1~5 ACCCD 6~10 DCCDB 11~12 AB

高三上信息选考练习卷18（寒假作业）

一、选择题（本大题共 12 小题，每小题 2 分，共 24 分。在每小题给出的四个选项中，只有一个特合题目要求，不选、多选、错选均不得分）

1. 下列关于数据、信息、知识和智慧的说法，正确的是 A

A. “结绳记事”方式中大小不一、数量不同的绳结是“数据”

B. 当人们孤立地看 3.14 时，这里的 3.14 是信息 数据

C. 面对同样的信息，不同的人理解可能不同，但最终建构的知识是相同的 不相同

知识

D. 智慧可由大量信息的积累、归纳、总结得出

2. 下列有关信息系统安全的说法，正确的是 C

A. 可以通过数据加密来检验数据的完整性

B. 加密后的数据无法传播计算机病毒

C. 防火墙既可以由软件和硬件组合而成，也可以只是软件系统

D. 对称密码体制和非对称密码体制的重要区别是加密和解密的算法是否一致

密钥

阅读下列材料，回答第 3 至 5 题：

某新式摄像头内置 AI 功能，基于海量数据训练，可以通过视觉识别技术初步判断车辆的危险行为和违规行为，比如打电话或者未系安全带，再将数据上传到服务器由交警进行审核，为构建城市交通大数据模型提供大量有参考意义的数据。

3. 下列关于该新式摄像头说法正确的是 C

A. 该摄像头通过强化学习方式实现车辆危险行为的判断 联结主义

B. 训练该 AI 功能的海量数据依赖于前期手工构造的知识库 X

C. 该摄像头应用于交通管理属于混合增强人工智能

D. 该摄像头在交通危险预警方面起到了很大的作用，可以完全替代人工判断 X

4. 下列关于该系统的说法，正确的是 C

A. 摄像头在信息系统中只充当传感器的作用 摄像头也可以充当执行器的作用，例如：当你需要查看不同角度时，可以控制摄像头旋转。

B. 该系统的用户只有交警和车辆驾驶员

C. 摄像头和服务器通信时需要网络设备、网络软件和网络协议支持

D. 如果需要提升摄像头的数据存储能力，应更换其处理器 存储器

5. 下列关于该系统中的数据，说法不正确的是 D

A. 可以通过增加训练数据提高该系统识别的准确性

B. 城市交通大数据的特点是处理速度快，数据体量大

C. 使用城市交通大数据进行实时导航主要利用流计算处理

D. 交通摄像头采集到的数据主要为结构化数据

6. 二维码是用某种特定的几何图形按一定规律在平面上分布的黑白相间的图形记录数据符号信息，下列关于二维码说法不正确的是 D

A. 二维码在计算机中也是采用二进制存储的

B. 一个由 24×24 黑白像素组成的二维码至少需要 72 字节的存储空间 24*24*1b/8=72B

C. 手机扫描商品标签上的二维码时会把模拟信号转换为数字信号

D. 手机扫描商品标签上的二维码后可以查看商品图片，故二维码能直接存储图像信息

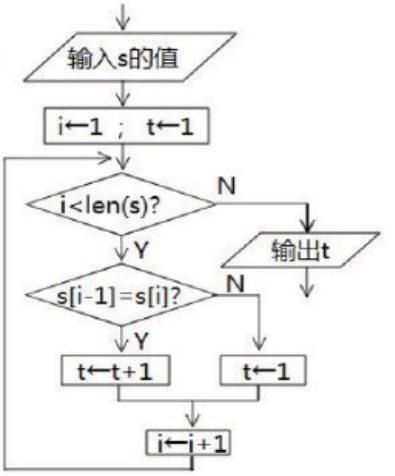
图像链接

t存放最后一次统计相同字符的个数

7. 某算法流程图如右图所示，若输出 t 为 3，则输入 s 的值可能是 C

A. "bbbaa" t=2　　B. "aabba" t=1

C. "bbaaa" t=3　　D. "abbab" t=1

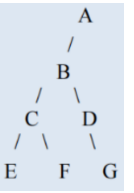


第 7 题图

8. 用一维数组表示二叉树，如下表所示；

0	1	2	3	4	5	6	7	8	9	10
A	B		C	D			E	F		G

下列有关该二叉树的说法正确的是 C



A. 该树中共有4 个叶子节点 3个

B. 该树是完全二叉树，其深度为 4

2^4-1-7

C. 该树的中序遍历为 E-C-F-B-D-G-A

D. 将此二叉树变成满二叉树还需增加4 个节点 8个

9. 用 python 列表模拟循环队列，并设置队首指针 head 指向队首元素，队尾指针 tail 指向队尾元素的下一个位置，则当列表长度 n=10，head=6，tail=3 时，队列中元素的个数为 D

