1 Numbers from 1 to N

Description

Write a program that enters from the console a positive integer n and prints all the numbers from 1 to N inclusive, on a single line, separated by a whitespace.

Input

- The input will consist of a single line - the number N

Output

- The output should consist of a single line - the numbers from 1 to N, separated by a whitespace

Constraints

- N will be a valid positive 32-bit integers
- Time limit: 0.1s
- Memory limit: 16MB

Sample tests

Input	Output
5	1 2 3 4 5
1	1

2 Not Divisible Numbers

Description

Write a program that reads from the console a positive integer N and prints all the numbers from 1 to N not divisible by 3 or 7, on a single line, separated by a space.

Input

- Will always consists of one valid integer number - the number N.

Output

- Should always consists of the numbers from 1 to N, which are not divisible by 3 or 7, separated by a whitespace.

Constraints

- 1 < N < 1500
- Time limit: 0.1s
- Memory limit: 16MB

Sample tests

Input	Output
10	1 2 4 5 8 10
3	1 2

3 MMSA (Min, Max, Sum, Average) of N Numbers

Description

Write a program that reads from the console a sequence of N integer numbers and returns the *minimal*, the *maximal* number, the *sum* and the *average* of all numbers (displayed with 2 digits after the decimal point).

- The input starts by the number N (alone in a line) followed by N lines, each holding an integer number.
- The output is like in the examples below.

Input

- On the first line, you will receive the number N.
- On each of the next N lines, you will receive a single floating-point number.

Output

- You output must always consist of *exactly* 4 lines - the minimal element on the first line, the maximal on the second, the sum on the third and the average on the fourth, in the following format:

```
min=3.00
max=6.00
sum=9.00
avg=4.50
```

Constraints

- 1 <= N <= 1000
- All numbers will be valid floating-point numbers that will be in the range `[-10000, 10000]`
- Time limit: 0.1s
- Memory limit: 16MB

Sample tests

Input	Output

3
2
5
1
	min=1.00
	max=5.00
	sum=8.00
	avg=2.67

3
2
-1
4
	min=-1.00
	max=4.00
	sum=5.00
	avg=1.67

4 Print a Deck

Description

Write a program that reads a card sign(as a char) from the console and generates and prints all possible cards from a standard deck of 52 cards up to the card with the given sign(without the jokers). The cards should be printed using the classical notation (like 5 of spades, A of hearts, 9 of clubs; and K of diamonds).

- The card faces should start from 2 to A.
- Print each card face in its four possible suits: clubs, diamonds, hearts and spades.

Input

- On the only line, you will receive the sign of the card to which, including, you should print the cards in the deck.

Output

The output should follow the format bellow(assume our input is `5`):

```
2 of spades, 2 of clubs, 2 of hearts, 2 of diamonds
3 of spades, 3 of clubs, 3 of hearts, 3 of diamonds
...
5 of spades, 5 of clubs, 5 of hearts, 5 of diamonds
```

-

Constraints

- The input character will always be a valid card sign.
- Time limit: 0.1s
- Memory limit: 16MB

Sample tests

Input	Output
5	2 of spades, 2 of clubs, 2 of hearts, 2 of diamonds 3 of spades, 3 of clubs, 3 of hearts, 3 of diamonds ... 5 of spades, 5 of clubs, 5 of hearts, 5 of diamonds

5 Calculate!

Description

Write a program that, for a given two numbers N and x, calculates the sum S = 1 + 1!/x + 2!/x^2 + … + N!/x^N.

- Use only one loop. Print the result with 5 digits after the decimal point.

Input

- On the first line you will receive one number - N.
- On the second line you will receive another number - x.

Output

- Output only one number - the sum of the sequence for the given N and x.

Constraints

- N will always be a valid integer between 2 and 10, inclusive.
- X will always be a valid floating-point number between 0.5 and 100
- Time limit: 0.1s
- Memory limit: 16MB

Sample tests

Input	Output
3 2	2.75000
4 3	2.07407
5 -4	0.75781

6 Calculate Again

Description

Write a program that calculates N! / K! for given N and K.

- *Use only one loop*.

Input

- On the first line, there will be only one number - N
- On the second line, there will be only one number - K

Output

- Output a single line, consisting of the result from the calculation described above.

Constraints

- 1 < K < N < 100
    - *Hint: overflow is possible*
- N and K will always be valid integer numbers
- Time limit: 0.1s
- Memory limit: 16MB

Sample tests

Input	Output
5 2	60
6 5	6
8 3	6720

7 Calculate 3!

Description

In combinatorics, the number of ways to choose N different members out of a group of N different elements (also known as the number of combinations) is calculated by the following formula: $\binom{n}{k} = \frac{n!}{k!(n-k)!},$ For example, there are 2598960 ways to withdraw 5 cards out of a standard deck of 52 cards. Your task is to write a program that calculates `N! / (K! * (N - K)!)` for given N and K.

- *Try to use only two loops.*

Input

- On the first line, there will be only one number - N
- On the second line, there will also be only one number - K

Output

- On the only output line, write the result of the calculation for the provided N and K

Constraints

- 1 < K < N < 100
    - *Hint: overflow is possible*
- Time limit: 0.1s
- Memory limit: 16MB

Sample tests

Input	Output
3 2	3
4 2	6
10 6	210

52 5	2598960

8 Catalan Numbers

Description

In combinatorics, the Catalan numbers are calculated by the following formula:

$$C_n = \frac{1}{n+1}\binom{2n}{n} = \frac{(2n)!}{(n+1)!\,n!} = \prod_{k=2}^{n} \frac{n+k}{k} \qquad \text{for } n \geq 0.$$

- Write a program to calculate the N th Catalan number by given N

Input

- On the only line, you will receive the number N

Output

- Output a single number - the N th Catalan number

Constraints

- N will always be a valid integer number in the range `[0, 100]`
    - *Hint: overflow is possible.*
- Time limit: 0.1s
- Memory limit: 16MB

Sample tests

Input	Output

0	1
5	42
10	16796
15	9694845

9 Matrix of Numbers

Description

Write a program that reads from the console a positive integer number N and prints a matrix like in the examples below. Use two nested loops.

