1 Python Assignment

1.1 To convert string 'Light is faster than sound' to 'LIGHT Is Faster Than SOUND'

In [1]:

```
my_str = "Light is faster than sound"
l1 = my_str.title()
l2 = l1.replace("Light","LIGHT")
l3 = l2.replace("sound","SOUND")
print(l3)
```

```
LIGHT Is Faster Than Sound
```

1.2 To check given word is palindrome or not

In [1]:

```
lst1 =input("Enter the word : ")
lst1 = lst1.lower()
lst2 = reversed(lst1)

if list(lst1) == list(lst2):
    print("The given word is a palindrome")
else:
    print("The given word is not a palindrome")
```

```
Enter the word : Madam
The given word is a palindrome
```

1.3 To find given number is a Prime number or not

In [3]:

```
num= int(input("Enter the number : "))
prime = True
for i in range(2,num):
    if (num % i) == 0:
        prime = False
        break
if prime:
    print("Given number {} is Prime".format(num))
else:
    print("Given number {} is not Prime".format(num))
```

```
Enter the number : 25
Given number 25 is not Prime
```

1.4 Check largest of the three numbers

1.4.1 Using loop

In [4]:

```
numb1 = int(input("Enter the first number : "))
numb2 = int(input("Enter the second number : "))
numb3 = int(input("Enter the third number : "))

if numb1 > numb2 and numb1 > numb3:
    greatest = numb1
elif numb2 > numb1 and numb2 > numb3:
    greatest = numb2
else:
    greatest = numb3

print("The greatest of the three numbers is",greatest)
```

```
Enter the first number : 15
Enter the second number : 15
Enter the third number : 25
The greatest of the three numbers is 25
```

1.4.2 Using function

In [5]:

```
def maximum(num1,num2,num3):
    if num1>num2 and num1>num3:
        return num1
    elif num2>num1 and num2>num3:
        return num2
    else:
        return num3

m = maximum(15,89,52)
print("The greatest of the given numbers is",m)
```

```
The greatest of the given numbers is 89
```

1.5 Pass and continue statements

In [6]:

```
for i in range(1,8):
    pass
       #pass means none // without pass program will give an error
```

In [7]:

```
for i in range(1,12):
    if i==5:
        continue
    print(i)
```

```
1
2
3
4
6
7
8
9
10
11
```

1.6 Print 1 to 10 on same line breaking an infinite loop

In [8]:

```
for i in range(1,20):
    if i==11:
        break
    print( i , end=" ")
```

```
1 2 3 4 5 6 7 8 9 10
```

1.7 Print all the unique combinations of 1,2 and 3

In [9]:

```
from itertools import permutations

per = permutations([1,2,3])

for i in list(per):
    print(i)
```

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

1.8 To find out the sum of the digits of a number

In [10]:

```
number = input("Enter the number : ")
sum = 0
for i in list(number):
    k= int(i)
    sum = sum + k
    print(int(i))
print("Sum of the digits of {} is {}".format(number,sum))
```

```
Enter the number : 1569
1
5
6
9
Sum of the digits of 1569 is 21
```

1.9 WAP to sort the list in descending order

1.9.1 (don't use any inbuilt function)

In [11]:

```
list_A = [5,3,1,2,4,0]
newList = []
while list_A:
    min = list_A[0]
    for x in list_A:
        if x<min:
            min = x
    newList.append(min)
    list_A.remove(min)
newList.reverse()
print(newList)
```

```
[5, 4, 3, 2, 1, 0]
```

1.10 Remove the numbers that occurs more than once.

1.11 Shallow and Deep copying Program

In [12]:

```
import copy
li_1 = [1,2,[2,3],4]
```

1.11.1 Shallow Copying

In [13]:

```
li1 = copy.copy(li_1)
li1
```

Out[13]:

```
[1, 2, [2, 3], 4]
```

1.11.2 Deep copying

In [14]:

```
li2 = copy.deepcopy(li_1)
li2
```

Out[14]:

```
[1, 2, [2, 3], 4]
```

1.12 *WAP to do the following

1.12.1 Creating a list of bank accounts with name alone

1.13 Create 3 tuples one of name ,age or salary and group them using zip function.

1.14 Using unpacking operator as well as indexing approach

In [15]:

```
name = ("Sakshi","Priya","Riya")
age = (21,22,20)
salary = (2255546,23000,23233)
customer = zip(name,age,salary)
#print(*customer)
print(set(customer))
```

```
{('Priya', 22, 23000), ('Sakshi', 21, 2255546), ('Riya', 20, 23233)}
```

