ASPC-A 2023 Day 5 AM Problem Set

This is the **problem set for Day 5 (AM)**.

It includes topics covered during the week in preparation for the final contest session.

This problem set has **6 problems**. The items in the problem list below are links to each problem. You can click a problem name to jump to its page in the PDF file.

Problem A: F91

McCarthy is a famous theorician of computer science. In his work, he defined a recursive function, called f91, that takes as input a positive integer N and returns a positive integer defined as follows:

- If $N \leq 100$, then $f91(N) = f91(f91(N + 11))$.
- If $N \geq 101$, then $f91(N) = N - 10$.

Write a program, that computes McCarthy's f91 function.

Input
Input consists of a line containing an integer N $(1 \leq N \leq 500)$.

Output
Print $f91(N)$.

Sample Input #1
```
420
```

Sample Output #1
```
410
```

Sample Input #2
```
12
```

Sample Output #2
```
91
```

Problem B: Do Androids Dream?

Tali'Zorah is an alien from a far away star system who has hundreds of androids managing her large estate. Sometimes, in the middle of the day, an android will drop what it's doing to ask her if it has a soul and is capable of having dreams and aspirations. Tali'Zorah knows that certain units do have souls, and it all depends on their unit number. She manually computes the answer using a specialized formula from the droids' manufacturer. However, more androids have been bothering her recently, and now everyone's becoming more occupied with this whole "soul" nonsense.

Your job is to write a program for the ever-busy Tali'Zorah that determines whether a given android unit has a soul. The rules are as follows:

1. If the result of the formula is 7 or 9, then the unit has a soul.
2. If the result of the formula is 5, then the unit does NOT have a soul.
3. If the input is less than or equal to 0, then the unit does NOT have a soul.
4. If the input is divisible by 3 or 8, then the unit number will be divided by that number. The formula will be recalculated using the quotient.
5. If none of the conditions above are fulfilled, the input is subtracted by 20. The formula will be recalculated using the difference.

Input
Input consists of a line containing an integer N $(1 \leq N \leq 1000)$, the unit number of an android.

Output
Print "YES" if the unit has a soul. Otherwise, print "NO".

Sample Input #1	Sample Output #1
600	NO
Sample Input #2	**Sample Output #2**
876	NO
Sample Input #3	**Sample Output #3**
7	YES
Sample Input #4	**Sample Output #4**
21	YES
Sample Input #5	**Sample Output #5**
576	NO

ASPC-A 2023 Day 5 AM Problem Set

aspc
ateneo programming summer camp

Problem C: Chocolate Box

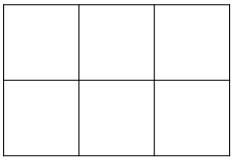
Chino likes chocolates! He has a rectangular box that is 2 inches by n inches.

He has chocolates which are rectangles which are 2 by 1 inches.

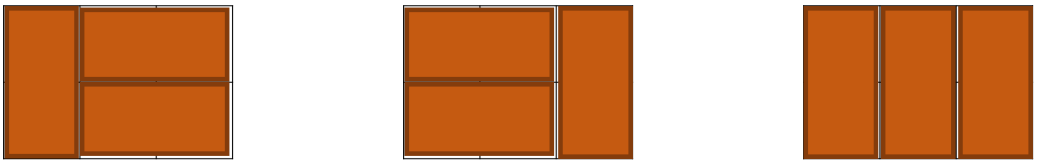
He wants to fill up his box with chocolates. He places chocolates inside the box so that

- The edges of the chocolates are parallel with the sides of the box.
- The chocolates don't overlap.

For example, if $n = 3$, he has a box which looks like:



There are 3 ways he can fill up a 2×3 inch box with chocolates.



If $n = 2$, there are two ways to fill up the box.

If Chino has a $2\times n$ box, how many ways can Chino fill it up with chocolates?

Input

Input consists of a line containing an integer n $(1\leq n\leq 25)$.

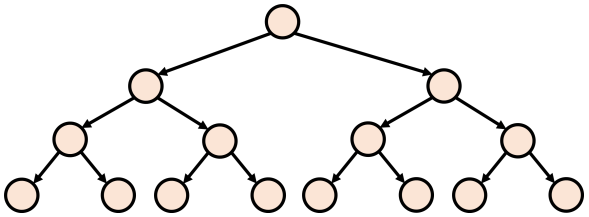
Output

Output the number of ways Chino can fill up the box.

