JLUFE Spring 2021 (Feb-July)

Final Assignment Report

JILIN UNIVERSITY OF FINANCE AND ECONOMICS

Department of College of Managment Science and Information Engineering

BSc in Information management and information system

(2021)

Final Assignment: Part 01

21/06/2021

MODULE: Data Mining

Submitted by: Clark(郭旭) 0314021805405 (1854)
QQ: 1145673148
Github ID: Clark5405

Instructions:

1. I have added tips and required learning resources for each question, which helps you to solve the exercise.
2. Finish the assignment on your **OWN** (**Any student find copying/sharing from classmates or internet will get '0' points!!!**)
3. **Accept this assignment** from the Github Clasroom link (https://classroom.github.com/a/yFXO50A4) This will create private repository of the assignment in your Github account.
4. In your repository `Clone` -> `Download ZIP` in your computer.
5. Change your: **Major**, **Name**, **Student number**, **Class number**, **QQ number** and **GitHub ID**
6. Once you finish the Assignment convert your .ipynb file into PDF (https://github.com/milaan9/91_Python_Tips/blob/main/000_Convert_Jupyter_Notebook_to_PDF.ipyn (both .pynb and .pdf file will be required!)
7. Create Folder name "**Solution**" and copy your 3 files:
   A. Your Jupyter Notebook file (**.ipynb**).
   B. Your PDF converted file (**.pdf**).
   C. **.zip** file containing both .ipynb and .pdf files and name your .zip file as your student number and name. For example: **0318021907632 Milan(米兰).zip**
8. Finally, in your repository `Add files` -> `upload files` upload the "**Solution**" folder and `Commit changes`.

Python Assignment 01

Question 1:

Write a python program that generates a list containing only common elements between the two lists (without duplicates). Make sure your program works on two lists of different sizes.

Expected Output:

```
List 1: [0, 2, 4, 6, 12, 13, 14, 18, 20, 24, 25, 26, 27]
List 2: [0, 4, 7, 9, 10, 11, 13, 14, 17, 18, 20, 33, 39]
List of common elements are:  [0, 4, 13, 14, 18, 20]
```

For extra points:

1. Generate the two list randomly to test this
2. Generate each list in one line of Python.

In [8]:

```
# Solution 1:
list1=[0,2,4,6,12,13,14,18,20,24,25,26,27]
list2=[0,4,7,9,10,11,13,14,17,18,20,33,39]
li=set.intersection(set(list1),set(list2))
print(list(li))
```

```
[0, 4, 13, 14, 18, 20]
```

Question 2:

Write a python program to find the gravitational force acting between two objects.

$$F = G\frac{m_1 m_2}{r^2}$$

Expected Output:

```
Enter the first mass (m1): 5000000
Enter the second mass (m2): 900000
Enter the distance between the centres of the masses (N): 30
Hence, the Gravitational Force is:  0.33 N
```

In [1]:

```
# Solution 2:
import math
from scipy.constants import G
t=float(input('输入第一个质量（m1）：'))
t2=float(input('输入第二个质量（m2）：'))
t3=float(input('输入质心之间的距离（N）：'))
f=G*t*t2/(t3*t3)
print("因此，重力为：" +str(f))
```

```
输入第一个质量（m1）：5000000
输入第二个质量（m2）：900000
输入质心之间的距离（N）：30
因此，重力为：0.33370400000000006
```

Question 3:

Write a python program that generates a new list that contains only even elements from the randomly generated list.

Expected Output:

```
Randomly generated list: [64, 63, 90, 13, 38, 27, 19, 51, 97, 32, 18, 75]
List of even elements: [64, 90, 38, 32, 18]
```

In [6]:

```
# Solution 3:
from random import randint
li=[]
ou=[]
for i in range(10):
    li.append(randint(0,100))
for i in li :
    if i%2==0:
        ou.append(i)
print('随机生成的列表：{}'.format(li))
print('偶数元素列表：：{}'.format(ou))
```

```
随机生成的列表：[4, 40, 11, 30, 83, 63, 97, 6, 63, 76]
偶数元素列表：：[4, 40, 30, 6, 76]
```

Question 4:

Write a python program to check if a substring is present in a given string.

