Practical 2
Function, Modules And Abstraction

1) Define a function that can convert a integer into a string and print it in console.
2) Define a function that can receive two integral numbers in string form and compute their sum and then print it in console.
3) Define a function that can accept two strings as input and print the string with maximum length in console. If two strings have the same length, then the function should print al l strings line by line.
4) Define a function which can print a dictionary where the keys are numbers between 1 and 3 (both included) and the values are square of keys.
5) Define a function which can generate a dictionary where the keys are numbers between 1 and 20 (both included) and the values are square of keys. The function should just print the values only.
6) Define a function which can generate and print a list where the values are square of numbers between 1 and 20 (both included).
7) Please write a program to randomly generate a list with 5 numbers, which are divisible by 5 and 7 , between 1 and 1000 inclusive.
8) Please write a program to print the running time of execution of "1+1" for 100 times.
9) The Program Description is given below:

itertools.permutations(iterable[, r])

This tool returns successive length permutations of elements in an iterable.

If is not specified or is None, then defaults to the length of the iterable, and all possible full length permutations are generated.

Permutations are printed in a lexicographic sorted order. So, if the input iterable is sorted, the permutation tuples will be produced in a sorted order.

Sample Code

```
>>> from itertools import permutations
>>> print permutations(['1','2','3'])
<itertools.permutations object at 0x02A45210>
>>>
>>> print list(permutations(['1','2','3']))
[('1', '2', '3'), ('1', '3', '2'), ('2', '1', '3'), ('2', '3', '1'), ('3', '1', '2'), ('3', '2', '1')]
>>>
>>> print list(permutations(['1','2','3'],2))
[('1', '2'), ('1', '3'), ('2', '1'), ('2', '3'), ('3', '1'), ('3', '2')]
>>>
>>> print list(permutations('abc',3))
[('a', 'b', 'c'), ('a', 'c', 'b'), ('b', 'a', 'c'), ('b', 'c', 'a'), ('c', 'a', 'b'), ('c', 'b', 'a')]
```

Task

You are given a string.
Your task is to print all possible permutations of size of the string in lexicographic sorted order.

Input Format

A single line containing the space separated string and the integer value.

Constraints

The string contains only *UPPERCASE* characters.

Output Format

Print the permutations of the string on separate lines.

Sample Input

```
HACK 2
```

Sample Output

```
AC
AH
AK
CA
CH
CK
HA
HC
HK
KA
KC
KH
```

Explanation

All possible size permutations of the string "**HACK**" are printed in lexicographic sorted order.