Solutions for Programming Questions (15CS664 – Python) For 2nd Internals

1. **Write a program to count number of lines in a file and to display the line and total count.**

   Given in Notes on Files (In Section 2.3.4, page no. 3)

2. **Write a program to read *n* number of lines from the keyboard and to write them onto a file.**

```
fname=input('Enter a file name:')
n=int(input('Enter number of lines:'))

fhand=open(fname,'w')

for i in range(n):
     line=input("Enter a line: ")
     fhand.write(line+"\n")

fhand.close()
```

3. **Write a program to search and print all the lines starting with a specific word (taken as keyboard input) in a file.**

```
fname=input('Enter a file name:')
word=input('Enter a word to be searched:')

fhand=open(fname)

for line in fhand:
     if line.startswith(word):
          print(line)

fhand.close()
```

4. **Write a Python program to count number of characters, words and lines in a file**

```
file=input("Enter a file name:")
try:
    fhand=open(file)
except:
    print("File not found")
    exit(0)

lineCount=0
wordCount=0
charCount=0
```

```
for line in fhand:
    lineCount+=1
    charCount+=len(line)
    ls=line.split()
    wordCount+=len(ls)

print('Number of lines=',lineCount)
print('Number of words=', wordCount)
print('Number of characters=', charCount)
fhand.close()
```

5. Write a Python program print to first 10 lines and last 10 lines in a file.

```
file=input("Enter a file name:")
try:
    fhand=open(file)
except:
    print("File not found")
    exit(0)

s=fhand.read()    #s stores whole file contents
ls=s.split('\n')  #split s based on new-line character

print('First 10 lines are:')
for i in ls[:10]:
    print(i)

print('Last 10 lines are:')
for i in ls[-10:]:
    print(i)

fhand.close()
```

6. Write a python program to read a file and write into another file.

```
file=input("Enter a file name:")

try:
    fhand1=open(file)         #source file
except:
    print("File not found")
    exit(0)

fhand2=open('destFile.txt','w')
for line in fhand1:
    fhand2.write(line)

print('File copied successfully')
fhand1.close()
fhand2.close()
```

Note: After running above file, check your folder for `'destFile.txt',` which would have been a copy of your source file, given as input.

7. **Write a python program to create a list and print all the items in reverse index order.**

```
>>> ls=[10, 20, -3, 45, 23]
>>> print(ls[::-1])
```

8. **Write a Python function that takes two lists and returns True if they have at least one common member.**

```
def common(ls1, ls2):
    flag=False
    for i in ls1:
        for j in ls2:
            if i==j:
                flag= True
                break

    return flag

a = [1, 2, 5, 4, 15]
b = [10, 6, 7, 5, 9]
print(common(a, b))    #prints True

a = [1, 2, 3, 4, 5]
b = [6, 7, 8, 9]
print(common(a, b))     #print False
```

9. **Write a python function to check whether the list is Ascending order or not.**
   Same as Program 15

10. **Write a Python function that takes two sorted lists and return merged List.**

```
ls1=[34,1,56,102,12]
ls2=[8,34,123,-2,56,1,78]
ls1.sort()     #if ls1 is not sorted already
ls2.sort()     #if ls2 is not sorted already
ls3=sorted(ls1+ls2)
print(ls3)
```

11. **Write a Python program to remove duplicates from a list**

```
def Remove(ls):
    uniqueLs = []

      for num in ls:
        if num not in uniqueLs:
            uniqueLs.append(num)
    return uniqueLs
```

```
ls = [2, 4, 10, 20, 5, 2, 20, 4]
print(Remove(ls))
```

12. Write a Python program to check a list is empty or not.

```
def isEmpty(ls):
    if len(ls)==0:
        return True
    else:
        return False

print(isEmpty([]))
print(isEmpty([3,10,4]))
```

13. Write a Python program to print all the even indexed elements.

```
print(ls[::2])
```

14. A list rotation consists of taking the last element and moving it to the front. For instance, if we rotate the list [1,2,3,4,5], we get [5,1,2,3,4]. If we rotate it again, we get [4,5,1,2,3]. Write a Python function *rotatelist(ls,k)* that takes a list ls and a positive integer k and returns the list ls after k rotations. If k is not positive, your function should return ls unchanged. Note that your function should not change ls itself, and should return the rotated list. Here are some examples to show how your function shouldwork.

>>> rotatelist([1,2,3,4,5],1) #output is [5, 1, 2, 3, 4]
>>> rotatelist([1,2,3,4,5],3) #output is [3, 4, 5, 1, 2]
>>> rotatelist([1,2,3,4,5],12) #output is [4, 5, 1, 2, 3]

Program:

```
def rotatelist(ls,k):
    mylist=ls[:]
    for i in range(0,k):
        mylist=[mylist[-1]]+mylist[0:-1]
    return mylist
```

Note: The expression `mylist[-1]` gives single value (last value in the list). We need to concatenate it with `mylist[0:-1]` using + operator to rotate the list for once. But, + operator can concatenate two lists, but not a single value and a list. Hence, we need to use additional square-bracket to enclose that single value as: `[mylist[-1]]`

15. Define a Python function *ascending(ls)* that returns True if each element in its input list is at least as big as the one before it. For empty list, it should be True. Here are some examples to show how your function should work.