Expected Output:

```
Enter string: Hello world
Enter word: world
Substring in string!
```

In [9]:

```
# Solution 4:
# Solution 4:
i=input('输入字符串：')
ii=input('输入单词：')
if ii in i:
    print('字符串中的子字符串！')
else:
    print('不是字符串中的子字符串！')
```

```
输入字符串：Hello world
输入单词：world
字符串中的子字符串！
```

Question 5:

Write a python program that asks the user last 2 digit of (your) student number and generates Fibonacci series.

Expected Output:

```
How many numbers that generates?: 12
Fibonacci series:
 [1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144]
```

In [43]:

```
# Solution 5:
d =int(input("生成多少个数字? :"))-2
gx=[1,1,]
a=1
b=1
for d in range(d):
     c = a + b
     gx.append(c)
     a = b
     b = c
print(gx)
```

```
生成多少个数字? :5
[1, 1, 2, 3, 5]
```

Question 6:

Write a python program using function that generates a new list that contains all the elements of the first list and removing all the duplicates.

Expected Output:

```
List:  [1, 2, 3, 4, 3, 2, 1]
Result List using loop:  [1, 2, 3, 4]
Result List using sets:  [1, 2, 3, 4]
```

For extra points:

1. Generate the result using two different functions using:
    - one using a loop and constructing a list
    - sets

In [4]:

```
# Solution 6:
def gener_li():
    from random import randint
    li=[]
    for i in range(10):
        li.append(randint(0,5))
    return li
li=gener_li()
print(li)
c=[]
for i in  range(len(li)):
    if  li[i] in li[i+1:]  :
        c.append(li[i])
print('使用循环的结果列表：{}'.format(list(set(c))))
li1=set(li[:5])
li2=set(li[5:])
li3=list(set.intersection(li1,li2))
print('使用集合的结果列表：{}'.format(li3))
```

```
[3, 3, 0, 0, 5, 0, 4, 2, 2, 5]
使用循环的结果列表：[0, 2, 3, 5]
使用集合的结果列表：[0, 5]
```

Question 7:

Write a python program using functions that asks the user for a string containing multiple words and print back to the user the same string, except with the words in reverse order.

Expected Output:

```
Please enter a sentence: My name is Milaan
The reverse sentence is: Milaan is name My
```

In [3]:

```
# Solution 7:
i=input('Please enter a sentence: ')
t=i.split(' ')
s=''
t.reverse()
for i in t:
    s+=(i+' ')
print(s)
```

```
Please enter a sentence: My name is Clark
Clark is name My
```

Question 8:

Write a python program using function that encrypts a given input with these steps:

Input: "apple"

- Step 1: Reverse the input: "elppa"
- Step 2: Replace all vowels using the following chart:

```
a => 0
e => 1
i => 2
o => 2
u => 3
# 1lpp0
```

- Step 3: Add "aca" to the end of the word: "1lpp0aca"

Expected Output:

```
Word:  apple
Encrypted word: 1lpp0aca
```

More Examples:

```
encrypt("banana") ➞ "0n0n0baca"
encrypt("karaca") ➞ "0c0r0kaca"
encrypt("burak") ➞ "k0r3baca"
encrypt("alpaca") ➞ "0c0pl0aca"
```

In [2]:

```
# Solution 8:
i=input('Input: ')
s=i[::-1]
p=''
d={'a':0,'e':1,'i':2,'o':2,'u':3}
for i in s:
        p+='{}'.format(d.get(i,i))
p=p+'aca'
print(p)
```

```
Input: apple
1lpp0aca
```

Question 9:

Write a python program using function that takes a number num and returns its length.

Expected Output:

```
    Enter number: 963969
    Total digits in given number:  6
```

In [7]:

```
# Solution 9:
def sun(n):
     return len(n)
t=sun(input("输入号码："))
print('给定数字的总位数：{}'.format(t))
```

```
输入号码：963969
给定数字的总位数：6
```

Question 10:

Write a python program using function that takes a string and returns the number (count) of vowels contained within it.

