华东师范大学计算机科学技术系作业

	华东师范大学计算机科学技术系作业	
课程名称：编程导论 Python	年级：2018级	作业成绩：
指导教师：杨燕	姓名：吴子靖	提交作业日期：2018年12月13日
专业：计算机系	学号：10185102141	作业编号： 9

一、用递归和非递归两种方式实现list的内置函数 pop ， 函数格式为 my_pop(L,i)。pop 函数将列表 L 中索引为 i 的元素弹出并返回其值。假设L=[1, 2, 3]，执行完 my_pop(1,2)后返回3以及新列表值[1, 2]。
（20分）

In [1]:

```
#非递归实现list的pop函数
def my_pop(L,i):
    if 0 <= i< len(L):
        return L[i],(L[0:i]+L[i+1:])
    else:
        print(" Index out of range")
L = [1,2,3];i = 2
my_pop(L,i)
```

Out[1]:

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

In [18]:

```
#递归实现list的pop函数
def my_pop_2(L,i):
    if i >= len(L) or i < 0:
        print("Index out of range")
    if i == 0:
        return L[0],L[1:]
    return 0+my_pop_2(L[1:],i-1)[0],[L[0]]+my_pop_2(L[1:],i-1)[1]
L = [1,2,3];i = 2
my_pop_2(L,i)
```

Out[18]:

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

二、 用递归和非递归两种方式实现 String 的 isalpha 函数。实现的函数格式为 my_isalpha(S)，my_isalpha 函数会判断字符串是否全部由字母组成，例如S="asd123"，在执行语句 my_isalpha(S) 后会返回False。
（20分）

In [8]:

```
#非递归实现 string的isalpha的函数
def my_isalpha(s):
    for e in s :
        if "a" <= e <= "z" or "A" <= e <= "Z":
            continue
        else:
            return False
    return True
s1="asd123";s2="AbGh"
print(my_isalpha(s1))
print(my_isalpha(s2))
```

```
False
True
```

In [9]:

```
#递归实现 string的isalpha的函数
def my_isalpha_2(s):
    if len(s) == 1:
        if "a" <= s[0] <= "z" or "A" <= s[0] <= "Z":
            return True
    return (my_isalpha_2(s[0]) and my_isalpha_2(s[1:]))
s1="asd123";s2="AbGh"
print(my_isalpha(s1))
print(my_isalpha(s2))
```

```
False
True
```

三、给定一个整数列表 L，输出一个列表 S，其中 S[i]=L[0] 到 L[i] 的和。例如 L=[1,2,4,2]，则输出 S=[1,3,7,9]。用简单递归方法实现此程序，再用二分递归方式来实现此程序。
（20分）

In [12]:

```
#简单递归
def s(L):
    if len(L) == 1:
        return [L[0]]
    return s(L[0:len(L)-1]) + [sum(L)]
L = [1,2,4,2]
print(s(L))
```

```
[1, 3, 7, 9]
```

In [37]:

```
#二分法递归
def s_2(L):
    if len(L) == 1:
        return [L[0]]
    L[len(L)//2] += sum(L[:len(L)//2])
    return s_2(L[:len(L)//2]) + s_2(L[len(L)//2:])
L = [1,2,4,2]
print(s_2(L))
```

```
[1, 3, 7, 9]
```

四、假设有4种钱币：1分钱、5分钱、10分钱和25分钱。请问100分钱有多少种不同钱币的组合方式？例如：5分钱有两种组合方式：5个1分钱和1个5分钱；10分钱有4种组合方式：10个1分钱，2个5分钱，5个1分钱加上1个5分钱，1个10分钱。请先写出递归关系式。
（20分）

In [15]:

```
#5分钱有两种组合方式，6分到9分的组合方式等价于5分的组合方式+1、2、3、4的组合方式，而这些不是5的倍数的面值只有一种组合方式，因此与5分的组合方式
#相同，所以对于一个不是5的倍数的正整数x，他的组合方式与（x//5）*5d的组合方式相等，因此现在只讨论5的倍数的组合方式
#先通过枚举找规律
#5分钱有1个5分，5个1分一共2张组合方式
#10分有10个1分钱，2个5分钱，5个1分钱加上1个5分钱，1个10分钱4种组合方式
#15分有15个1分，10个1分和1个5分，5个1分和2个5分，5个1分和1个10分，3个5分，1个5分和1个10分一共6种组合方式
#20分有20个1分，15个1分和1个5分，10个1分和2个5分，10个1分和1个10分，5个1分和3个5分，5个1分和1个10分和1个5分，4个5分，2个5分和1个10分，2个10分
#一共9种组合方式
#25分有25个1分，20个1分和1个5分，15个1分和2个5分，15个1分和1个10分，10个1分和3个5分，10个1分和1个5分和1个10分，5个1分和4个5分，
#5个1分和2个5分和1个10分，5个1分和2个10分，5个5分，3个5分和1个10分，1个5分和2个10分，1个25分，一共13种组合方式
#30分有30个1分,25个1分和1个5分,20个1分和2个5分，20个1分和1个10分，15个1分和3个5分，15个1分和1个5分和1个10分，10个1分和4个5分，10个1分和2个5分
#和1个10分，10个1分和2个10分，5个1分和5个5分，5个1分和3个5分和1个10分，5个1分和1个5分和2个10分，5个1分和1个25分，6个5分，4个5分和1个10分，2个
#5分和2个10分，1个5分和1个25分，3个10分一共18种方法
#不难发现 f(n)=f(n-5)+((n-5)//5)
def f(n):
    if n%5 != 0:
        n = (n//5)*5
    if n == 5:
        return 2
    if n == 10:
        return 4
    return f(n-5)+((n-5)//5)
print("100的组合方式为:",f(100))
```

```
100的组合方式为: 193
```

五、用递归方式将一个多层嵌套 list 展开成一层，以 list=['and','B',['not','A'],[1,2,1,[2,1],[1,1,[2,2,1]]],['not','A','A'],['or','A','B','A'],'B']为例，则 list 展开结果为['and','B','not','A',1,2,1,2,1,1,1,2,2,1,'not','A','A','or','A','B','A','B']。（20分）

In [48]:

```
def transform(L):
    if len(L)<2 and type(L[0]) != list:
        return [L[0]]
    elif len(L)<2 and type(L[0]) == list:
        return transform(L[0])
    if type(L[0]) != list:
        return ([L[0]] + transform(L[1:]))
    return transform(L[0]) + transform(L[1:])
L=['and','B',['not','A'],[1,2,1,[2,1],[1,1,[2,2,1]]],['not','A','A'],['or','A','B','A'],'B']
print(transform(L))
```

```
['and', 'B', 'not', 'A', 1, 2, 1, 2, 1, 1, 1, 2, 2, 1, 'not', 'A', 'A',
'or', 'A', 'B', 'A', 'B']
```