>>> ascending([]) #returns True
>>> ascending([3,3,4]) #returns True
>>> ascending([7,18,17,19]) #returns False

Program:

```
def ascending(ls):
    for i in range(len(ls)-1):
        if ls[i]<=ls[i+1]:
            continue
        else:
            return False

    return True
```

16. Define a Python function *alternating(ls)* that returns True if the values in the input list alternately go up and down (in a strict manner). For empty list, it should be True
For instance:

```
>>> alternating([])                    #True
>>> alternating([1,3,2,3,1,5])         #True
>>> alternating([3,2,3,1,5])      #True
>>> alternating([3,2,2,1,5])      #False
>>> alternating([3,2,1,3,5])      #False
```

17. Write a program to count frequency of each character in an input string, using dictionary.
Given in Notes (Section 3.2.1, Page No. 16)

18. Write a program to count frequency of words in a given file, using dictionaries. Ignore the punctuation marks attached to the words in file and treat lowercase and uppercase letters as the same.
Given in Notes (Section 3.2.4, Page No. 19-20)

19. Consider a dictionary with strings as keys and numbers as values. Write a program to sort the elements of this dictionary based on keys.
Given in Notes (Section 3.3.4, Page No. 29)
(To sort in ascending order, just use `ls.sort()`instead of `ls.sort(reverse=True)`

20. Read a string from keyboard input. Create a list containing tuples, where each tuple represents a word in the input string and length of that string. Write a program sort the words in descending order of their length.
Given in Notes (Section 3.3.1, Page No. 26-27)

21. Write a program to display most frequent 10 words in a text file. (Hint: use dictionary, list and tuple)
Given in Notes (Section 3.3.5, Page No. 29-30)

22. Write a python program to input a line of text name and find the frequency of each word.

```
s=input("Enter a string:")
d=dict()
```

```
for ch in s.split():
    d[ch]=d.get(ch,0)+1

print(d)
```

23. Write a python program to display value and key of a dictionary using a tuples

```
tel_dir={'Tom': 3491, 'Jerry':8135, 'Mickey':1253}
>>> for key, val in tel_dir.items():
     print(key,val)
```

24. Write a python program to read first name, last name and age, create a dictionary based on name as key and age as value.

Given in Notes (Section 3.3.6, Page No. 30)

25. Write a program to extract only email-ID's in a text file. Use suitable regular expression.

Given in Notes (Section 3.4.2, Page No. 36 – Refer Page 37 also for perfect RegEx!!)

26. Write a program to validate USN (both UG and PG) of VTU students. (Hint: UG USN format: 1RN15EC001, PG USN format: 1RN15MCA01)

```
import re

def val_USN(s):
   ug_pat='[1-4]{1}[a-zA-Z]{2}[0-9]{2}[a-zA-Z]{2}[0-9]{3}'
   pg_pat='[1-4]{1}[a-zA-Z]{2}[0-9]{2}[a-zA-Z]{3}[0-9]{2}'

   pat=ug_pat + '|' + pg_pat #combine two patterns using |(OR)

   if re.search(pat,s):
        print('valid')
   else:
        print('invalid')

val_USN('1rn15ec001')
val_USN('1abc034kx')
val_USN('1rn15mca01')
```

27. Write a program to validate a mobile number in India in the form of +91 99999 99999. (Note the white spaces after +91 and after first 5 digits)

```
import re

def ph_match(s):
    pat='^\+91 [0-9]{5} [0-9]{5}'
    if re.search(pat,s):
        print('valid phone number')
    else:
```

```
        print('invalid phone number')

ph_match('+91 94483 01894')        #valid
ph_match('83 94498 45823')         #invalid
```

28. Write a program to extract string from a file where the string should start with only uppercase letters and should end with a digit, and must contain a special character ^, anywhere in-between.

```
import re
fname=input("Enter file name:")
try:
    fhand=open(fname)
except:
    print("File cannot be opened")
    exit(0)

for line in fhand:
    pat='^[A-Z].+\^.+[0-9]$'
    line=line.rstrip()
    x=re.findall(pat,line)

    if len(x)>0:
        print(x)
```

29. Python program that matches a string that has an 'a' followed by anything, ending in 'b'

```
import re
s=input("Enter a string:")
x=re.findall('a.+b$', s)
print(x)
```

30. Python program that display lines that do not contain number in a text file.

```
import re
fname=input("Enter file name:")
try:
    fhand=open(fname)
except:
    print("File cannot be opened")
    exit(0)

for line in fhand:
    line=line.rstrip()

    if re.search('^([^0-9]*)$',line):
        print(line)
```