1.15 Using list comprehension and random number generation between 10 to 100

1.15.1 Print Even numbers

In [16]:

```
import numpy as np
my_random = np.random.randint(10,100,90)
values = [x for x in my_random if x%2 == 0]
print("The even numbers from the generated random numbers are : ",values)
```

```
The even numbers from the generated random are :  [70, 66, 34, 52, 68, 80, 7
4, 82, 64, 62, 76, 60, 32, 94, 88, 98, 60, 88, 32, 36, 26, 10, 86, 36, 22, 8
4, 46, 28, 78, 42, 34, 42, 56, 98, 74, 60, 52, 46, 32, 48, 56, 84, 94, 16, 5
4, 50, 92, 16, 22, 94, 44]
```

1.15.2 Print Odd numbers

In [17]:

```
import numpy as np
random_no = np.random.randint(10,100,90)
no_1 = [x for x in random_no if x%2 != 0]
print("The odd numbers from the generated random numbers are : ",no_1)
```

```
The odd numbers from the generated random numbers are :  [63, 55, 65, 47, 5
5, 69, 85, 71, 11, 15, 93, 65, 45, 15, 65, 79, 39, 57, 45, 11, 95, 63, 23, 6
1, 39, 65, 19, 75, 53, 85, 29, 99, 85, 75, 39, 33, 33, 13, 35, 91, 69, 57, 4
3, 13, 69, 77, 49]
```

1.15.3 Print numbers divisible by 3

In [18]:

```
rand_no = np.random.randint(10,100,90)
no_2 = [x for x in rand_no if x%3 == 0]
print("The number divisible by 3 are : ",no_2)
```

```
The number divisible by 3 are :  [81, 45, 33, 75, 72, 96, 63, 63, 15, 30, 9
0, 21, 48, 21, 15, 84, 66, 57, 18, 21, 33, 18, 12, 21, 81, 48, 33]
```

1.16 Two matrices of size 3x3 each, perform the following matrix operations

1.16.1 Matrix Addition

In [19]:

```
import numpy as np
mat1 = np.array([[1,2,3],[1,2,3],[1,2,3]])
mat2 = np.array([[1,2,3],[1,2,3],[1,2,3]])
mat3 = np.array([[1,2,3],[1,2,3],[1,2,3]])
np.sum([mat1,mat2,mat3])
```

Out[19]:

```
54
```

1.16.2 Matrix subtraction

In [20]:

```
mat1 = np.array([[1,2,3],[1,2,3],[1,2,3]])
mat2 = np.array([[2,2,3],[2,2,3],[2,2,3]])
mat3 = np.array([[1,2,3],[1,2,3],[1,2,3]])
np.subtract(mat1,mat2,mat3)
```

Out[20]:

```
array([[-1,  0,  0],
       [-1,  0,  0],
       [-1,  0,  0]])
```

1.16.3 Matrix multiplication / dot product

In [21]:

```
m1 = mat2.dot(mat3)
m1
```

Out[21]:

```
array([[-7,  0,  0],
       [-7,  0,  0],
       [-7,  0,  0]])
```

1.16.4 Matrix Transpose

In [22]:

```
m2 = mat1.transpose()
m2
```

Out[22]:

```
array([[1, 1, 1],
       [2, 2, 2],
       [3, 3, 3]])
```

1.17 interchange any two rows

In [23]:

```
a = np.array([[4,3, 1],[5 ,7, 0],[9, 9, 3],[8, 2, 4]])

a[[0, 2]] = a[[2, 0]]
print(a)
```

```
[[9 9 3]
 [5 7 0]
 [4 3 1]
 [8 2 4]]
```