10-6+3=7

(head-tail)%n

(6-3)%10

A. 3　　B. 5　　C. 6　　D. 7

10. 有如下两个 Python 程序段：

```
#程序段 1
def f(n):
    if n==1:            #代码（1）
        return 1        #代码（2）
    else:
        return n*f(n-1)
print(f(10))
```

递归

```
#程序段 2
def f(n) :
    ans=1
    for i in range(1,n+1):
        ans=ans*i
    return ans
print(f(10))
```

功能：求n!

迭代

因此其下标如下所示，head 有元素，tail 无元素，元素个数为 7 个。

0	1	2	3	4	5	6	7	8	9
			tail			head			
			无元素			有元素			

下列关于两个程序段的说法，正确的是 B

A. 程序段 1 和程序段 2 都采用了递归算法

B. 若问题规模为 n，程序段 1 和程序段 2 的时间复杂度都是 O(n)

C. 程序段 1 和程序段 2 的输出结果不相同

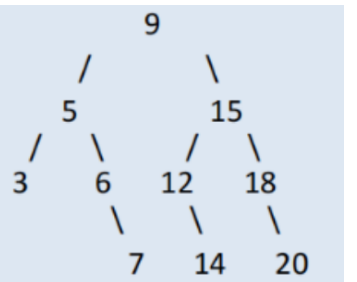
D. 程序段 1 中，代码（1）和代码（2）两处中的数字 1 同时修改为 0，输出结果不变 结果为0

11. 已知 a=[3, 5, 6, 7, 9, 12, 14, 15, 18, 20]，某二分查找算法的 Python 程序段如下：

```
key=int(input("请输入待查找的数："))
i=0;j=9;s=""
while i<=j:
    m=(i+ j) //2
    if key==a[m]:
        break
    if key<a[m]:
        j=m-1
        s=str(a[m])+" "+s
    else:
        i=m+1
        s=s+" "+str(a[m])
print(s)
```

如果下一次是向左找，倒序

如果下一次是向右找，正序



执行该程序段，输入待查找的数 key，则输出的内容不可能的是 A

key=7

9 5 6

key=2

key=13

key=21

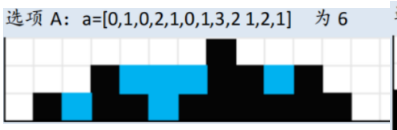
A. 6 5 9　　B. 3 5 9　　C. 14 15 9 12　　D. 9 15 18 20

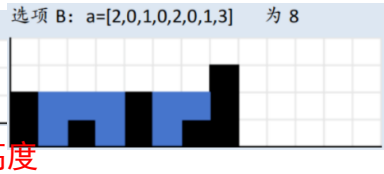
执行该程序段，输入待查找的数 key，则输出的内容不可能的是 A

key=7 key=2 key=13 key=21

A. 6 5 9 9 5 6　　B. 3 5 9　　C. 14 15 9 12　　D. 9 15 18 20

12. 在城市建设中，如果某处地形高度低于两侧的高度，则会形成洼地，并在降雨时积水。工程师每隔相同单位间距记录地形高度存储在数组 a 中，为分析该区间内地形数据以评估降雨后的积水情况，编写如下模拟程序：






```
#输入表示地形高度的数组 a，代码略
ans =0;n =len(a);st =[0] *n;top =-1
for i in range(n):
    while top>-1 and a[i]>a[st[top]]:
        right = st[top]
        top -= 1
        if top ==-1:
            break
        left = st[top]
        w =i - left -1
        h =min(a[left], a[i]) -a[right]
        ans +=w*h
    top += 1
    st[top] =i
print(ans)
```

