复旦大学计算机科学技术学院

模拟试卷

课程名称：_____ 课程代码：_____

卷　　别：☐A 卷　☐B 卷　■C 卷

姓　　名：_____ 学　　号：_____

提示：请同学们秉持诚实守信宗旨，谨守考试纪律，摒弃考试作弊。学生如有违反学校考试纪律的行为，学校将按《复旦大学学生纪律处分条例》规定予以严肃处理。

（装订线内不要答题）

题号	一			二		三	四			总分
	1	2	3	1	2	1	1	2	3	
得分										

一、阅读程序写输出（30%）

第 1 题（10%）

```
test1 = [0, 1, 2, 3, 4]
test2 = [5, 6, 7, 8, 9]
test3 = test2
test3[::2] = test1[::2]
print(test1[:2], test2[:2], test3[:2])
```

第 2 题（10%）

```
m = [[j*4+i+1 for i in range(4)] for j in range(3)]
n = [i[::] for i in m]
for i in range(3):
    n[i][i] = 100*(i+1)
print(m[1][1]+n[2][2])
```

第 3 题（10%）

```
#某系统需要处理核酸抗原检测信息，因此使用了如下的正则表达式和数据格式
#来记录姓名(长度为 2-12)、学号、抗原检测结果、日期、电话。数据存在多行，
#用换行符分隔。
```

```
reg = r'(([0-9]{11}),([CT]),([0-9]{4}-[0-9]{2}-[0-9]{2}))'

data  = '托尼·斯达克,25307130001,C,2022-04-01,13812345678\n 史蒂夫·罗杰斯,25307130001,T,2022-04-01,13912345678\n 布鲁斯·班纳,25309990003,T,2022-04-01,14012345678...'
```

请写出使用如下代码时，数据结果

```
import re
match = re.search(reg,data)
print(match.group(1))
print(match.group(4))
print(match.group(3,4))
```

二、程序填空（20%）

注意:
(1) 每空限填一行代码。
(2) 除要求填空的位置之外，请勿改动程序中的其他内容。

第 1 题（10%）

请补充下面的程序使其自动生成 n 个互不相同的 100 以内的随机正整数。要求 n 是一个小于等于 10 的正整数。

```
import random

def generateList(n):
    lst = []
    if not (_____1_____ and 0<n and n <= 10):
        print('Parameter n is invalid.')
        return

    while True:
        if len(lst) == n:
            break
        r = random.randint(1,100)
        if r not in lst:
            lst.append(r)
        _____2_____

if __name__=='__main__':
    n = 5
    lst = generateList(n)
    if lst:
        print('Generate:', lst)
    else:
        print('Generate: Nothing')
```

请按照程序中的序号填充代码。注意每空只会包含一行代码。

序号	填空
1	
2	

第 2 题（10%）

程序要求实现文件的自动换行功能，即对于文本文件的那些过长的行进行拆分，每行每到指定的宽度之后自动换行，使得输出的结果中每行的长度最多为指定的宽度。

```
def word_wrap(pathname, textwidth):
    """ pathname 为文本文件名，textwidth 为指定的每行的最大宽度"""
    with open(pathname) as file:
        lines = file.readlines()
        ____1____:
            while len(line) > textwidth:
                ____2____
                print(first)
                line = line[textwidth:]
            print(line)

word_wrap('sample.txt', 80)
```

请按照程序中的序号填充代码。注意每空只会包含一行代码。

序号	填空
1	
2	

三、程序改错（10%）

神曲推荐。从音乐 APP 中获取了歌单，每首歌曲有时长和评论数量两个指标，向用户推荐评论数量较高、歌曲时长较短的歌曲。

以下程序对给定歌单（每行一首歌曲：歌曲名称、以 mm:ss 表示的时长、评论数量），计算每首歌曲的时长（换算为秒），先按照评论数量从大到小排序、再按照歌曲时长从小到大排序，最后输出推荐歌曲的曲名:

['There For you', 'Yummy', "I Don't Care", 'The Way I Am', 'Perfectly Wrong', 'South of the Boarder']

程序中两处有错误，请给出错误行的编号（每行最右边注释里的数字。如果代码行后面没有注释，表示该行没有错误）和改正后的整行代码。注意：改正后的代码只能是一行，不允许跨越多行。

```
def analyse(info):
    lines = info.split('\n')                                          # 1
    playlist = {}                                                     # 2
    for line in lines:                                                # 3
        name, time, num = line.strip().split()                        # 4
        playlist[name] = [time, int(num)]                             # 5
    return playlist                                                   # 6

def recommend(playlist):
    for song in playlist:                                             # 7
        min, sec = playlist[song][0].split(':')                       # 8
        playlist[song][0] = int(min) * 60 + int(sec)                  # 9
    return sorted(playlist.items(), key=lambda x:(x[1][1], x[1][0]))  #10

if __name__ == '__main__':
    info = '''I Don't Care,03:41,37382
    Perfectly Wrong,03:32,4184
    South of the Boarder,02:53,2728
    There For you,03:41,139173
    The Way I Am,03:06,17821
    Yummy,03:28,64764'''
    playlist = analyse(info)                                          # 11
    playlist = recommend(playlist)                                    # 12
    print([song[0] for song in playlist])                             # 13
```

