This book divide into partitions :-

1) to operate with normal programs as that appear in consol.
2) to operate with file.
3) to find output of programs.

1) to print string as triangle:

```
s="AliAhmed"
for a in range(len(s)):
   print s[0:a+1]
```

output:

A

Al

Ali

AliA

AliAh

AliAhm

AliAhme

AliAhmed

3) write program to calculate fibonancii

```
def fib(n):
    a,b=0,1
    for i in range(n):
        print(a)
        a,b=b,a+b
    return a

# note1 that a and b in this example execute in same time.
# note2 that you need to call this function to do.(no care if
you understand what I mind).
```

4) write program to convert first character to capital letter.

```
BLANK_CHAR=" "
def capitalize(s):
    d=""
    a=s.split()
    for element in a:
          s=element[0].upper()+element[1:]
          d=d+s+" "
    print d
capitalize(" i love to see allah always and ever")
```

5) write program to unlink nested list and print each element in single line.

```
def PrintNsestedLst(Lst):
   if len(Lst) == 0:
      return
   if not isinstance(Lst, list):
      print Lst
   else:
      for l in Lst:
         PrintNsestedLst(l)
PrintNsestedLst(['a','b','c',['d','e',['f','g']]])
```

6) write program to compute the permutation of list.

```
import math
def perm(list):
    tail=len(list)
    p=0
    q=1
    for counter in range(math.factorial(tail)):
        a=list[p]
        b=list[q]
        list[p]=b
        list[q]=a
        print(list)
        p=p+1
        q=q+1
        if(p==tail):
            p=0
        if(q==tail):
            q=0
perm([1,2,3]) # call the function here
```

7) write program that give person name and his age sequence.

```
index=1
name=['ali','ahmed','moahmed','badouin']
age=[10,20,30,40]
for a in age:
    name.insert(index,a)
    index=index+2
print(name)
```

8) write program to take every letter in middle word in string and print word from this characters.

```
a=""
list=["ali","ahmed","mohamed","ahmed"]
for counter in list:
    a=a+counter[len(counter)/2]
print(a)
```

9) write program that find indices for special character.

```
list=[]
counter=0
str=raw_input("enter string here \n")
search=raw_input("enter letter you search about it here\n")
for a in str:
   if a==search:
     list.append(counter)
   counter+=1
print(list)
# in the first give the string that containing the character
```

```
# in the second give the character you search about it here
```

10) write program that find gcd for two number.

```
def GCD(a,b):
     while(b!=0):
        (a,b)=(b,a%b)
     return a
print GCD(12,4)
```

```
another way to find gcd
import fractions
print fractions.gcd(12,3)
```

11) write program to find average of list.

```
def average(average_lst):
     avrg=0;count=0
     for element in average_lst:
          avrg=avrg+element
          count=count+1
     print avrg/count
average([1,2,3,4,5,6,7,8,9])
```

12) write a program that divide list into sub list you define the length of each list.

```
l=[1,2,3,4,5,6,7,8,9,10,11,12,13,14]
r,i=[],[]
limit=int(raw_input("enter numebr you want to seperate the list
about it\n"))
reminder=len(l)%limit
counter=0
for a in l:
    if counter==limit-1:
        i.append(a)
        r.append(i)
        i=[]
        counter=0
    else:
     i.append(a)
     counter+=1
if reminder>0:
    r.append(l[len(l)-reminder:])
print(r)
```

13) write program that made letter is key and count of repeat letter in string value.

```
a="abcdabcdabcdabcdabcdabcdddccbaabcdbacbd"
dictionary={}
for letter in a:
   if letter in dictionary:
       dictionary[letter]+=1
   else:
       dictionary[letter]=1
print(dictionary)
```

14) write program that shift right the element in the list as[1,2,3,4]>>become [4,1,2,3]

```
def shift_right(L):
   temp=L[len(L)-1]
   for a in range(len(L)-1,0,-1):
       L[a]=L[a-1]
   L[0]=temp
   print(L)
shift_right([1, 2, 3, 4, 5,6])
```

15) write program that shift left the element in the list as [1,2,3,4]>>>>[2,3,4,1]

```
def shift_left(L):
    temp=L[0]
    for a in range(len(L)-1):
        L[a]=L[a+1]
    L[-1]=temp
    print(L)
shift_left([1,2,3,4,5,6,7])
```