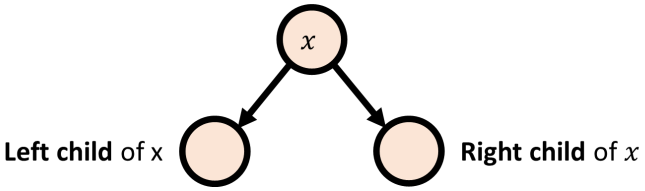
Sample Input #1	Sample Output #1
3	3
Sample Input #2	**Sample Output #2**
5	8

Problem D: Binary Tree

In computer science, a **perfect binary tree** is a diagram which looks like this:

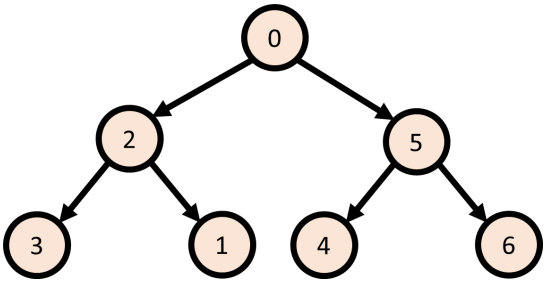


We call the circles **nodes**. The top-most node is called the **root**, and the bottom nodes are called **leaves**. In a **perfect binary tree**, nodes which are not leaves have **exactly two children**: they have a left child and a right child. Leaves do not have children.



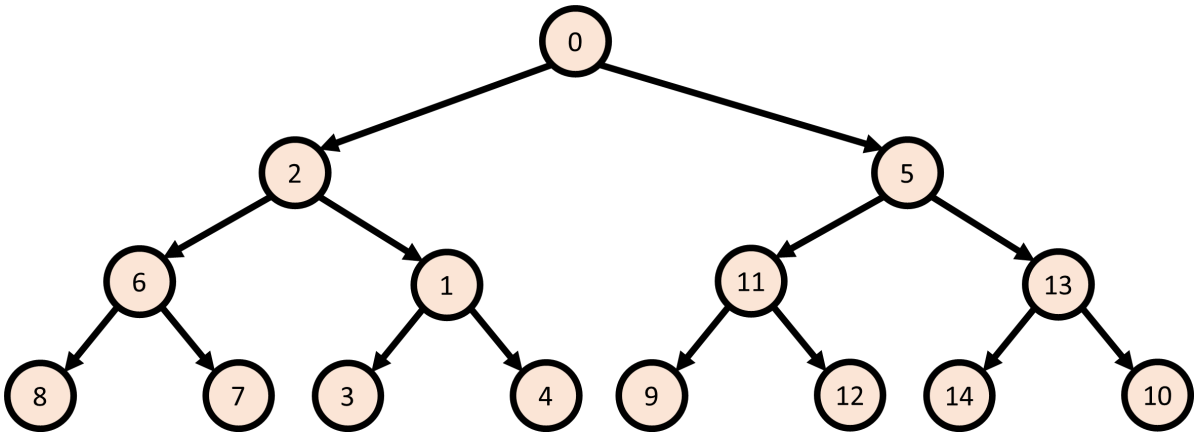
One way to write this is `x[L,R]` where L is the left child of x and R is the right child of x.

We can write bigger trees by nesting this notation. Here is an example of a perfect binary tree, where we number the n nodes with an integer from 0 to $n - 1$.



We can write this tree as `0[2[3,1],5[4,6]]`

Here is another example of a perfect binary tree.



We can write this tree as `0[2[6[8,7],1[3,4]],5[11[9,12],13[14,10]]]`

In this problem, we will give you a perfect binary tree as input.

You will have to output the same perfect binary tree in a more convenient form.

Input

The first line of input contains an integer n $(3 \leq n \leq 64)$, the number of nodes in the perfect binary tree. The nodes in the perfect binary tree are numbered from 0 to $n - 1$.

The second line of input contains a list of n integers $L_0, L_1, L_2, \ldots, L_{n-1}$, separated by spaces.

L_x will denote the node number of the **left child** of x. If x is a leaf, then $L_x = -1$.

The second line of input contains a list of n integers $R_0, R_1, R_2, \ldots, R_{n-1}$, separated by spaces.

R_x will denote the node number of the **right child** of x. If x is a leaf, then $R_x = -1$.

All inputs will be valid perfect *binary trees.*

Node 0 will be the root of the perfect *binary tree.*

Output

Output the binary tree in the notation described in the problem statement.

Sample Input #1
```
7
2 -1 3 -1 -1 4 -1
5 -1 1 -1 -1 6 -1
```

Sample Output #1
```
0[2[3,1],5[4,6]]
```

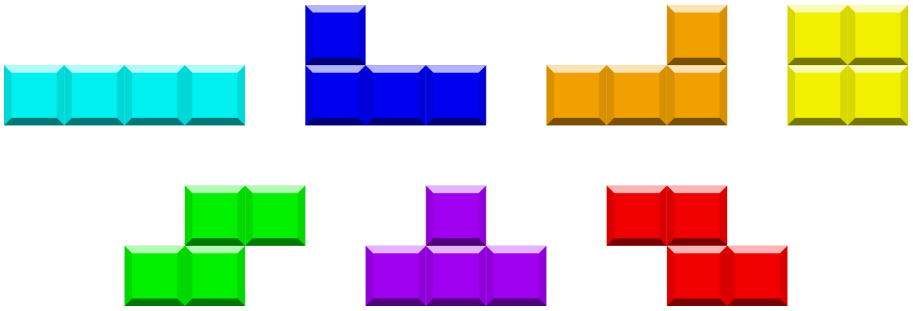
Sample Input #2
```
3
2 -1 -1
1 -1 -1
```

Sample Output #2
```
0[2,1]
```

Problem E: Sequence Verifier

Kevin needs more help for his Tetris game.

