Simple Averages

Create a program that allows a user to enter 1 to 10 integer numbers on a single line of input (you may assume that the input contains no more than 10 numbers and no less than 1 number).
Compute both the average and median of the list and display the result.

Example 1:
Input list: 1 10 13
Average: 8
Median: 10

Example 2:
Input list: 1 2 3 4 5 6 7 8 9 10
Average: 5.5
Median: 5.5

Calculate average of given set of Numbers

```
>>> givenNumbers = "1 2 3 4 5 6 7 8 9 10"
>>> numList = givenNumbers.split(" ")
>>> sum = 0
>>> for i in numList:
    sum = sum + int(i)

>>> average = sum / len(numList)
>>> print average
5
```

Change to give float as answer.
Change to take input from the user.

Factorial Calculator

If n is an integer greater than 0, n factorial (n!) is the product: n* (n-1) * (n-2) * (n-3)... * 1. By convention, 0! = 1. You must write a program that allows a user to enter an integer between 1 and 7. Your program must then compute the factorial of the number entered by the user.
Your solution MUST actually perform a computation (i.e., you may not simply print “5040” to the screen as a literal value if the input is 7).

Example 1:

```
Enter a number: 4
4! = 24
```

Example 2:

```
Enter a number: 7
7! = 5040
```

```
givenNo = 1
fact = 1

if givenNo != 0:
    for i in range(1,int(givenNo)+1):
        fact = fact * i
else:
    fact = 1

print fact
```

Change to set range of givenNo between 1 and 7.
Change to give input from the user.

Diamond Printer

This program will print a shape on the screen using asterisks (“*”) characters. The user will be prompted to enter an ODD number between 1 and 99 (you may assume test cases will not be beyond this range and that all test cases will be odd numbers). The shape that will be printed resembles a diamond, where the number provided by the user represents the number of *’s printed on the middle line. The line above and below will be centered and will have 2 less *’s than the middle line. This reduction by 2 *’s for each line continues until a line with a single * is printed at the top and bottom of the figure.

Example 1:

```
Enter an odd number (1-100): 1
*
```

Example 2:

```
Enter an odd number (1-100): 5
   *
 ***
*****
 ***
  *
```

```
sizeOfDiamond = 5
i = 1

while i <= sizeOfDiamond:
    print "*" * i
    i = i + 2

i = 2
while i < sizeOfDiamond:
    print "*" * (sizeOfDiamond - i)
    i = i + 2
```

```
import random

wordDict = {    'orange' : 'fruit for winter',
            'apple' : 'fruit that keeps the doctor away',
            'momo' : 'Nepal\' national food',
            'sekuwa' : 'drunken delights',
            'boost' : 'secret of my energy',
            'rain' : 'falls from the sky'}

randomSelection = random.randint(0,len(wordDict)-1)
selectedWord = wordDict.keys()[randomSelection]
selectedMeaning = wordDict[selectedWord]

noOfLettersInWord = len(selectedWord)
print selectedWord, selectedMeaning
print "__  " * noOfLettersInWord
```

String Library

```
>>> import string
>>> dir(string) → shows functions associated with string module
>>> help('string.lower') → shows help on what the lower function does
>>> string.lower.__doc__ → shows the function's docs
>>> dir(__builtins__) → shows builtin functions
```

Simples examples of String operations:

Split a string of date format into dd-mm-yyyy

```
>>> givenDate = “12-11-2010”
>>> day, month, year = givenDate.split(“-”)
```

Question : write a function that will take a string with date in dd-mm-yyyy or dd/mm/yyyy format and return back a day, month and year.

Question : in a given string shift every character by 13 spaces. A becomes N, B becomes O, and so on up to M, which becomes Z, then the sequence reverses: N becomes A, O becomes B, and so on to Z, which becomes M.

```
givenStr = "example one" #"abctsp"
newStr = ""

for i in givenStr:
    iNum = ord(i)
    iNum = iNum + 13
    if iNum > 122:
        iNum = (iNum - 122) + 96
    else:
        pass
    newStr = newStr + chr(iNum)

print newStr
```

Change for capital letters. Change for both capital and small letters.