- *Challenge: achieve the same effect without nested loops*

Input

- The input will always consist of a single line, which contains the number N

Output

- See the examples.

Constraints

- 1 <= N <= 20
- N will always be a valid integer number
- Time limit: 0.1s
- Memory limit: 16MB

Sample tests

Input	Output
2	1 2 2 3

3	1 2 3
	2 3 4
	3 4 5

4	1 2 3 4
	2 3 4 5
	3 4 5 6
	4 5 6 7

10 Odd and Even Product

Description

You are given N integers (given in a single line, separated by a space).

- Write a program that checks whether the product of the odd elements is equal to the product of the even elements.
- Elements are counted from 1 to N, so the first element is odd, the second is even, etc.

Input

- On the first line you will receive the number N
- On the second line you will receive N numbers separated by a whitespace

Output

- If the two products are equal, output a string in the format "yes PRODUCT_VALUE", otherwise write on the console "no ODD_PRODUCT_VALUE EVEN_PRODUCT_VALUE"

Constraints

- N will always be a valid integer number in the range `[4, 50]`
- All input numbers from the second line will also be valid integers
- Time limit: 0.1s
- Memory limit: 16MB

Sample tests

Input	Output

5
2 1 1 6
3
	yes 6

5
4 3 2 5
2
	no 16 15

11 Binary to Decimal

Description

Using loops write a program that converts a binary integer number to its decimal form.

- The input is entered as string. The output should be a variable of type long.
- *Do not use the built-in .NET functionality.*

Input

- You will receive exactly one line containing an integer number representation in binary

Output

- On the only output line write the decimal representation of the number

Constraints

- All input numbers will be valid 32-bit integers
- Time limit: 0.1s
- Memory limit: 16MB

Sample tests

Input	Output
0	0
11	3
1010101010101011	43691
1110000110000101100101000000	236476736

12 Decimal to Binary

Description

Using loops write a program that converts an integer number to its binary representation.

- The input is entered as long. The output should be a variable of type string.
- *Do not use the built-in .NET functionality.*

Input

- On the only input line you will receive the decimal integer number.

Output

- Output a single string - the representation of the input decimal number in it's binary representation.

Constraints

- All numbers will always be valid 32-bit integers.
- Time limit: 0.1s
- Memory limit: 16MB

Sample tests

Input	Output
0	0
3	11
43691	1010101010101011
236476736	1110000110000101100101000000

13 Decimal to Hex

Description

Using loops write a program that converts an integer number to its hexadecimal representation.

- The input is entered as long. The output should be a variable of type string.
- *Do not use the built-in .NET functionality.*

Input

- On the first and only line you will receive an integer in the decimal numeral system.

Output

- On the only output line write the hexadecimal representation of the read number.

Constraints

- All numbers will always be valid 64-bit integers.
- Time limit: 0.1s
- Memory limit: 16MB

Sample tests

Input	Output
254	FE
6883	1AE3
338583669684	4ED528CBB4

14 Hex to Decimal

Description

Using loops write a program that converts a hexadecimal integer number to its decimal form.

- The input is entered as string. The output should be a variable of type long.
- *Do not use the built-in .NET functionality.*

Input

- The input will consists of a single line containing a single hexadecimal number as string.

Output

- The output should consist of a single line - the decimal representation of the number as string.

Constraints

- All numbers will be valid 64-bit integers.
- Time limit: 0.1s
- Memory limit: 16MB

Sample tests

Input	Output
FE	254
1AE3	6883
4ED528CBB4	338583669684

15 GCD

Description

Write a program that calculates the greatest common divisor (GCD) of given two integers A and B.

- Use the Euclidean algorithm (find it in Internet).

Input

- On the first and only line of the input you will receive the 2 integers A and B, separated by a whitespace.

Output

- Output a single number - the GCD of the numbers A and B.

Constraints

- The numbers A and B will always be valid integers in the range `[2, 500]`.
- Time limit: 0.1s
- Memory limit: 16MB

Sample tests

Input	Output
3 2	1
60 40	20
5 15	5

16 Trailing 0 in N!

Description

Write a program that calculates with how many zeroes the factorial of a given number N has at its end.

- Your program should work well for very big numbers, e.g. N = 100000.

Input

- On the only input line, you will receive a single integer - the number N

Output

- Output a single number - the count of trailing zeroes for the N!

Constraints

- N will always be a valid positive integer number.
- Time limit: 0.1s
- Memory limit: 16MB

Sample tests

Input	Output	Explanation
10	2	3628800
20	4	2432902008176640000
100000	24999	think why

17 Spiral Matrix

Description

Write a program that reads from the console a positive integer number N (1 ≤ N ≤ 20) and prints a matrix holding the numbers from 1 to N*N in the form of square spiral like in the examples below.

Input

- The input will always consist of a single line containing a single number - N.

Output

- Output a spiral matrix as described below.

Constraints

- N will always be a valid integer number.
- 1 ≤ N ≤ 20
- Time limit: 0.1s
- Memory limit: 16MB

Sample tests

Input	Output
2	1 2 4 3

3	1 2 3
	8 9 4
	7 6 5

4	1 2 3 4
	12 13 14 5
	11 16 15 6
	10 9 8 7