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：黎明	提交作业日期：2018年11月29日
专业：计算机系	学号：*	作业编号： 7

一、异常处理练习。假设输入一组任意长度列表，我们要对该列表中第10个元素进行加1操作，请利用try-except模型自己实现一个异常处理，可以捕获IndexError异常。
（15分）

In [1]:

```
def foo(lst):
    try:
        lst[10] += 1
    except IndexError as ie:
        print("Index error occurs.")
    except:
        print("Other error occurs.")

foo([])
```

```
Index error occurs.
```

二、 用二分法的递归方式求n个元素列表的最大值和最小值，改写本章的<程序：求数列最大最小值——二分法>，传递参数时不用分片，而是用它在原来列表的索引及长度。然后分析程序的开销，开销的增长趋势是什么？
（15分）

In [29]:

```
import random

global lst

def bMinMax(idx=0, length=len(lst)):
#     print(idx, length)
    if not length:
        return None, None
    elif length == 1:
        return lst[idx], lst[idx]
    elif length == 2:
        return min(lst[idx:idx+length]), max(lst[idx:idx+length])
    half = length//2
    front_half = bMinMax(idx, half)
    back_half = bMinMax(idx+half, length-half)
    return min(front_half[0], back_half[0]), max(front_half[1], back_half[1])

lst = [random.randint(0, 300) for i in range(10)]
print(lst)
print(bMinMax())
```

```
[55, 178, 129, 288, 105, 150, 217, 95, 200, 265]
(55, 288)
```

分析：增长趋势应该是$log_2 n$

三、用二分法的递归方式实现求给定数列L中所有元素的平均数。例如给定数列L=[12,32,45,78,22]，则该数列平均数为(12+32+45+78+22)/5=37.8。
（15分）

In [39]:

```
def avg(s,l,r):
    if l == r:
        return s[l]
    mid = (l+r)//2
    return (avg(s,l,mid)*(mid-l+1)+avg(s,mid+1,r)*(r-mid)) / (r-l+1)
l = [12,32,45,78,22]
avg(l,0,len(l)-1)
```

Out[39]:

```
37.8
```

四、用递归方法实现求给定正整数n的阶乘n!。例如n=3，则n的阶乘为$1 * 2 * 3 = 6$。
（15分）

In [2]:

```
def factorial(n):
        if n==1:
            return 1
        else:
            return n*factorial(n-1)
n=int(input("请输入一个正整数:"))
print("%d的阶乘是%d" %(n,factorial(n)))
```

```
请输入一个正整数:8
8的阶乘是40320
```

五、用递归方法实现求给定列表L中所有元素的最小值。如L=[11,15,9,14,8,5]，则最小数为5。
（15分）

In [2]:

```
def findmin(list):
    if len(list)==1:
        return list[0]
    return list[0] if list[0]<findmin(list[1:]) else findmin(list[1:])
print(findmin([9,3,5,2,3,7,4,3,0,3,3,3,4,2,2]))
```

```
0
```

In [2]:

```
def find_min(L):
    if len(L)==0:return []
    if len(L)==1:return L[0]
    min1=find_min(L[:len(L)//2])
    min2=find_min(L[len(L)//2:])
    if min1<min2:return min1
    else:return min2
L=[9,3,5,2,3,7,4,3,0,3,3,3,4,2,2]
find_min(L)
```

Out[2]:

```
0
```

六、调用import time库，编写一个程序能测试<程序：非递归实现my_remove>和它的较好程序的时间差异。建议所要删除的元素是一个长列表的最后一个。
（15分）

In [21]:

```
#非递归
def my_remove(L,x):
    A=L[:]
    for i in range(len(L)):
        if L[i]==x:
            A=A[:i]+A[i+1:]
            break
    return A
L=[1,2,3,4,5,6,8]
start=time.clock()
L=my_remove(L,8)
end=time.clock()
print(L)
print(end-start)
```

```
[1, 2, 3, 4, 5, 6]
5.688900000677677e-05

E:\Program\anaconda\lib\site-packages\ipykernel_launcher.py:9: DeprecationWarning: t
ime.clock has been deprecated in Python 3.3 and will be removed from Python 3.8: use
time.perf_counter or time.process_time instead
  if __name__ == '__main__':
E:\Program\anaconda\lib\site-packages\ipykernel_launcher.py:11: DeprecationWarning:
time.clock has been deprecated in Python 3.3 and will be removed from Python 3.8: us
e time.perf_counter or time.process_time instead
  # This is added back by InteractiveShellApp.init_path()
```

In [20]:

```
#递归
import time
def my_remove(L,x):
    if not x in L:
        return L
    A=[]
    for i in range(len(L)):
        if L[i]==x:
            A=A+L[i+1:]
            break
        A=A+[L[i]]
    return A
L=[1,2,3,4,5,6,8]
start=time.clock()
L=my_remove(L,8)
end=time.clock()
print(L)
print(end-start)
```

```
[1, 2, 3, 4, 5, 6]
7.054299999253999e-05

E:\Program\anaconda\lib\site-packages\ipykernel_launcher.py:13: DeprecationWarning:
time.clock has been deprecated in Python 3.3 and will be removed from Python 3.8: us
e time.perf_counter or time.process_time instead
  del sys.path[0]
E:\Program\anaconda\lib\site-packages\ipykernel_launcher.py:15: DeprecationWarning:
time.clock has been deprecated in Python 3.3 and will be removed from Python 3.8: us
e time.perf_counter or time.process_time instead
  from ipykernel import kernelapp as app
```

七、解释<程序：递归实现my_remove2使用二分法>的终止条件，为何要考虑长度为1的情形？（10分）

如果不考虑长度为1的情形，则当出现长度为1且不包含要删除元素的情况是，会导致递归无法结束。