高三上信息专项练习——算法基础(作业 36)

1.有一排苹果箱，其中第 i 箱苹果的个数为 2^{i-1} 个，现要取 m 个苹果，求取走的箱子个数。实现该功能的程序段如下，横线处应填入的代码为（ B ）

```
m = int(input("请输入要取走的苹果个数:"))
t = 0
while m != 0:
    if _____:
        t += 1
    m//=2
print("箱子数为:", t)
```

进制换算
若m%2==1,表示取走该箱子

更正，缩进往外一步

A. m%2 == 0　　B. m%2 == 1　　C. m//2 == 0　　D. m//2 == 1

2.三类商品的销量分别为 a,b,c（单位：亿元）。现对销量进行降序排序，Python 代码如下：

```
if a < b:
    a, b = b, a
elif b < c:
    b, c = c, b
if a < c:
    a, c = c, a
print("销量依次为:", a, b, c)
```

任意输入 3 个数给 a,b,c，以下能检测出错误的测试数据为（ D ）

A. 2,3,1　　B. 3,2,1　　C. 3,1,2　　D. 1,3,2

3.提取数字字符串中以逗号分隔的数字并转换为整数存入数组，再将数组中的元素进行分类，第一类为“小于 30”；第二类为“30~60”；第三类为“大于 60”。例如输入数字字符串为“34,23,45,99,24,56,9,87,”，输出结果为 [23,9,24] [34,45,56] [87,99]。

（1）实现上述功能的 Python 程序段如下，请在划线处填入合适的代码。

```
s = input("请输入字符串(数字之间用逗号分隔):")
ch = ""; a = []
for i in range(len(s)):
    if s[i] != ",":
        ch += s[i]
    if s[i] == "," or i == len(s) - 1:
        a.append(int(ch))
        ①ch=""
i = 0; j = len(a) - 1; k = 0
while k <= j:
    if ②a[k]<30:
        a[i], a[k] = a[k], a[i]
        i += 1
    elif a[k] > 60:
        a[k], a[j] = a[j], a[k]
        j -= 1
        k-=1③
    k += 1
print(a[:i], a[i:j+1], a[j+1:])
```

i位置上的值是小于30的值，已经排好

将大于60的数往后换，后面j位上的值也排好，当前判断对象k位上的值是后面j位刚换过来的，还没判断，下一次就要轮到判断的，所以k=k-1搭配k=k+1，让k留在原位

i==len(s)-1，决定了最后一串数字没有逗号可能被存入a

区别在这里

（2）若两组输入的数据分别为“12,78,65,7,45,2,55,”、“12,78,65,7,45,2,55”，则两组输出的结果 A （单选，填字母：A. 一致；B. 不一致）。

4.七段数码管常用于显示数字，如图 a 所示。给每段数码管编号，通过点亮不同的段显示不同的数字。例如，数字 0 需要点亮 A,B,C,D,E,F 段，数字 1 需要点亮 B,C 段。

数码管经常发生故障：常亮和不亮。系统会根据运行日志判定数码管是否发生故障，如图 b 所示，日志由多行字符串组成，每行字符串中第一个字符为显示的数字，后为亮起的数码管编号。例如，“1BC”表示显示数字 1 时，B、C 段亮起。日志中可能有同一个数字的多次记录，但不会自相矛盾。

编写程序，用 7 个字符来标示数码管的检测结果。对于每一段，如果有证据表明它常亮，标记为“X”，若有证据表明它不亮，标记为“x”，若正常，标记为“0”。

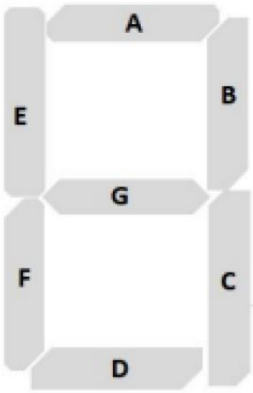


图 a

D常亮X
A不亮
缺A
缺A

图 b

（1）某段日志如图 b 所示，则检验结果为 A （单选，填字母）。

A. x00X000　　B. x00x000　　C. X00x000　　D. X00X000

（2）实现上述功能的 Python 程序如下，请在划线处填入合适的代码。

```
check = [[1,1,1,1,1,1,0],[0,1,1,0,0,0,0],[1,1,0,1,0,0,1,1],
         [1,1,1,1,0,0,1],[0,1,1,0,1,0,1],[1,0,1,1,1,0,1],
         [1,0,1,1,1,1,1],[1,1,1,0,0,0,0],[1,1,1,1,1,1,1],
         [1,1,1,1,1,0,1]]          # check 标记每个数字应点亮的段
ans = ['0'] * 7
f = open("log.txt")
line = f.readline().strip()
while line:
    t = ①int(line[0])
    L = len(line)
    a = [0] * 7
    for j in range(1, L):
        a[②ord(line[j])-65]=1
    for j in range(7):
        if ③a[j]==1 and check[t][j]==0:
            ans[j] == 'X':
        #elif 若数码管不亮，则标记为“x”,代码略
    line = f.readline().strip()
f.close()
for i in range(7):
    print(ans[i], end="")
```