16) write program that remove space if increase about two space in the string

```
result = ""
str = "      ali    ahmed mohamed          no  this is     more"
temp=str[len(str)-1]
for a in range(0, len(str) - 1):
    if str[a] == " ":
        if str[a + 1] == " ":
            result = result + ""
        else:
            result=result+" "
    else:
        result = result + str[a]
print(result+""+temp)
```

17) write program that print all number primary for define number.

```
counter=0
number=int(raw_input("enter numer here\n"))
for first in range(2,number+1):
       c=0
       for second in range(2,first):
             if first%second==0:
                 c=c+1
       if c==0:
            counter+=1
            print(first)
print("the count of primary number is =%d",counter)

'''note that do you can instead of for second in range(2,first):
We write for second in range(2,9): and small the exe
instruction?????????
No because there number like than 91177 it not primary because
it divide about 73, 1249 and it don't divide about from 2 to 9
So that divide the number about 2->9 not correct'''.
```

---

18) write program that take from user numbers then find the average and summation and find all primary number that previous the max number user enter it.note that the program remain take numbers from user until user enter the word "done" then it out and calculate the previous operation if user enter any thing except number as "abc" or any thing program require try again and user enter number without program out.

```
def summation(L):
    sum=0
    for element in L:
        sum=sum+element
    return sum
```

```
def avrg(L):
    counter=0
    for element in L:
        counter+=1
    return summation(L)/counter
def All_primary(L):
    lst_primary=[]
    number=max(L)
    counter=0
    for first in range(2,number+1):
         c=0
         for second in range(2,first):
             if first%second==0:
                 c=c+1
         if c==0:
            counter+=1
            lst_primary.append(first)
    return lst_primary

l=[]
chk="dd"
while(chk!="d"):
     chk=(raw_input("no="))
     try:
         a=int(chk)
         l.append(a)
     except:
         print("try again.............")

sum=summation(l)
avg=avrg(l)
primary=All_primary(l)
print("the summation=",sum)
print("the average=",avg)
print("the primary number =",primary)
```

---

19) write program do the following:
You have dictionary that have key and value and you have list require from you:
Pass list on keys in dictionary if any element in list equal key in dictionary replace the value for key with the element in list

```
# Example list=[1,2,3,4]   dic={1:"ali",2:"ahmed",5:"shymaa"}
# The output is [ali,ahmed,3,4]

counter=-1
check_lst=[2,5,6,4]
dic={1:"ali",2:"ahmed",3:"mohamed",4:"badouin",5:"shymaa"}
for element in check_lst:
 counter+=1
 for key in dic:
     if element==key:
         check_lst[counter]=dic[key]
print check_lst
```

---

20) write program that return upperLetter in list and return lowerLetter in another list

```
list_contain_upper_lower_case=[]
upper=[]
lower=[]
message="ThisMessAgeFromMe TO InviteYOU"
for letter in message:
    if letter.isupper():
        upper.append(letter)
    elif letter.islower():
```

```
        lower.append(letter)
list_contain_upper_lower_case.append(upper)
list_contain_upper_lower_case.append(lower)
print list_contain_upper_lower_case
```

21) write program to calculate the upper lower words in each list in parent list

note that word consider upper if start with uppercase >>Ali and consider lower if it start with lowercase letter >>>ali

```
def count_letter_case(L):
    result=[]
    for lists in L:
         low=0
         up=0
         for item in lists:
              if item[0].islower():
                  low=low+1
              else:
                  up=up+1
         result.append((low,up))
    print(result)
count_letter_case([["ali","ahmed"],["hader","Ali","aLi"]])
```

---

22) write program that convert vowel (a,e,u,I,o) to special symbol as $ or @ …. Etc and other letter convert to underscore(_) and other staying of string remaining as it.

```
def hide_letters(secret,vowel_symbol):
    L=["a","e","u","i","o"]
    result=""
    for letter in secret:
         if letter.isalpha():
             if letter.lower() in L:
                 result+=vowel_symbol
             else:
                 result+="_"
         else:
             result+=letter
    return result
secret_text=hide_letters("ILove PythOn3.4","$")
print(secret_text)
hide_letters("gui in python complex","@")
```

---

23) write program that take integer number(it may be negative or positive) and print the sum from this number to 2*number.

```
def take(start,end):
   result=0
   for x in range(start,end+1):
         result+=x
   return result
sum_to_it=int(raw_input("enter number: "))
if sum_to_it>0:
    summation=take(start=sum_to_it,end=2*sum_to_it)
    print "sum:",summation
elif sum_to_it<0:
    summation=take(start=2*sum_to_it,end=sum_to_it)
    print "sum: ",summation
```

