# Numbers From 1 to N

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

## Sample tests

| Input | Output |
| --- | --- |
| 5 | 1 2 3 4 5 |
| 1 | 1 |

# Not Divisable Number

## 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

## Sample tests

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

# Find Average

You need to calculate the average of a collection of values. Every value will be valid number. The average must be printed with two digits after the decimal point.

## Input

* On the first line, you will receive N - the number of the values you must read
* On the next N lines you will receive numbers.

## Output

* On the only line of output, print the average with two digits after the decimal point.

## Input

4

1

1

1

1

## Output

1.00

## Input

3

2.5

1.25

3

## Output

2.25

# Min Max Sum Average

## Description

Write a program that reads from the console a sequence of N real 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 real 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 real 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 integer numbers that will be in the range [-10000, 10000]

## 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 |

# Calculate Discount

You need to calculate the discounted price for each item in your shopping cart. The discount is 65%, a real deal for you.

## Input

* On the first line, you will receive N - the number of the items in your shopping cart
* On the next N lines you will each item's price

## Output

* On each line in the output print the discounted price of the item with two digits after the decimal point (Math rules for rounding apply)

## Input

2

50

40

## Output

17.50

14.00

## Input

4

9.99

19.99

29.99

39.99

## Output

3.50

7.00

10.50

14.00

# Convert Degrees

You need to provide conversion of the temperature given in Celsius to their Fahrenheit representation.

* [Explanation](https://www.rapidtables.com/convert/temperature/how-celsius-to-fahrenheit.html)

## Hint

* Search how you could split the list of values and then you can iterate over them.
  + [JavaScript](https://www.w3schools.com/jsref/jsref_split.asp)
    - [forEach](https://gomakethings.com/es6-foreach-loops-with-vanilla-javascript/)
  + [Java](https://stackoverflow.com/questions/3214002/splitting-a-space-separated-list)
    - [For-each loop](https://www.geeksforgeeks.org/for-each-loop-in-java/)
  + [C#](https://docs.microsoft.com/en-us/dotnet/csharp/how-to/parse-strings-using-split)
    - [For-each loop](https://www.programiz.com/csharp-programming/foreach-loop)

## Input

* On the first line, you will receive a list of values separated by a single space.

## Output

* On each line in the output print the converted temperature. Print a whole number rounded to the nearest integer (Math rules apply) and without digits after the decimal point.

## Input

0 15 30

## Output

32

59

86

# Print Deck of Cards

## Description

Write a program that reads a card sign(as a string) from the console and generates and prints all possible cards from a [standard deck of 52 cards](http://en.wikipedia.org/wiki/Standard_52-card_deck) 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(10 is 10).
* 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.

## 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 |

# Calculate Complex Sum

## Description

Write a program that, for a given two numbers N and x, calculates the sum S = 1 + 1!/x + 2!/x2 + … + N!/xN.

* 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

## Sample tests

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

# Matrix Numbers

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.

## Input

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

## Output

* See the examples

## Input

2

## Output

1 2

2 3

## Input

3

## Output

1 2 3

2 3 4

3 4 5

# Odd and Even Product

## Description

You are given N integers, each on a new line

* Write a program that checks whether the product of the odd lines is equal to the product of the even lines.
* Lines are counted from 1 to N, so the first line is odd, the second is even, etc.

## Input

* On the first line you will receive the number N
* On each of the next N lines, you will receive a number

## 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 will also be valid integers in range [-1000000, 1000000]

## Sample tests

### Input

5

2

1

1

6

3

### Output

yes 6

### Input

5

4

3

2

5

2

### Output

no 16 15

# Find Maximum Value

Write a program that finds the maximum number from a given sequence.

## Input

* On the first line you will receive one number - N - the count of numbers to follow.
* On the next N lines you will receive the sequence of numbers, each on a new line.

## Output

* Output only one number - the maximum number you find

## Constraints

* 1 <= N <= 20
* -200 <= each number <= 200

## Input

3

1

2

3

## Output

3

## Input

4

5

7

3

6

## Output

7

# Find Largest Three Values

Write a program that, that finds the three largest numbers in a sequence and prints them in descending order in the following format:

{largest}, {second\_largest} and {third\_largest}.

See the example for clarity

## Input

* On the first line you will receive one number - N - the count of numbers to follow.
* On the next N lines you will receive the sequence of numbers, each on a new line.

## Output

* Output the three largest numbers in the already described format.

## Constraints

* 3 <= N <= 20
* -500 <= each number <= 500

## Input

3

3

1

2

## Output

3, 2 and 1

## Input

6

9

11

3

2

1

8

## Output

11, 9 and 8

# Prime Factors

Write a program that finds the prime factors of a given non-prime number. List the factors in ascending order.

* 12 = 2 2 3
* 100 = 2 2 5 \* 5

## Input

* On the first line you will receive one number - N

## Output

* Output each prime factor on a newline.

## Constraints

* 4 <= N <= 1000
* N is guaranteed to not be a prime

## Input

12

## Output

2

2

3

## Input

100

## Output

2

2

5

5