1.18 Join tuples if similar initial element

In [24]:

```
# Python3 code to demonstrate working of
# Join Tuples if similar initial element
# Using loop

# initializing list
test_list = [(5, 6), (5, 7), (6, 8), (6, 10), (7, 13)]

# printing original list
print("The original list is : " + str(test_list))

# Join Tuples if similar initial element
# Using loop
res = []
for sub in test_list:
    if res and res[-1][0] == sub[0]:
        res[-1].extend(sub[1:])
    else:
        res.append([ele for ele in sub])
res = list(map(tuple, res))

# printing result
print("The extracted elements : " + str(res))
```

```
The original list is : [(5, 6), (5, 7), (6, 8), (6, 10), (7, 13)]
The extracted elements : [(5, 6, 7), (6, 8, 10), (7, 13)]
```

1.19 Sort the lists in tuples - eg

In [25]:

```
test_tup = ([7,5,4],[8,2,4],[0,7,5])
print ("The original tuple is : " +str(test_tup))

res = tuple((sorted(sub) for sub in test_tup))
print("The tuple after sorting lists : "+ str(res))
```

```
The original tuple is : ([7, 5, 4], [8, 2, 4], [0, 7, 5])
The tuple after sorting lists : ([4, 5, 7], [2, 4, 8], [0, 5, 7])
```

1.20 Given a list,

1.20.1 Find all possible subsets and store them in a list of lists (of length 2^n)

In [26]:

```
### Correct syntax
def powersets(s):
    x = len(s)
    for i in range (1 << x):
        print([s[j] for j in range(x) if (i & (1 << j))])
powersets([1,2,3])
```

```
[]
[1]
[2]
[1, 2]
[3]
[1, 3]
[2, 3]
[1, 2, 3]
```

In [27]:

```
# 2
#from itertools import chain,combinations
#def powerset(iterable):
#    "powerset ([1,2,3])--> () (1,) (2,) (3,) (1,2) "
#    s = list(iterable)
#   return chain.fromm_iterable(combinations(s,r) for ir in range(1,len(s)+1))
```

In [28]:

```
# 3
import itertools
def findsubsets(s,n):
    return list(itertools.combinations(s,n))

s = [1,2,3]
n = len(s)
print(findsubsets(s,n))
```

```
[(1, 2, 3)]
```

1.20.2 b) Demonstrate the following

1.20.2.1 i) Insert more elements into it

1.20.2.2 ii) Find and replace

1.20.2.3 iii) Delete individual elements tillthe sublist becomes empty

1.20.2.4 iv)Sorting

1.21 Given two lists A and B of integers. Sort the lists A and Btogether, with respect to the list A. (Use zip and "sorted" for sorting)

In [29]:

```
A = [1,3,2]
B = [6,-2,-1]
A = sorted(A)
zipping = zip(A,B)
print(*zipping)
```

```
(1, 6) (2, -2) (3, -1)
```

1.22 Remove duplicate of string with and without set

1.22.1 With set()

In [30]:

```
string = "Hello"
string = sorted(set(string))
print(string)
```

```
['H', 'e', 'l', 'o']
```

1.22.2 Without set

In [31]:

```
my_string = "hello hey"
my_string = list(my_string)
print("List Before ", my_string)
temp_list = []

for i in my_string:
    if i not in temp_list:
        temp_list.append(i)

my_string = temp_list

print("List After removing duplicates " +str(my_string))
```

```
List Before  ['h', 'e', 'l', 'l', 'o', ' ', 'h', 'e', 'y']
List After removing duplicates ['h', 'e', 'l', 'o', ' ', 'y']
```

1.23 for a dictionary, arrange them in the alphabetical order of the keys,insert afew key:value pair 1)the alphabetical order of keys 2)sort using "values"

In [32]:

```
diction = {"Sakshi":12,"Age":21}
diction.update({"Salary":2000000})
print(sorted(diction))
print(sorted(diction.values()))
```

```
['Age', 'Sakshi', 'Salary']
[12, 21, 2000000]
```