---

24) write program that uncompress compress string (i.e 2a3s1d convert to aassssd)

```
def uncompress(string):
    alph = []
    no = []
    uncompresse=""
    for alph_no in string:
        if alph_no.isalpha():
            alph.append(alph_no)
        elif alph_no.isdigit():
            no.append(alph_no)
    for index in range(len(no)):
        uncompresse+=alph[index]*int(no[index])
    print uncompresse
uncompress("2a5b4c1a")
# note thaht the output must be aabbbbbcccca.
# if may you (أمكنك)  to compress file try……it.
```

---

25) write program that count the number of only one letter in list.

```
def count_of_one_letter(L):
    result = 0
    for slice in L:
        for item in slice:
            if len(item) == 1:
                print(item)
                result += 1
    print("count of letter in list= ", result)
count_of_one_letter([["E", "FE", "ftp", "dhcp", "S"], ["C",
"stp", "O", "vtp"], ["D"]])
```

---

26) write program that give it message and return the ecryption message if may you to decrypt encrypted message try…….it.

```
import string
import math
def build_code(code):
    ua = string.ascii_uppercase  # ABCDEFGHIJKLMNOPQRSTUVWXYZ
    res = {}
    for letter in ua:
        res[letter] = ua[code]
        res[letter.lower()] = ua[code].lower()
        code += 1
        if code > 25:
            code = 0
    return res
def apply_coder(text, coder):
    dic = build_code(coder)
    ans = ""
    for letter in text:
        if letter in dic.keys():
            ans += dic[letter]
        else:
            ans += letter
```

```
    return ans
message_you_want_to_encrypt = input("enter your message here: ")
encrypt = apply_coder(message_you_want_to_encrypt, 4)
print("message after encrypt: ",encrypt)
decrypt = apply_coder(encrypt, -4)
print("message after decrypt: ",decrypt)
```

---

27) write program that compress uncompress string (i.e aassssd convert to 2a3s1d)

```
def compress(get_uncompress):
 uncompress=get_uncompress+";"
 counter=0
 end=0
 count_letter=""
 for element in range(len(uncompress)):
     if end <len(uncompress)-1:
         end+=1
         if uncompress[element]==uncompress[element+1]:
          counter+=1
          if end+1==len(uncompress):
              count_letter+=str(counter)+""+uncompress[element]
         else:
           counter+=1
           count_letter+=str(counter)+""+uncompress[element]
           counter=0
 print(count_letter)
```

---

3)operate with file

1) Create folder

```
import os
folder1=raw_input("enter folder1 name you want create it here")
folder2=raw_input("enter folder2 name you want create it here")
os.makedirs("E:\lectures\OperatingSystem\lectures\Lecture 4\%s"% folder1)
os.mkdir("E:\lectures\OperatingSystem\lectures\Lecture 4\%s"% folder2)

    # note1 that it take from user name of folder to create it in put it into
    variable folder1 and folder2.

    # note2 that I put path to create file in it so that the folder I will
    create it will create I the folder1 and folder2 in the folder that found in
    folder lectute 4.

    # what different between makedirs, makedir.

    # makedirs can create nested folder example about that you can create
    ===parent\child\grandson here will create 3 folder nested.

    # makedir create only one folder in once example = parent
    # note3 that os(operating system) this module is content about method that make you to operate
    with file like(create, delete, rename, ……).
```

2) Print number of line that user input it

```
  which=int(raw_input("no of line="))
  handle = open("E:\lectures\Dynamic Language\Lectures\Test\py.txt",
  'r')
  index = 1
  for a in handle:
      if index==which:
          print a.strip()
          break
      index = index + 1
```

note1 that int=>Integer to convert string into integer because raw_input take string from user and operate about this value as number we must convert it into Integer.
note2 that a is hold one line and it transfer to next line direct.
note3 that function calles STRIP() to delete indention.