In Tetris, player get one piece at a time. There are 7 different types of pieces.



https://upload.wikimedia.org/wikipedia/commons/thumb/3/39/Tetrominoes_IJLO_STZ_Worlds.svg/2000px-Tetrominoes_IJLO_STZ_Worlds.svg.png

Players get pieces at random.

He wants his game mechanics to reflect those of official Tetris games, so when his generates a sequence of pieces to give to the player, he does the following:

- Generate a random permutation of {0,1,2,3,4,5,6}
- Give the player the pieces in that order.
- If the game runs out of pieces, generate another random permutation of {0,1,2,3,4,5,6}

He wants to check if his random permutation generator works properly.

Your task: Given a set of 7 numbers, check if the set only contains the numbers 0,1,2,3,4,5 and 6. Each number should occur in the set only once.

Input

The input consists of 7 integers, separated by spaces.

Output

Output `"Sequence Valid"` (without the quotes) if the input satisfies the conditions in the problem statement. Otherwise, output `"Please debug"` (without the quotes.)

Sample Input #1
```
6 5 4 2 1 3 0
```

Sample Output #1
```
Sequence Valid
```

Sample Input #2
```
0 3 2 1 1 4 5
```

Sample Output #2
```
Please debug
```

Problem F: Cipher Substitution

Substitution ciphers are the simplest of ciphers where each letter is replaced with another letter. In one form or another, they've been in use for over 2000 years.

Input
The first two lines of input determine how the letters are supposed to be substituted.
The first line of input consists of the **plaintext alphabet** in the first line (it will be exactly 52 characters long and will only contain upper-case and lower-case letters, no duplicates)
The second line of input contains the substitution alphabet (it will also be exactly 52 characters long and will only contain upper-case and lower-case letters, no duplicates)

These two lines will be followed by 8 lines of arbitrary lengths.

Output
Program output should be the substitution alphabet in the first line and the plaintext alphabet in the second line. The next 8 lines should also be the last 8 lines from the input, but after the substitution cipher has been applied.
To apply a substitution cipher, simply replace all instances of the characters in the plaintext alphabet with the corresponding characters in the substitution alphabet. Characters not in the plaintext alphabet are ignored.
For example, if the first character in the **plaintext alphabet** is a, and the first character in the **substitution alphabet** is Z, then you should replace every lowercase a with an uppercase Z.

Sample Input #1:
```
abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ
ZYXWVUTSRQPONMLKJIHGFEDCBAzyxwvutsrqponmlkjihgfedcba
Welcome to ASPC!
This is a difficult problem.
If you can solve this, you're awesome!
(And you'll probably be in the top eight, too.)
Remember, our helpful helpers
are always ready to help you
if you need help!
Enjoy week 2! ^_^
```

Sample Output #1:
```
ZYXWVUTSRQPONMLKJIHGFEDCBAzyxwvutsrqponmlkjihgfedcba
abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ
dVOXLNV GL zhkx!
gSRH RH Z WRUURXFOG KILYOVN.
rU BLF XZM HLOEV GSRH, BLF'IV ZDVHLNV!
(zMW BLF'OO KILYZYOB YV RM GSV GLK VRTSG, GLL.)
iVNVNYVI, LFI SVOKUFO SVOKVIH
ZIV ZODZBH IVZWB GL SVOK BLF
```

RU BLF MVVW SVOK!
vMQLB DVVP 2! ^_^

Sample Input #2:

```
ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz
abcdefghijklmABCDEFGHIJKLMnopqrstuvwxyzNOPQRSTUVWXYZ
As I said before,
this is a difficult problem.
That's why I provided a second sample.
Ah, so much text.
Non-letter characters aren't affected.
Did you notice that?
1 is one, 2 is two, 3 is three.
[]()[]()[]()[]()[]()[]()[]()[]()[]()[]()[]
```

Sample Output #2:

```
abcdefghijklmABCDEFGHIJKLMnopqrstuvwxyzNOPQRSTUVWXYZ
ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz
aS i Snvq orsORr,
TuvS vS n qvssvpUyT PROoyrz.
GunT'S WuY i PROVvqrq n SrpONq SnzPyr.
au, SO zUpu TrXT.
AON-yrTTrR punRnpTrRS nRrN'T nssrpTrq.
dvq YOU NOTvpr TunT?
1 vS ONr, 2 vS TWO, 3 vS TuRrr.
[]()[]()[]()[]()[]()[]()[]()[]()[]()[]()[]
```