# itertools.product()



## itertools.product()

This tool computes the cartesian product of input iterables.

It is equivalent to nested for-loops.

For example, product (A, B) returns the same as ((x,y) for x in A for y in B).

#### **Sample Code**

```
>>> from itertools import product
>>>
>>> print list(product([1,2,3],repeat = 2))
[(1, 1), (1, 2), (1, 3), (2, 1), (2, 2), (2, 3), (3, 1), (3, 2), (3, 3)]
>>>
>>> print list(product([1,2,3],[3,4]))
[(1, 3), (1, 4), (2, 3), (2, 4), (3, 3), (3, 4)]
>>>
>>> A = [[1,2,3],[3,4,5]]
>>> print list(product(*A))
[(1, 3), (1, 4), (1, 5), (2, 3), (2, 4), (2, 5), (3, 3), (3, 4), (3, 5)]
>>>
>>> B = [[1,2,3],[3,4,5],[7,8]]
>>> print list(product(*B))
[(1, 3, 7), (1, 3, 8), (1, 4, 7), (1, 4, 8), (1, 5, 7), (1, 5, 8), (2, 3, 7), (2, 3, 8), (2, 4, 7), (2, 4, 8), (2, 5, 7), (2, 5, 8), (3, 3, 7), (3, 3, 8), (3, 4, 7), (3, 4, 8), (3, 5, 7), (3, 5, 8)]
```

### Task

You are given a two lists A and B. Your task is to compute their cartesian product  $A \times B$ .

#### **Example**

```
A = [1, 2]
B = [3, 4]
A \times B = [(1, 3), (1, 4), (2, 3), (2, 4)]
```

**Note**: A and B are sorted lists, and the cartesian product's tuples should be output in sorted order.

## **Input Format**

The first line contains the space separated elements of list A.

The second line contains the space separated elements of list B.

Both lists have no duplicate integer elements.

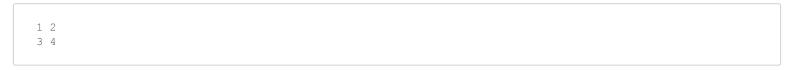
#### **Constraints**

```
0 < A < 30
0 < B < 30
```

### **Output Format**

Output the space separated tuples of the cartesian product.

# **Sample Input**



# **Sample Output**

```
(1, 3) (1, 4) (2, 3) (2, 4)
```