当栈不空，并且当前柱子高度 > 栈顶柱子高度

可能形成了"凹槽"，可以接水

#right取出栈顶元素（凹槽的底部）

#弹出栈顶元素

#若栈空，无法形成凹陷区域，直接退出

#新栈顶元素是凹陷区域的左边界索引

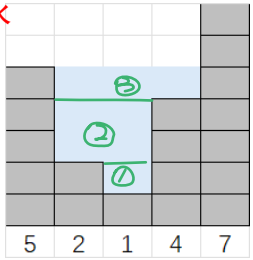
高度：左右边界的较小值 - 底部高度

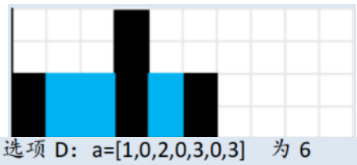
#累计积水量

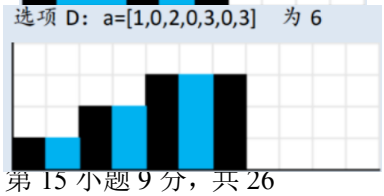
入栈的元素，
可以视为可能的凹槽的底部

算法：
形成单调递减栈

遇到i，高度> 栈顶
可能形成"凹槽"
栈顶出栈，为凹槽底部
i为凹槽右边界
当前栈顶为左边界
计算1次体积
....







已知程序执行后的输出值为 6，则输入的数组 a 不可能为 B

A. [0, 1, 0, 2, 1, 0, 1, 3, 2, 1, 2, 1]　　B. [2, 0, 1, 0, 2, 0, 1, 3]

C. [2, 0, 0, 4, 0, 2]　　D. [1, 0, 2, 0, 3, 0, 3]

二、非选择题（本大题共 3 小题，其中第 13 小题 7 分，第 14 小题 10 分，第 15 小题 9 分，共 26 分）

13. 小明为了研究生物实验室微生物培养皿中适宜的温度数据，通过从服务器数据库导出某培养皿中温度传感器采集到近 24 小时内全部的温度数据，查找当日温度连续不高于设定阈值（25℃）的最长时间段（若含有多段长度相同的最大值，输出最早的一个时间段）。

（1）若采集到的时间点和温度数据用数组 a 来表示，a[0][0]是第一个采集时间点，a[0][1]是第一个温度值，则[[1,21]，[2,24]，[3,25]，[4,26]，[5,22]，[6,23]]中符合要求的最长时间段是 ▲ B（单选，填字母：A.1 至 4/B.1 至 3/C.5 至 6）

（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。

```
#获取采集到的温度数据，用数组 a 存放，代码略
s=25;maxlen=0;n=len(a)
start=end=0;i=0
while i<n:
    if a[i][1]<=s:
        c=1    ①
        for j in range(i+1,n) :
            if  a[j][1]>s:
                break
        c=c+1
```

10 30 20 21 22 31

或a[j][1]>25

将 i 之后<=阈值的数据全部遍历

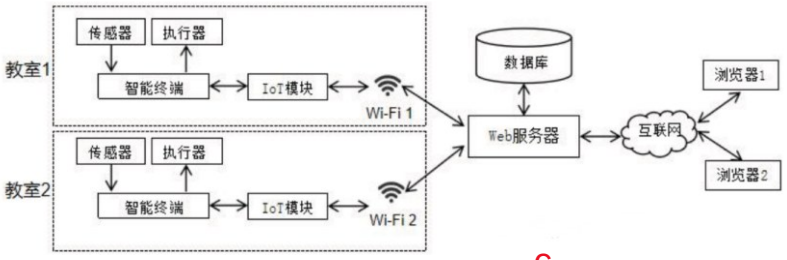
```
        if c>maxlen:
            maxlen=c
            start=i
            end=_____③_____
        i=i+c
    else:
        i=i+1
print("最长时间段为",a[start][0]，"至"，a[end][0])
```

i+maxlen-1 或i+c-1 或 start+c-1 或 start+maxlen-1

i跳到下一个位置

错误：j / j -1
j不一定指向>s的数据

14. 某技术学习兴趣小组搭建了早读分贝自动监测模拟系统，该系统结构示意图如第 14 题图 a 所示，每个教室里智能终端连接物联网模块、声音传感器和爆闪灯等设备，该系统能够实时监测每个教室的早读分贝，早读分贝高于一定阈值时爆闪灯自动亮灯，用户可以通过浏览器查看历史数据和当前爆闪灯状态。请回答下列问题：