1.24 *Make a grade sheet

1.25 *Nested Dictionary

1.25.1 Without description

In [33]:

```
students_dict = { 1:{"Name": "HeMan","Gender": "Male"},
           2: {"Name": "Sakshi","Gender": "Female"}}
res = {key : dict(sorted(val.items(), key = lambda ele: ele[1]))
       for key, val in students_dict.items()}
print("The sorted dictionary : " + str(res))
```

```
The sorted dictionary : {1: {'Name': 'HeMan', 'Gender': 'Male'}, 2: {'Gende
r': 'Female', 'Name': 'Sakshi'}}
```

1.25.2 With description

In [34]:

```
# Python3 code to demonstrate working of
# Sort Nested keys by Value
# Using sorted() + generator expression + lamda

# initializing dictionary
test_dict = {'Nikhil' : {'English' : 5, 'Maths' :  2, 'Science' : 14},
             'Akash' : {'English' : 15, 'Maths' :  7, 'Science' : 2},
             'Akshat' : {'English' : 5, 'Maths' :  50, 'Science' : 20}}

# printing original dictionary
print("The original dictionary : " + str(test_dict))

# Sort Nested keys by Value
# Using sorted() + generator expression + lamda
res = {key : dict(sorted(val.items(), key = lambda ele: ele[1]))
       for key, val in test_dict.items()}

# printing result
print("The sorted dictionary : " + str(res))
```

```
The original dictionary : {'Nikhil': {'English': 5, 'Maths': 2, 'Science': 1
4}, 'Akash': {'English': 15, 'Maths': 7, 'Science': 2}, 'Akshat': {'Englis
h': 5, 'Maths': 50, 'Science': 20}}
The sorted dictionary : {'Nikhil': {'Maths': 2, 'English': 5, 'Science': 1
4}, 'Akash': {'Science': 2, 'Maths': 7, 'English': 15}, 'Akshat': {'Englis
h': 5, 'Science': 20, 'Maths': 50}}
```

1.26 Find even or odd

In [35]:

```
n = int(input("Please enter a number : "))
if n%2 == 0:
    print("The given number is even")
else:
    print("The given number is odd")
```

```
Please enter a number : 25
The given number is odd
```

1.27 no.of vowels and consonants

In [36]:

```
str1 = input("Please enter a word : ")
vowels = 0
consonants = 0
for i in str1:
    if i == "a" or i=="e" or i=="i" or i=="o" or i=="u" or i=="A" or i=="E" or i=="I" or i
        vowels = vowels + 1
    else:
        consonants +=1
print("The number of vowels in the given word : ",vowels)
print("The number of consonants in the given word : ",consonants)
```

```
Please enter a word : handsome
The number of vowels in the given word :  3
The number of consonants in the given word :  5
```

1.28 *Program to find given matrix is symmetric or not

In [37]:

```
def symmetric(a,n1):
    for i in range(n1):
        for j in range(n1):
            if (a[i][j]) != (a[j][i]):
                return False
    return True

a = [[1,2,3],[3,4,5],[6,7,8]]
print("Given matrix : ")
print(a)

if (symmetric(a,3)):
    print("Given matrix is symmetric")
else:
    print("Given matrix is not symmetric")
```

```
Given matrix :
[[1, 2, 3], [3, 4, 5], [6, 7, 8]]
Given matrix is not symmetric
```

1.29 Use filter and lambda to find all the ages greater than or equal to 18.

In [38]:

```
ages = [81,23,10,19,91,25,55,41,49,60,18,32,65,10,12,13,1,2,3]
greater_= list(filter((lambda x : x >= 18),ages))
print(greater_)
```

```
[81, 23, 19, 91, 25, 55, 41, 49, 60, 18, 32, 65]
```

1.30 Map and reduce

1.30.1 Map

In [39]:

```
ages = [81,23,10,19,91,25,55,41,49,60,18,32,65,10,12,13,1,2,3]
greater_= list(map((lambda x : x >= 18),ages))
print(greater_)
```

```
[True, True, False, True, True, True, True, True, True, True, True, True, Tr
ue, False, False, False, False, False, False]
```

1.30.2 Reduce

In [40]:

```
from functools import reduce
ages = [81,23,10,19,91,25,55,41,49,60,18,32,65,10,12,13,1,2,3]
greater_=reduce((lambda x,y : x + y),ages)
print(greater_)
```

```
610
```