第六章 序列：字符串、列表、元组-习题

6-1

in

6-2

```
#! usr/bin/env python

import string
import keyword

def idcheck():
    alphas = string.letters + '_'
    nums = string.digits
    alphasnums = alphas+nums

    print 'Welcome to the Identifier Checker v1.0\nTestees must be at least 2 chars long.'
    while True:
        myInput = raw_input('Identifier to test--please input an identifier:')
        if myInput in keyword.kwlist[:]:
            print '%s is a keyword' % myInput
            continue

        if myInput == 'q':
            break
        if len(myInput) > 0:
            if myInput[0] not in alphas:
                print '要以下划线或者字母开始'
            else:
                for otherChar in myInput[1:]:
                    if otherChar not in alphasnums: #alphas+nums,一般来说，从性能的的角度来考虑，把重复操作
作为参数放到循环里面进行是非常低效的.
                        print '其他字符必须是字母或者数字--invalid: remaining symbols must be alphanumeri
c'
                        break
```

```
                    else:
                        print 'okay as an identifier'
                        break
```

6-3

```
def sortNum(aNumber):
    numList = []
    for num in aNumber.split(','):
        numList.append(int(num))
    numList.sort(reverse=True)
    print numList
def sortDict(aDict):
    values = aDict.values()
    values.sort(reverse=True)
    print values

aNumber = raw_input("Enter a number split with ',':\n")
sortNum(aNumber)
aDict = {'a':8,'d':9,'e':1,'b':3,'c':12,'f':6,'g':7}
sortDict(aDict)
```

6-4

```
def average():
    gradeList = []
    listSum = 0

    while True:
        aScore = raw_input("Please Enter your point:\n")
        if aScore == "-1":
            break
        gradeList.append(float(aScore))
        listSum += float(aScore)
    print "the average Score is %.2f" % (listSum/len(gradeList))
```

6-8

```
units = ["", "one", "two", "three", "four",  "five",
    "six", "seven", "eight", "nine "]
teens = ["", "eleven", "twelve", "thirteen",  "fourteen",
    "fifteen", "sixteen", "seventeen", "eighteen", "nineteen"]
tens = ["", "ten", "twenty", "thirty", "forty",
    "fifty", "sixty", "seventy", "eighty", "ninety"]
thousands = ["","thousand", "million",  "billion",  "trillion",
    "quadrillion",  "quintillion",  "sextillion",  "septillion", "octillion",
    "nonillion",  "decillion",  "undecillion",  "duodecillion",  "tredecillion",
    "quattuordecillion",  "sexdecillion",  "septendecillion",  "octodecillion",
    "novemdecillion",  "vigintillion "]
def numToWords(num):
    words = []
    if num == 0:
        words.append("zero")
    else:
        numStr = "%d" % num
        numStrLen = len(numStr)
        groups = (numStrLen + 2) / 3
        numStr = numStr.zfill(groups * 3)
        for i in range(0, groups*3, 3):
            h = int(numStr[i])
            t = int(numStr[i+1])
            u = int(numStr[i+2])
            g = groups - (i / 3 + 1)

            if h >= 1:
                words.append(units[h])
                words.append("hundred")
```

```
            if t > 1:
                words.append(tens[t])
                if u >= 1:
                    words.append(units[u])
            elif t == 1:
                if u >= 1:
                    words.append(teens[u])
                else:
                    words.append(tens[t])
            else:
                if u >= 1:
                    words.append(units[u])

            if g >= 1 and (h + t + u) > 0:
                words.append(thousands[g])
        return words

print numToWords(1000000000)
print "-".join(numToWords(1001000025))
print "-".join(numToWords(1234567890))
print numToWords(0)
```

6-10

swapcase()函数

```
def findchr(string, char):
    if char not in string:
        return -1
    else:
        strLength = len(string)
        i = 0
        while i < strLength:
            if string[i:i+1] == char:
                return i
            i+=1

def rfindchr(aString,myChar):
    if myChar not in aString:
        print "-1"
    else:
        strLength = len(aString)
        i = 0
        resultList = []
        while i < strLength:
            if aString[i:i+1] == myChar:
                resultList.append(i)
            i += 1
        return resultList[-1]
        #print "last appear @ %d" % resultList[-1]

def subchr(aString, origchar, newchar):
    result = []
    if origchar not in aString:
        return -1
```

```
    else:
        strLength = len(aString)
        i = 0
        while i < strLength:
            if origchar == aString[i:i+1]:
                result.append(newchar)
            else:
                result.append(aString[i])
            i += 1
    return str(result)
```

6-12

```
def convertToDate(date):
    y1 = int(date.split("-")[0])
    m1 = int(date.split("-")[1])
    d1 = int(date.split("-")[-1])
    return (y1, m1, d1)

def diff(date1, date2):
    y1 = int(date1.split("-")[0])
    m1 = int(date1.split("-")[1])
    d1 = int(date1.split("-")[-1])
    dateone = datetime.date(y1, m1, d1)
    y2 = int(date2.split("-")[0])
    m2 = int(date2.split("-")[1])
    d2 = int(date2.split("-")[-1])
    datetwo = datetime.date(y2, m2, d2)
    return (dateone - datetwo).days

def daysFromBirthday(birthday):
    d1 = datetime.date(convertToDate(birthday)[0],convertToDate(birthday)[1],convertToDate(birthday)
[2])
    print (datetime.date.today() - d1).days
```