第 14 题图 a

（1）下列关于该系统中数据传输说法正确的是_____▲_____（单选，填字母：A.只能由服务器传输至智能终端/B.只能由智能终端传输至服务器/C.服务器与智能终端之间相互传输）

C

（2）下列关于该监测系统的说法，正确的是_____▲_____（多选，填字母）（注：全部选对的得 2 分，选对不全的得 1 分，不选或有错的得 0 分）

ABD

A. 该系统采用了 B/S 的架构模式

B. 服务器可以通过向智能终端发送指令，控制执行器的开启或关闭

C. 若某一时刻物联网模块突然损坏，浏览器将无法查看传感器的历史分贝数据

D. 为有效保障数据资源安全，进入系统时要进行身份认证

不能看实时数据

（3）用 Flask 框架实现在网页中“显示分贝值”功能的部分代码如下：

服务端代码

```
@app.route("/show")
def voice():
    #代码略
app.run(host="192.168.0.2",port=8080)
```

则访问该页面的 URL 是 http://_____▲_____

192.168.0.2:8080/show

（4）系统工作一段时间后，发现浏览器无法查看某个教室的实时分贝数据，但能查看到其他教室的实时分贝数据，请简要说明系统中可能造成上述问题的两个原因（传感器故障不会引起上述问题）：①_____ ②_____

该教室的智能终端故障/该教室物联网模块故障

该教室智能终端与物联网模块的连接故障

IOT 与服务器..?

（5）年级组整理出近两个月的早读分贝数据，部分数据如第 14 题图 b 所示。现要统计 2 月份各班分贝值大于或等于 80 的天数，并绘制柱形图。

实现上述功能的部分 Python 程序如下：

```
#导入相关模块，代码略
df=pd.read_excel("data.xlsx")
df1=df[df.月份==2]
```

df2=_____①_____ C 在2月份数据的基础上筛选>阈值的数据

df3=_____②_____ D

plt.bar(df3["班级"],df3["分贝"])

#设置绘图参数，代码略

print(_____③_____ G) #由高到低输出各班早读分贝达标天数

在下列选项中选择①②③处对应的代码。

A. df[df["分贝"]>=80]　　B. df1.分贝>=80　　C. df1[df1.分贝>=80]

D. df2.groupby("班级",as_index=False).count() 计算次数

E. df2.groupby("班级",as_index=False).mean()

F. df3.sort_values("分贝",ascending=True)

G. df3.sort_values("分贝",ascending=False)

班级	月份	日期	分贝
1班	2	14	78
2班	2	14	76
3班	2	14	67
4班	2	14	71
5班	2	14	68
6班	2	14	71
7班	2	14	74
9班	3	28	77
10班	3	28	77
11班	3	28	80
12班	3	28	80
13班	3	28	79
14班	3	28	78

第 14 题图 b

15. 某音乐播放器中的“最近播放功能”是一种记录用户近期音乐播放历史的功能，该功能实现逻辑如下：（a）缓存容量限制 n（n>0）：最近播放列表中最多存储 n　首歌曲。（b）当播放列表中歌曲播放时：若已在缓存中，将其移动到最近播放列表最前端。若不在缓存中且缓存未满，直接加入最近播放列表最前端。若不在缓存中且缓存已满，删除最近播放列表末尾的歌曲，再将新歌曲加入最近播放列表最前端。

现给定用户的歌单列表 data，每首歌曲的信息包含歌曲编号、歌曲名称、歌曲推荐值（0-100 之间），data 中歌曲按照歌曲推荐值降序排序，编写程序模拟播放 data 歌单列表过程中实时输出最近播放记录。当 n=3，data　排序后数据如下表所示时，播放过程中的最近播放记录如下表最后一行显示。

