJ-NDB

Problem F: Octal Code

Mg Mg and Aung Aung are friends. Mg Mg lives in Magway and Aung Aung lives in Yangon. They connect with each other with octal code patterns. They could detect as an octal digit or octal code including 0-7. Then they could transform a three digits of octal code into *a character* since they knows how to look up the ASCII table. Suppose, if Mg Mg receives a code sequence “110151040115147040115147”, he decode it into the message “Hi Mg Mg”.

Input

The first line of the input has a single positive integer number, P, $1 \leq P \leq 100$, which is the number of data sets. Each line contains a *number* of octal digits that you have to transform into a text message. Three digits of octal code correspond to a single character. There are less than 255 octal digits in each line.

Output

For each line of octal codes, print out the corresponding text message.

Sample Input	Sample Output
2 110151040115147040115147 110157167040141162145040171157165077	Hi Mg Mg How are you?

Char		Hex	Oct	Dec
NUL	^@	00	000	0
SOH	^A	01	001	1
STX	^B	02	002	2
ETX	^C	03	003	3
EOT	^D	04	004	4
ENQ	^E	05	005	5
ACK	^F	06	006	6
BEL	^G	07	007	7
BS	^H	08	010	8
TAB	^I	09	011	9
LF	^J	0A	012	10
VT	^K	0B	013	11
FF	^L	0C	014	12
CR	^M	0D	015	13
SO	^N	0E	016	14
SI	^O	0F	017	15
DLE	^P	10	020	16
DC1	^Q	11	021	17
DC2	^R	12	022	18
DC3	^S	13	023	19
DC4	^T	14	024	20
NAK	^U	15	025	21
SYN	^V	16	026	22
ETB	^W	17	027	23
CAN	^X	18	030	24
EM	^Y	19	031	25
SUB	^Z	1A	032	26
ESC	^[	1B	033	27
FS left		1C	034	28
GS right		1D	035	29
RS up		1E	036	30
US down		1F	037	31

Char	Hex	Oct	Dec
Space	20	040	32
!	21	041	33
"	22	042	34
#	23	043	35
$	24	044	36
%	25	045	37
&	26	046	38
' apostr.	27	047	39
(	28	050	40
)	29	051	41
*	2A	052	42
+	2B	053	43
, comma	2C	054	44
- dash	2D	055	45
. period	2E	056	46
/	2F	057	47
0	30	060	48
1	31	061	49
2	32	062	50
3	33	063	51
4	34	064	52
5	35	065	53
6	36	066	54
7	37	067	55
8	38	070	56
9	39	071	57
:	3A	072	58
;	3B	073	59
<	3C	074	60
=	3D	075	61
>	3E	076	62
?	3F	077	63

Char	Hex	Oct	Dec
@	40	100	64
A	41	101	65
B	42	102	66
C	43	103	67
D	44	104	68
E	45	105	69
F	46	106	70
G	47	107	71
H	48	110	72
I	49	111	73
J	4A	112	74
K	4B	113	75
L	4C	114	76
M	4D	115	77
N	4E	116	78
O	4F	117	79
P	50	120	80
Q	51	121	81
R	52	122	82
S	53	123	83
T	54	124	84
U	55	125	85
V	56	126	86
W	57	127	87
X	58	130	88
Y	59	131	89
Z	5A	132	90
[	5B	133	91
\	5C	134	92
]	5D	135	93
^ caret	5E	136	94
_ underline	5F	137	95

Char	Hex	Oct	Dec
`	60	140	96
a	61	141	97
b	62	142	98
c	63	143	99
d	64	144	100
e	65	145	101
f	66	146	102
g	67	147	103
h	68	150	104
i	69	151	105
j	6A	152	106
k	6B	153	107
l	6C	154	108
m	6D	155	109
n	6E	156	110
o	6F	157	111
p	70	160	112
q	71	161	113
r	72	162	114
s	73	163	115
t	74	164	116
u	75	165	117
v	76	166	118
w	77	167	119
x	78	170	120
y	79	171	121
z	7A	172	122
{	7B	173	123
| pipe	7C	174	124
}	7D	175	125
~ tilde	7E	176	126
Delete	7F	177	127