3) Print multiline from file

```
def print_line(list):
    handle = open("E:\lectures\Dynamic
Language\Lectures\Test\py.txt", 'r')
    index = 1
    for a in handle:
        if index in list:
            print a.strip()
        index = index + 1


empty_lst = []
while (True):
    s = int(raw_input("lin_no="))
    if s<=0:
         print_line(empty_lst)
         break
    empty_lst.append(s)
```

4) Print each word in single line

```
handle = open("E:\lectures\Dynamic
Language\Lectures\Test\py.txt", 'r')
index = 1
for a in handle:
   words=a.split()
   for word in words:
       print word

# note1 that in function thsht cslled split it split line into
words and I do loop about this words so that it print each word
in single line.
```

5) require to input name of File to open it

```
ch = raw_input("enter name of file here\n")
file_path="E:\lectures\Dynamic Language\Lectures\Test\%s"%ch
handle=open(file_path,'r')
for a in handle:
    print a

# note1 that in some version of idle you use input insetead of
raw_input.
# note2 that your file you want to open it there in file called Test.
```

6) Write program to list filter files from special path.

```
import os
path=os.walk("E:\lectures")
for a,b,c in path:
  for sub in c:
    if sub.endswith(".m4a"):    # Filter only files extensions .m4a in that path
      print sub
```

7) Write program to find the largest line in file.

```
handle=open("E:\g\g.txt", 'r')
print max(handle,key=len)
```

8) Write program that merge two file in list

```
 # note that I want to take one line from file1 then first
line in
 # file2 then second line in file1 then second line in file2
an so on

index=1
merge_files=[]
```

```
file1=open("f1.txt","r")
file2=open("f2.txt","r")
for line_f1 in file1:
    merge_files.append(line_f1.strip())
for line_f2 in file2:
    merge_files.insert(index,line_f2.strip())
    index+=2
print(merge_files)
```