Expected Output:

```
    Enter string: Celebration
    Total vowels in the string: 5
    Identified vowels are:  ['e', 'e', 'a', 'i', 'o']
```

More examples:

```
count_vowels("Palm") ➞ 1
count_vowels("Prediction") ➞ 4
```

In [8]:

```python
# Solution 10:
def c(t):
    j=0
    for i in t:
        if i in ['e','i','u','a','o']:
            j=j+1
    return j
j=c(input('输入字符串'))
print('字符串中的元音总数：{}'.format(j))
```

```
输入字符串eeaio
字符串中的元音总数：5
```

Question 11:

Write a python program to draw pattern as below:

Expected Output:

```
--- --- ---
|   |   |   |
 --- --- ---
|   |   |   |
 --- --- ---
|   |   |   |
 --- --- ---
```

For extra point:

1. Generate solution by asking the user what size game board they want to draw, and draw it for them to the screen using Python's print statement.

Expected Output:

```
Enter the size of board you want to draw: 4
 --- --- --- ---
|   |   |   |   |
 --- --- --- ---
|   |   |   |   |
 --- --- --- ---
|   |   |   |   |
 --- --- --- ---
|   |   |   |   |
 --- --- --- ---
```

In [1]:

```
# Solution 11:
for i in range(6):
    if i%2==0:
        print('--- --- --- ')
    else:
        print('|   |   |   |')
```

```
--- --- ---
|   |   |   |
--- --- ---
|   |   |   |
--- --- ---
|   |   |   |
```

Question 12:

Write a python program to ask user for a string and then perform following operations:

1. Calculate the num of digits
2. Calculate the num of characters
3. Calculate the num of vowels
4. Calculate the num of lowercase letters
5. replace ' ' with '_' in the string
6. Print and Store the ouput to 'output.txt' file.

Expected Output:

```
Enter string: Hello World 123
Output printed in'output.txt'
```

Expected Output in output.txt:

```
The entered string is: Hello World 123
The number of digits is: 3
The number of characters is: 15
The number of vowels is: 3
The number of lowercase letters is: 8
The modified string is: Hello_World_123
```

In [13]:

```
# Solution 12:
s=input('输入的字符串是：')
print('位数为：{}'.format(len(s.split(' '))))
print('字符数为：{}'.format(len(s)))
k=0
x=0
for  i in s:
     if i in ['e','i','u','a','o']:
         k+=1
     elif i in'qwertyuioplkjhgfdsazxcvbnm':
         x+=1
print('元音的数目是：{}'.format(k))
print('小写字母的数目是：{}'.format(x))
t=s.replace(' ',r'_ ')
print('修改后的字符串是：{}'.format(t))
f=open('output.txt','w')
f.write(t)
f.write('\n')
```

```
输入的字符串是：Hello World 123
位数为：3
字符数为：15
元音的数目是：3
小写字母的数目是：5
修改后的字符串是：Hello_ World_ 123
```

Out[13]:

```
1
```

Question 13:

Write a python program using function that takes as input three variables from user, and returns the largest of the three. Do this without using the Python `max()` function!

Expected Output: Please enter three integers separated by comma: 12, 66, 31 The maximum value is: 66

In [17]:

```
# Solution 13:
def g():
    t=input('请输入三个以逗号分隔的整数：').split(',')
    return t
t=g()
for i in range(2):
    n=t.index(min(t))
    del t[n]
print('最大值为：{}'.format(t[0]))
```

```
请输入三个以逗号分隔的整数：66,12,31
最大值为：66
```

Question 14:

Write a python program where user, will have a number in head between 0 and 100. The program will guess a number, and you, the user, will say whether it is too "high", too "low", or your number. Also, in the end program should print out how many guesses it took to get your number.

