**Assignment-1**

Write a python program that reads a text file, scramble the words in the file as per the rules given below and write the output to the new text file named by appending the word 'scrabble' to input filename (Consider your own text file as a input).

RULES:

1. Words less than or equal to 3 characters need not be scrabbled.

2. Don't scrabble first and last character.

3. Special characters as last characters are to be maintained as it is. Special characters

like ( . , ; , ? ,, !) eg:- **N**ationa**l.** (here last two characters to be kept as it is.)

4. Maintain sequence of line of text file.

5. Use functions, modules as per requirement.

6. It should be based on functionality, coding standards, modularity.

7. Functionality like number of lines, number of words, vowels in the text file.

**Execute the below code and see the output which is useful for implementing the assignment**

import random

s = 'sakeeb'

mylist = list(s)

random.shuffle(mylist) #RUN THIS STATEMENT AGAIN THEN PRINT mylist AND SEE OUTPUT  
print(mylist)

**Assignment-2**

You are given mobile numbers. Sort them in ascending order then print them in the standard format shown below:

+91 xxxxx xxxxx

The given mobile numbers may have +91, 91 or 0 written before the actual digit number. Alternatively, there may not be any prefix at all.

**Input Format**

The first line of input contains an integer for number of mobile phone numbers.   
lines follow each containing a mobile number.

**Sample Input**

3

07895462130

919875641230

9195969878

**Sample Output**

+91 78954 62130

+91 91959 69878

+91 98756 41230

**Concept**

Like most other programming languages, Python has the concept of closures. Extending these closures gives us decorators, which are an invaluable asset.   
To solve the above question, make a list of the mobile numbers and pass it to a function that sorts the array in ascending order. Make a decorator that standardizes the mobile numbers and apply it to the function.

**Sample Code:**

l = [input() for \_ in range(int(input()))]

def wrapper(f):

def fun(l):

# complete the function

######Write Your Code Here To Get Desired Results##########

return fun

@wrapper

def sort\_phone(l):

print(\*sorted(l), sep='\n')

sort\_phone(l)

**Assignment-3**

You are given some information about people. Each person has a first name, last name, age and sex. Print their names in a specific format sorted by their age in ascending order i.e. the youngest person's name should be printed first. For two people of the same age, print them in the order of their input.  
  
For *Henry Davids*, the output should be:

Mr. Henry Davids

For *Mary George*, the output should be:

Ms. Mary George

**Input Format**

The first line contains the integer to show the number of people.  
lines follow each containing the space separated values of the first name, last name, age and sex, respectively.

**Sample Input**

3

Mike Thomson 20 M

Robert Bustle 32 M

Andria Bustle 30 F

**Sample Output**

Mr. Mike Thomson

Ms. Andria Bustle

Mr. Robert Bustle

**Concept**

For sorting a nested list based on some parameter, you can use:

people.sort(key=lambda li:int(li[index]))

Sample Code:

def person\_lister(f):

def inner(people):

######Write Your Code Here To Get Desired Results##########

return inner

@person\_lister

def name\_format(person):

return ("Mr. " if person[3] == "M" else "Ms. ") + person[0] + " " + person[1]

if \_\_name\_\_ == '\_\_main\_\_':

people = [input().split() for i in range(int(input()))]

print(\*name\_format(people), sep='\n')

**Assignment-4**

Mr. XYZ works in a door mat manufacturing company. One day, he designed a new door mat with the following specifications:

* Mat size must be N x M. (N is an odd natural number, and M is 3 times N)
* The design should have 'WELCOME' written in the center.
* The design pattern should only use |, . and - characters.

**Sample Designs**

Size: 7 x 21

---------.|.---------

------.|..|..|.------

---.|..|..|..|..|.---

-------WELCOME-------

---.|..|..|..|..|.---

------.|..|..|.------

---------.|.---------

Size: 11 x 33

---------------.|.---------------

------------.|..|..|.------------

---------.|..|..|..|..|.---------

------.|..|..|..|..|..|..|.------

---.|..|..|..|..|..|..|..|..|.---

-------------WELCOME-------------

---.|..|..|..|..|..|..|..|..|.---

------.|..|..|..|..|..|..|.------

---------.|..|..|..|..|.---------

------------.|..|..|.------------

---------------.|.---------------

**Input Format**

A single line containing the space separated values of N and M.

**Output Format**

Output the design pattern.

**Sample Input**

9 27

**Sample Output**

------------.|.------------

---------.|..|..|.---------

------.|..|..|..|..|.------

---.|..|..|..|..|..|..|.---

----------WELCOME----------

---.|..|..|..|..|..|..|.---

------.|..|..|..|..|.------

---------.|..|..|.---------

------------.|.------------

# Enter your code here. Read input from STDIN. Print output to STDOUT

li = [eval(x) for x in input().strip().split()]

N,M=li[0],li[1]

# Enter your code here for printing the desired pattern

**Assignment-5**

Given an integer, print the following values for each integer from to :

1. Decimal
2. Octal
3. Hexadecimal (capitalized)
4. Binary

The four values must be printed on a single line *in the order specified above* for each value n.

**Input Format**

A single integer denoting n Decimal Numbers

**Output Format**

Print lines where each line (in the range 1-17 ) contains the respective decimal, octal, capitalized hexadecimal, and binary values of n.

**Sample Input**

17

**Sample Output**

1 1 1 1

2 2 2 10

3 3 3 11

4 4 4 100

5 5 5 101

6 6 6 110

7 7 7 111

8 10 8 1000

9 11 9 1001

10 12 A 1010

11 13 B 1011

12 14 C 1100

13 15 D 1101

14 16 E 1110

15 17 F 1111

16 20 10 10000

17 21 11 10001