---

2) find output

Find output

```
x = 0
y = 1
a = cmp(x,y)
if a < x:
    print "a"
elif a == x:
    print "b"
else:
    print "c"
# note that the value here will be x-y=-1  so that a=-1
```

```
x = 1
y = "2"
z = 3

sum = 0
for i in (x,y,z):
    if isinstance(i, int):
        sum += i
print sum
# this will add int value only x,z so that the sum=4
```

```
def getinput():
    print "0: start"
    print "1: stop"
    print "2: reset"
    x = raw_input("selection: ")
    try:
        num = int(x)
        if num > 2 or num < 0:
            return None
        return num
    except:
        return None

num = getinput()
if not num:
    print "invalid"
else:
    print "valid"
# will return invalid
# note that here every thing will give false unless 0 will give
you true because not reverse
```

```
kvps = { '1' : 1, '2' : 2 }
theCopy = kvps
kvps['1'] = 5
print theCopy                # will give {'1': 5, '2': 2}
```

```
sum = kvps['1'] + theCopy['1']
print sum                       # 10
```

```
kvps = { '1' : 1, '2' : 2 }
theCopy = kvps.copy()   # i copy kvps in thecopy so that in
change in kvps don't effect in theCopy
kvps['1'] = 5
sum = kvps['1'] + theCopy['1']
print sum
```

```
aList = [1,2]
bList = [3,4]
kvps = { '1' : aList, '2' : bList }
print kvps
theCopy = kvps.copy()
kvps['1'][0] = 5
print kvps
print theCopy         # note that in list in state dic if you
copy the list afetr
sum = kvps['1'][0] + theCopy['1'][0]
print sum              # that you change in his values the
varibale that copy the
                       # list in it will changer he other
```

```
import copy
aList = [1,2]
bList = [3,4]
kvps = { '1' : aList, '2' : bList }
theCopy = copy.deepcopy(kvps)    # if change in the value not
effect about it
kvps['1'][0] = 5
sum = kvps['1'][0] + theCopy['1'][0]
print sum      # 6
```

```
def f(): pass
print type(f())                 #none type
print type(1J)                  #complex
print type(lambda:None)         # function
```

```
d = lambda p: p * 2
t = lambda p: p * 3
x = 2
x = d(x)
x = t(x)
x = d(x)
print x        # 24
```

```
x = 4.5
y = 2
print x//y           #2.0
print x/y           #2.25
```

```
nums = set([1,1,2,3,3,3,4])       # set function that delete
repeated number from the list
print nums
print len(nums)
```

```
print sorted([1,2,3,22,2,4,234324,231])   # sorted it take the
list direct in it.
```

```
L = [4, 10, 8]
x = L.sort()                    # x here is none
L.append(20)
L2 = L[:]
print L[:]
print id(L)
print id(L2)
```

```
def repeat_word(word, num_times):
    print(__name__)
    word = word * num_times
    print("Repeated word is:", word)
    return word
if __name__ =="__main__":
     word ="Yes"
     print("Original word is:", word)
     print("New word is:", word)
     word = repeat_word(word, 2) +"!"
     print("New word is:", word)
repeat_word("word", 3)
```

```
word = "computer"
print word[8]            #IndexError: string index out of range
```

```
tweet1 ="Computer science students organize UofT Hacks to create
new solutions"
tweet2 ="Meet Ross, the @IBMWatson-powered lawyer @IBM"
print -len(tweet2)
-45
print tweet2[-len(tweet2)]       # this will be print the numebr
in -45  M
```

```
total = 0
i = 0
while i <= n:
total = total + i
i = i + 2
Rewrite this code so that it uses a for loop instead of a while
loop.
total = 0
for i in range(0, n+1, 2): # alternate: for i in range(0, n+2,
2):
total = total + i
```

```
tweet1 ="#uoft_cs Turing award winner Steve Cook"
tweet2 ="Want cheap snacks? Visit @cssu office in BA2283"
print(tweet1[tweet2.find("W"):tweet2[-1]])    #TypeError
```

```
result = ""
list = ['123', '234', '345', '456', '567']
for First_character in list:
```

```
    result += First_character[0] * 2
print(result)
```

```
letters = ""
string = "asdf23f34vg45gb345"
for a in string:
    if a.isalpha():
        letters += a
print(letters)
```

```
letters = ""
string = "asdf23f 34vg45gb3  45"
for a in string:
    if a.isalpha():
        letters += "-"
    elif a.isdigit():
        letters += "#"
    else:
        letters += a
print(letters)
```

```
L = [["a", "b"], ["c", "d"]]
for item in L:
     item.append("0")
print L
```

```
s = "pizzapizza"
count = 0
i = 1
while i < len(s):
    if s[i] > "m":
        print(s[i])
        count = count + 1
    i = i + 1
print count
# the letters  of element that large from m letter
```

```
def alter(d):
    d["Dec"] = "vacation!"
months = {"Jan": "school"}
alter(months)
print months
# output
# {'Jan': 'school', 'Dec': 'vacation!'}
# because we add value in the dictionary not need to retuen any
value from the function
```

```
i=0
s="123"
for char_1 in s:
    for char_2 in s:
            for char_3 in s:
                print char_1 + char_2 + char_3
                i+=1
print(i)
#find all arrange of number
```

```
d = {1: "ali", 2: "ahmed", 3: "mohamed", 4: "badouin", 5: "FCI",
6: "ali"}
L = []
for k in d:
    if d[k] not in L:
        L.append(d[k])
        L.sort()
print(L)
# note that by default arrange the values according to key[-(1-
>9); 0->9; A->Z; a->z]
# sort function arrange the element as [-(1->9); 0->9; A->Z; a-
>z] this according to values
# if d[k] not in L:  this expression to prevent the repetation
in values
```

```
def myfunc(x, y, z, a):
    print x + y                          # will print x+y=1+2=3
nums = [1, 2, 3, 4]
myfunc(*nums)
# note that here * will convert list to sequence of number
```

```
def dostuff(param1, *param2):
   print type(param2)      # this type will be tuple

dostuff('apples', 'bananas', 'cherry', 'dates')
```

```
def hanoi(n, fro, to, spare):
    if n == 1:
        print str(fro) + " to " + str(to)
    else:
        hanoi(n - 1, fro, spare, to)
        hanoi(1, fro, to, spare)
        hanoi(n - 1, spare, to, fro)
hanoi(3, 1, 2, 3)
# not need to find out
```

```
a=12
b=13
print cmp(b,a)
 # this will find the different between b and a =1
 # will subtract the first value from the next value.
```

```
x = sum(range(5))
print x
 # note that range will print 0,1,2,3,4 so that x=10
```

```
kvps = { '1' : 1, '2' : 2 }
theCopy = dict(kvps)
# if value change don't effect about he copy
kvps['1'] = 5
sum = kvps['1'] + theCopy['1']
print sum                           #sum=6
```

```
print type(1/2)      # will print  int because 1/2 will give 0
```

```
a = [1,2,3,None,(),[],]
print len(a)
#6
```

```
D1 = {}
D1['a'] = 1
D1['b'] = 2
D1['a'] = 3
print(len(D1))
```

```
D2 = {1: 'hi', 2: 'bye'}
print(D2[-1])
# note that here the key -1 not found in the dictionary so that
it will
# give you error(KeyError: -1 is not a key in D2)
```

```
x = [1, 2]
L1 = [x, [8, 9]]
L2 = L1[:]
L2[0][1] = 999
print L1
print L2
```

---

print len("\n")

if there problem message me on
engaliahmedmohamed@yahoo.com