歌曲编号	A1001	A1002	B2201	B2202	B2201	C1001	D1008
歌曲名称	《吻别》	《晴天》	《望》	《孤勇者》	《望》	《如愿》	《光亮》
歌曲推荐值	86	80	73	73	73	70	68
最近播放记录	《吻别》	《晴天》 《吻别》	《望》 《晴天》 《吻别》	《孤勇者》 《望》 《晴天》	《望》 《孤勇者》 《晴天》	《如愿》 《望》 《孤勇者》	《光亮》 《如愿》 《望》

请回答下列问题：

（1）对于上表中 data 数据，若 n 更改为 5 时，data 列表中歌曲播放完毕后，最近播放记录中最后一首歌曲名称为_____▲《晴天》

（2）定义如下函数 bubble_sort(data)，data 列表中每个元素包含三项，分别是歌曲编号、歌曲名称、歌曲推荐值，该函数实现对 data 列表按照歌曲推荐值降序排序。

```
def bubble_sort (data) :
    n = len(data)
    for i in range(n-l):
        flag= False
        for j in range(n -i -1):
            if _____▲_____ :
                data[j], data[j+1] = data[j+1], data[j]
                flag=True
        if not flag:
            break
    return data
```

data[j][2] < data[j+1][2]

歌曲编号	A1001	A1002	B2201	B2202	B2201	C1001	D1008
歌曲名称	《吻别》	《晴天》	《望》	《孤勇者》	《望》	《如愿》	《光亮》
歌曲推荐值	86	80	73	73	73	70	68
最近播放记录	《吻别》	《晴天》 《吻别》	《望》 《晴天》 《吻别》	《孤勇者》 《望》 《晴天》 《吻别》	《望》 《孤勇者》 《晴天》 《吻别》	《如愿》 《望》 《孤勇者》 《晴天》 《吻别》	《光亮》 《如愿》 《望》 《孤勇者》 《晴天》

①请在划线处填入正确的代码。

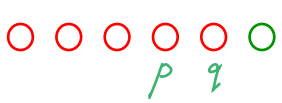
②加框处代码替换为下列哪个选项不影响函数功能__A__（单选，填字母）

A. n-2,i-1,-1　　B. 1,n-i　　C. n-l,i,-1　　D. i,n-i

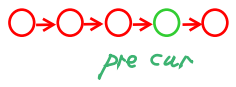
（3）实现模拟音乐播放时显示最近播放记录的部分 Python 程序如下，请在划线处填入合适的代码。

```
#函数功能为在链表 lst 中查找歌曲编号为 songid 的位置，head 为头节点，lst 中每个节点有三个区域，分别为歌曲编号、歌曲名称和指向下个节点的指针
def find song(lst,head,songid):
    pre=p=head
    while p!=-l:
        if lst [p][0]==songid:
            break
        pre=p
        p=lst [p][2]
    return pre,p
’’’
读取缓存容量限制 n，用户的播放数据存入 data 列表 data[0]包含 3 项，data[0][0]存放歌曲编号，data[0][1]存放歌曲名称，data[0][2]存放歌曲推荐值，代码略
’’’
lst=[] ; head=-1; length=0 #存储播放记录中歌曲数量
data=bubble_sort(data)
for song in data:
    pre,cur=find_song(lst,head, song[o])
    if cur==-l:    在原链表lst中找不到该歌曲
        lst.append([song[0],song[1],-1])#列表 lst 中末尾追加一个新的节点
        length+=1
        if length<=n:
            if length==l:    只有一首歌的情况
                head=0
        else:    歌曲数超过n首，删除最后一首
            p=q=head
            while lst[q][2]!=-1:    停止时，q是尾节点,p是q的前驱
                p=q
                q=lst[q][2]
            lst[p][2]=-1或    lst[p][2]=lst[q][2]
                              或 lst[p][2]=lst[lst[p][2]][2]    删q节点
        if length>l:
            lst[-1][2]=head    只要长度大于1首歌曲，都需要将新加入的歌曲调整到链头
            head=len(lst)-1
    else:    已经存在该歌曲,将该曲调整到链头
        if cur!=head:    cur节点调整到链头
            lst[pre][2]=lst[cur][2]    删除cur
            lst[cur][2]=head    cur添加到链头之前
            head=cur
    #从 lst 中 head 节点开始输出每个节点的歌曲名称，代码略
```

若n=5



H



最大问题：lst实际长度可能会>5，程序需要再优化