Expected Output:

```
Guess a number between 0 and 100 and tell whether high or low when prompted!
My guess is 50. Is that high, low or same? low
My guess is 75. Is that high, low or same? low
My guess is 88. Is that high, low or same? low
My guess is 94. Is that high, low or same? low
My guess is 97. Is that high, low or same? low
My guess is 99. Is that high, low or same? same
Congrats to me! I guessed it in 6 tries.
```

In [21]:

```
# Solution 14:
import random
def guess_number():
    true_num = random.randint(1, 100)
    user_num = int(input("请输入一个整数:"))
    count = 1
    while true_num != user_num:
        if true_num > user_num:
            print("太小了，请重新输入！")
        elif true_num < user_num:
            print("太大了，请重新输入！")
        count += 1
        user_num = int(input("请输入一个整数："))
    print("恭喜您，您猜对了！您一共猜了%d次" % count)

guess_number()
```

```
请输入一个整数:50
太小了，请重新输入！
请输入一个整数：80
太大了，请重新输入！
请输入一个整数：60
太小了，请重新输入！
请输入一个整数：70
太小了，请重新输入！
请输入一个整数：75
太大了，请重新输入！
请输入一个整数：72
太小了，请重新输入！
请输入一个整数：73
太小了，请重新输入！
请输入一个整数：74
恭喜您，您猜对了！您一共猜了8次
```

Question 15:

Write a python program using function that takes an list(ordered) of numbers (from smallest to largest) and another number. The function decides whether or not the given number is inside the list and returns (then prints) an appropriate boolean.

> Hint: Use binary search.

Expected Output:

```
List:  [2, 4, 6, 8, 10]
Find  '5': False
Find '10': True
Find '-1': False
Find  '2': True
```

For extra point:

1. Generate list randomly and select he number randomly to be search from the list.

In [2]:

```
# Solution 15:
def g(li,n):
    if n in li:
        print(True)
    else:
        print(False)
li=input('列表：')
t=input('查找')
g(li,t)
```

```
列表：2，4，6，8
查找5
False
```

Question 16:

Write a python program to generate password. Be creative with how you generate passwords - strong passwords have a mix of lowercase letters, uppercase letters, numbers, and symbols. The passwords should be random, generating a new password every time the user asks for a new password. Include your code in a main method.

Expected Output:

```
Please choose strong or weak:
strong
password: 6|Av.0T^9
do you want a new password? y/n
n
```

For extra points:

1. Ask the user if they want password to be strong(9 characters) or weak(6 characters)?

In [4]:

```
# Solution 16:
import random
def main(n):
    k = ''
    for i in range(8):
        m = random.choice(n)[0]
        k = k + m
    print(k)
while True:
    t=input('请选择强或弱：')
    if t=='强':
        main("asdfgh&jklpowqzxcvbnm234QWAZSX NMiuytreJKLPOIU:,.?+_)(*^%$@!15EDCRFVTGBYHUJ67890.]")
        y=input('你想要一个新密码吗？N/Y')
    else:
        main("123456789qwertyausidffghijokplmznxbcv")
        y = input('你想要一个新密码吗？N/Y')
    if y=='N':
        break
```

```
请选择强或弱：强
p%Q2+cim
你想要一个新密码吗？N/YY
请选择强或弱：弱
ahmzva7t
你想要一个新密码吗？N/YN
```

Question 17:

Write a python program using function that picks a random word from a list of words from the dictionary (https://github.com/milaan9/92_Python_Assignments/blob/main/sowpods.txt). Each line in the file contains a single word.

> Hint: use the Python random library for picking a random word.

Expected Output:

```
Random word: POTENTIATING
```

In [11]:

```
# Solution 17:
def main():
import random
f=open('SUN_Database.txt','r')
n=f.readlines()
return random.choices(n)[0]
print('随机词：{}'.format(main()))
```

```
  File "<ipython-input-11-937ac12eac1c>", line 3
    import random
         ^
IndentationError: expected an indented block
```

Question 18:

Write a python program where a text(.txt) file is given **nameslist.txt (https://github.com/milaan9/92_Python_Assignments/blob/main/nameslist.txt)** that contains list of a bunch of names, count how many of each name there are in the file, and print out the results to the screen.