读取行数据，并去掉头尾空格

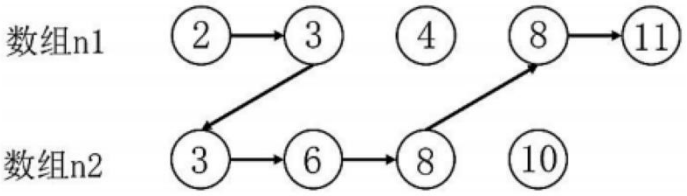
获取每行的第一位的数字，它决定了要亮哪些段的灯

桶

将A~G投进0~6号桶

不该亮的灯管亮了，就是常亮

5.有两个内部都已经升序排列的数组 n1 和 n2。现要找出一条得分最高的路径。选择两数组最小数作为起点，若遇到共同元素，可以从当前数组切换到另一个数组继续遍历（重复元素只计算一次得分）。现需找出所有可能路径中的最高总得分。（示例：最大得分为 30，该路径为 [2,3,6,8,11]）



1->3 5 7->9->11 13

3->6->7 8 11->15

（1）若 n1=[1,3,5,7,9,11,13]，n2=[3,6,7,8,11,15]，则可能路径中最高总得分为__52__。

（2）实现上述功能的部分 Python 程序如下，请在划线处填入合适的代码。

```
# 输入数组 n1、n2 的值，代码略
s1, s2, i, j = 0, 0, 0, 0
while i < len(n1) and j < len(n2):
    if n1[i] < n2[j]:
        s1 += n1[i]
        ① i+=1
    elif n1[i] > n2[j]:
        s2 += n2[j]
        j += 1
    else:  # 遇到共同元素
        if ② s1>s2 :
            s1 = s1 + n1[i]
            s2 = s1
        else:
            s2 = s2 + n2[j]
            s1 = s2
        i += 1
        j += 1
while ③ i<len(n1) or j<len(n2):
    if i < len(n1):
        s1 += n1[i]
        i += 1
    else:
        s2 += n2[j]
        j += 1
print("最大得分为", max(s1, s2))        # 函数 max 可求得所有参数中的最大值
```

一开始从最小数开始

遇到相等时可以切换到另一组，由于从最小值开始累加，所以小数的这组累加的比较多
s2=s1,说明把s1中累加好的值传递给s2，即从n1对切换到n2队

两队只要有一队没有遍历完，就需要将其遍历

6.回文字符串是指字符串从左向右读和从右向左读一样的字符串，如：“ABCCBA”“上海自来水来自海上” 等。“最长回文子串” 是指一个字符串中长度最大的回文子字符串，现需要编写 Python 程序输出 “最长回文子串” 的长度。

考虑到回文字符串有两种形式：①长度为奇数，则最中间的一个字符就是回文中心，如 “ABCBA”；②长度为偶数，则最中间的两个相同字符是回文中心，如 “ABCCBA”。因此我们可任选字符串中的一个字符或两个相邻且相同字符作为回文中心，同时向两边扩展出尽可能长的回文，而 “最长回文子串” 必定由其中一次扩展得到。

（1）若输入的字符串为 “ABCCBADDDCAABAACA”，则最长回文子串长度为_____7_____。

（2）函数 find () 返回字符串 s 中以 left 和 right 为回文中心扩展出的最长回文子串长度。left 和 right 是字符的索引值。请在划线处填入合适的代码。

```
def find(left, right):
    if left == right:  # 以一个字符为回文中心
        p = 1
    else:  # 以两个相同字符为回文中心
        p = 2
    while left > 0 and right < len(s) - 1 and s[left - 1] == s[right + 1]:
        left -= 1
        right += 1
        p+=2
    return p
```

两边值相同，回文字符长度增加

（3）实现上述功能的主程序如下，请在划线处填入合适的代码。

```
s = input("输入字符串:")
n = len(s)
maxlen = 0
left = 0
right = 0
for i in range(n):
    left, right = i, i
    m = find(left, right)
    if m > maxlen:
        maxlen = m
    left, right = i, i + 1
    if right < n and s[left]==s[right]:
        m = find(left, right)
        if m > maxlen:
            maxlen = m
print("最长回文子串长度为:", maxlen)
```

以一个值为中心向两边扩展

以两个值为中心向两边扩展

或s[i]==s[i+1]