四、编程（40%）

第 1 题（10%）

请编写函数，对一个只含有数字的字符串，统计偶数位置（从 0 开始）的数字与它后面相邻的奇数位置数字的最大差距。

例如数字字符串'03951421'中，0 号位置和 1 号位置数字分别是 0 和 3，差距为 3。同理后面各对数字里，9 和 5 差距为 4，1 和 4 差距为 3，2 和 1 差距为 1。所以函数统计出偶奇位置数字最大差距是 4，将其返回。

```
import random
import string
def maxdiff(s):
    # 请统计 s 中偶奇位置数字的最大差距
    # 请在下面注释 Program 和 End 之间添加代码，不得修改程序的其它部分。
    #**********Program**********




    #**********End**********
s = [random.choice(string.digits) for i in range(10)]
s = ''.join(s)
print('数字串%s 的偶奇位置数字最大差距为%d'%(s,maxdiff(s)))
```

第 2 题（10%）

请统计列表 intList 中各个数字出现的次数,并按整数从小到大按序将结果输出，程序运行的效果如下：

```
1．1出现次数为 3
2．2出现次数为 4
3．5出现次数为 3
4．6出现次数为 4
5．8出现次数为 4
6．9出现次数为 2
```

```
def main( ):
    intList = = [1,2,5,8,8,6,6,8,5,2,2,1,6,6,2,9,9,1,8,5]

    #请统计列表 intList 中各个数字出现的次数
    #并按整数从小到大的顺序将统计结果输出
    # 请在下面注释 Program 和 End 之间添加代码，不得修改程序的其它部分。
    #**********Program**********




    #**********End**********
    if __name__ == '__main__':
        main()
```

第 3 题（20%）

Kelly 教练的田径队里有 4 位最好的 U10 选手，他们每次跑 600m 所花费的时间被 Kelly 教练分别记录在 4 个文本文件中：James.txt，Sarah.txt，Julie.txt 和 Mikey.txt。这些时间的记录方式并不一致，有"分:秒"，"分.秒"，"分-秒"三种方式。Kelly 教练需要一种快速方法获得每位选手跑得最快的 3 个不同时间记录，请编程实现。

James.txt 的内容如下：

James Lee,2002-3-14,2-34,3:21,2.34,2.45,3.01,2:01,2:01,3:10,2-22,2-01,2.01,2:16

其他三人文件内容格式类似。

要求：

1) 编写函数 sanitize(time_string)，将 time_string 规范化表示为"分.秒"方式。

2) 编写函数 get_coach_data(filename)，获得 filename 文件中最快的 3 个不同时间记录。要求使用异常处理语句处理文件不能正常打开的情况。

3) 利用以上 2 个函数获得每位选手跑得最快的 3 个时间。

下图是文件都能正常打开时的程序运行情况：

James Lee's fastest times are: ['2.01', '2.16', '2.22']
Julie Jones's fastest times are: ['2.11', '2.23', '2.59']
Mikey McManus's fastest times are: ['2.22', '2.31', '2.38']
Sarah Sweeney's fastest times are: ['2.18', '2.21', '2.22']

下图是文件 james.txt 不能正常打开时的程序运行情况：

File error: [Errno 2] No such file or directory: 'james.txt'
Julie Jones's fastest times are: ['2.11', '2.23', '2.59']
Mikey McManus's fastest times are: ['2.22', '2.31', '2.38']
Sarah Sweeney's fastest times are: ['2.18', '2.21', '2.22']

注意：部分源程序给出如下。请勿改动主函数 main 和其它函数中的任何内容，
仅在函数的注释标志之间填入所编写的若干语句。

```
def sanitize(time_string):
    #**********Program**********




    #**********  End  **********

def get_coach_data(filename):
    #**********Program**********




    #**********  End  **********

def main():
    athletes = []
    athletes.append('james.txt'))
    athletes.append(get_coach_data('julie.txt'))
```

```
    athletes.append(get_coach_data('mikey.txt'))
    athletes.append(get_coach_data('sarah.txt'))

    for ath in athletes:
        print(ath['Name']+"'s fastest times are: "+ath['Times'])

if __name__ == '__main__':
    main()
```