Expected Output:

```
{'Darth': 31, 'Luke': 15, 'Leia': 54}
```

For extra point:

1. Instead of using the **nameslist.txt (https://github.com/milaan9/92_Python_Assignments/blob/main/nameslist.txt)** file from above (or instead of, if you want the challenge), take this **SUN_Database.txt (https://github.com/milaan9/92_Python_Assignments/blob/main/SUN_Database.txt)** file, and count how many of each “category” of each image there are. This text file is actually a list of files corresponding to the SUN database scene recognition database, and lists the file directory hierarchy for the images. Once you take a look at the first line or two of the file, it will be clear which part represents the scene category.

Expected Output:

```
abbey: 50
airplane_cabin: 50
airport_terminal: 50
alley: 50
amphitheater: 50
...
...
...
wrestling_ring: 50
yard: 50
youth_hostel: 50
```

In [30]:

```
# Solution 18:
f=open('nameslist.txt','r')
t=f.readlines()
print(t)
l=0
k=0
j=0
for i in t:
    if i =='Darth\n':
        l+=1
    elif i =='Luke\n':
        k+=1
    elif i=='Leia\n':
        j+=1
p={'Darth':l,'Luke':k,'Leia':j}
print(p)
```

```
['Darth\n', 'Luke\n', 'Darth\n', 'Leia\n', 'Darth\n', 'Leia\n', 'Leia\n', 'Luke\n',
'Darth\n', 'Leia\n', 'Darth\n', 'Darth\n', 'Leia\n', 'Leia\n', 'Darth\n', 'Leia\n',
'Darth\n', 'Leia\n', 'Luke\n', 'Darth\n', 'Leia\n', 'Leia\n', 'Darth\n', 'Leia\n',
'Darth\n', 'Darth\n', 'Leia\n', 'Leia\n', 'Luke\n', 'Luke\n', 'Leia\n', 'Darth\n',
'Darth\n', 'Luke\n', 'Leia\n', 'Darth\n', 'Darth\n', 'Leia\n', 'Leia\n', 'Leia\n',
'Leia\n', 'Leia\n', 'Luke\n', 'Darth\n', 'Luke\n', 'Leia\n', 'Leia\n', 'Leia\n', 'Le
ia\n', 'Luke\n', 'Leia\n', 'Darth\n', 'Leia\n', 'Leia\n', 'Darth\n', 'Leia\n', 'Leia
\n', 'Darth\n', 'Darth\n', 'Leia\n', 'Darth\n', 'Leia\n', 'Darth\n', 'Luke\n', 'Leia
\n', 'Luke\n', 'Darth\n', 'Darth\n', 'Luke\n', 'Darth\n', 'Leia\n', 'Darth\n', 'Leia
\n', 'Luke\n', 'Leia\n', 'Leia\n', 'Leia\n', 'Leia\n', 'Leia\n', 'Darth\n', 'Leia
\n', 'Leia\n', 'Leia\n', 'Leia\n', 'Leia\n', 'Leia\n', 'Leia\n', 'Luke\n', 'Leia\n',
'Leia\n', 'Leia\n', 'Leia\n', 'Leia\n', 'Leia\n', 'Darth\n', 'Luke\n', 'Darth\n', 'L
eia\n', 'Leia\n', 'Darth']
{'Darth': 30, 'Luke': 15, 'Leia': 54}
```

Question 19:

Write a python program where two .txt files are given that have lists of numbers in them, find the numbers that are overlapping. One '**primenumbers1_1000.txt (https://github.com/milaan9/92_Python_Assignments/blob/main/primenumbers1_1000.txt)**' file has a list of all prime numbers under 1000, and the other '**happynumbers1_1000.txt (https://github.com/milaan9/92_Python_Assignments/blob/main/happynumbers1_1000.txt)**' file has a list of **happy numbers (https://en.wikipedia.org/wiki/Happy_number)** up to 1000.

Expected Output:

```
The list of overlapping numbers:
 [7, 13, 19, 23, 31, 79, 97, 103, 109, 139, 167, 193, 239, 263, 293, 313, 331, 367, 379, 3
83, 397, 409, 487, 563, 617, 653, 673, 683, 709, 739, 761, 863, 881, 907, 937]
```

For extra point:

1. Generate solution with functions using list comprehensions

In [12]:

```
# Solution 19:
f=open('primenumbers1_1000.txt','r')
t1=set(f.readlines())
f2=open('happynumbers1_1000.txt','r')
t2=set(f2.readlines())
jia=t1&t2
print(jia)
```

```
{'739\n', '109\n', '13\n', '331\n', '907\n', '239\n', '19\n', '313\n', '23\n', '79
\n', '683\n', '397\n', '167\n', '709\n', '367\n', '487\n', '673\n', '139\n', '31\n',
'937\n', '293\n', '563\n', '617\n', '761\n', '193\n', '7\n', '97\n', '383\n', '653
\n', '379\n', '263\n', '881\n', '103\n'}
```

Question 20:

Create a function that takes a string as an argument and returns the Morse code equivalent.

For example:

```
encode_morse("HELP ME !") ➞ ".... . .-.. .--.   -- .   -.-.--"
```

Expected Output:

```
Enter a sentence: I love
..   .-.. --- ...- .
Enter morse code: .--. -.-- - .... --- -.
PYTHON
```

This dictionary can be used for coding:

```
char_to_dots = {
    'A': '.-',    'B': '-...', 'C': '-.-.', 'D': '-..',  'E': '.',     'F': '..-.',
    'G': '--.',   'H': '....', 'I': '..',   'J': '.---', 'K': '-.-',   'L': '.-..',
    'M': '--',    'N': '-.',   'O': '---',  'P': '.--.', 'Q': '--.-',  'R': '.-.',
    'S': '...',   'T': '-',    'U': '..-',  'V': '...-', 'W': '.--',   'X': '-..-',
    'Y': '-.--', 'Z': '--..',

    '0': '-----', '1': '.----', '2': '..---', '3': '...--', '4': '....-',
    '5': '.....', '6': '-....', '7': '--...', '8': '---..', '9': '----.',

    ' ': ' ',       '&': '.-...',  "'": '.----.', '@': '.--.-.', ')': '-.--.-',
    '(': '-.--.',   ':': '---...', ',': '--..--', '=': '-...-',  '!': '-.-.--',
    '.': '.-.-.-', '-': '-....-', '+': '.-.-.',  '"': '.-..-.',  '?': '..--..',
    '/': '-..-.'
}
```

In [13]:

```
# Solution 20:
def f(s):
    t=''
    g ={
        'A': '.-', 'B': '-...', 'C': '-.-.', 'D': '-..', 'E': '.', 'F': '..-.',
        'G': '--.', 'H': '....', 'I': '..', 'J': '.---', 'K': '-.-', 'L': '.-..',
        'M': '--', 'N': '-.', 'O': '---', 'P': '.--.', 'Q': '--.-', 'R': '.-.',
        'S': '...', 'T': '-', 'U': '..-', 'V': '...-', 'W': '.--', 'X': '-..-',
        'Y': '-.--', 'Z': '--..',

        '0': '-----', '1': '.----', '2': '..---', '3': '...--', '4': '....-',
        '5': '.....', '6': '-....', '7': '--...', '8': '---..', '9': '----.',
        ' ': ' ', '&': '.-...', "'": '.----.', '@': '.--.-.', ')': '-.--.-',

        '(': '-.--.', ':': '---...', ',': '--..--', '=': '-...-', '!': '-.-.--',
        '.': '.-.-.-', '-': '-....-', '+': '.-.-.', '"': '.-..-.', '?': '..--..',
        '/': '-..-.'
    }
    for  i in s:
        i=i.upper()
        t+=g[i]
    return  t
s=input('输入一句话：')
gg=f(s)
print(gg)
```

```
输入一句话：Clark5405
-.-..-...-.-.-.-.........------.....
```

In [ ]: