高三上信息选考练习卷5

一、选择题（本大题共 12 小题，每小题 2 分，共 24 分，每小题列出的四个备选项中只有一个是符合题目要求的，不选、错选、多选均不得分。）

阅读下列材料，回答第 1 至 6 题：

某小区的门禁系统支持业主人脸识别和刷卡进出。业主在识别仪前验证身份时，系统会记录开门方式（人脸或刷卡）、时间、人员身份等信息，拍照保存验证照片，并将数据存储在本地服务器中后，由服务器进行数据验证。

1.以下关于该系统中数据的说法，正确的是 D

A.该系统数据的表现形式均是图像

B.该系统数据只能存储在服务器中

C.系统记录的开门方式、时间和人员身份等信息均属于非结构化数据

D.系统数据中包含的人脸生物特征信息是敏感信息

2.以下措施对提升系统安全无效的是 C

A.定期修改门禁密码，并设置复杂的密码

B.对存储在本地服务器的数据进行加密

C.允许业主使用出身年月日为门禁密码

D.定期检查门禁系统的软件和硬件，及时更新系统补丁

3.下列关于该系统组成的说法，正确的是 B

A.该系统中的用户不包含业主

B.更换性能更好的服务器 CPU 可提升系统性能

C.记录业主信息的数据库是该系统的应用软件

D.刷卡机上不需要安装系统软件

4.以下关于该系统网络的说法，正确的是 C

A.业主刷脸时与服务器传输数据必须通过网关 在同一个局域网内不需要通过网关

B.门禁系统各个刷卡终端间必须使用有线网络

C.门禁系统向服务器发送数据时遵循 TCP/IP 协议

D.若该系统的服务器故障，业主可以通过刷卡进入社区

5.门禁系统记录的开门时间、方式和人员身份等信息需要高效存储和快速查询。以下关于该系统数据存储和编码的做法合理的是 B

A.为了节省存储空间，所有验证照片应以 bmp 格式存储 bmp未经压缩，比较大

B.可将开门时间中日期和时间转换为整数，便于计算和排序

C.人员身份信息以明文形式存储可便快速查询 不安全

D.系统数据仅使用本地存储设备，避免使用云端存储，就能防止数据泄露 不能

6.该系统引入人工智能技术对陌生人的自动识别和预警，以下说法正确的是 B

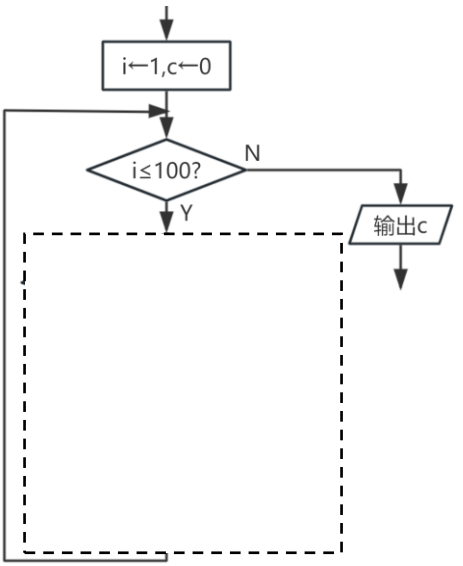
A.系统需要事先采集所有业主的人脸图像并手动标注 标注是联结主义数据训练的基础

B.系统通过深度学习算法提取人脸特征 投入使用后，不会称呼“标注”

C.增加模型数据训练量可以保证陌生人识别无差错

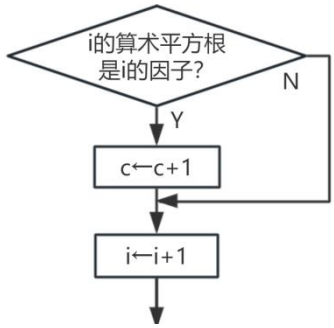
D.系统的陌生人识别功能需定期更新特征提取模型

7.统计 100 及以内的完全平方数个数的部分流程图如第 7 题图所示，以下选项中填入虚线框能实现该功能的是 A

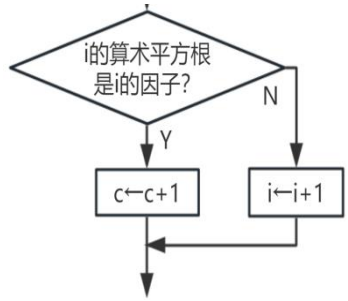


第 7 题图

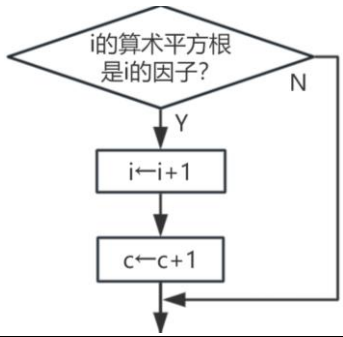
A.



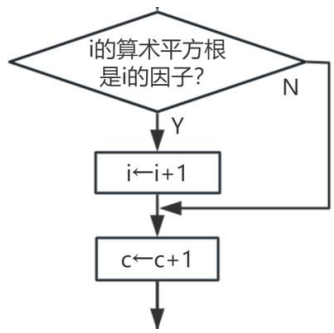
B.



C.

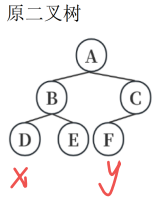


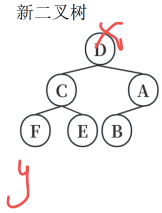
D.



8.一棵 6 个节点的完全二叉树，按前序遍历顺序将节点依次取出，取出后将节点逆序，再将节点按中序遍历顺序放回原二叉树节点中形成新树，原树中最左边的叶子 x 与最右边叶子 y，在新树中 x 是 y 的 C

A.孩子　　B.兄弟　　C.父亲的父亲　　D.孩子的孩子



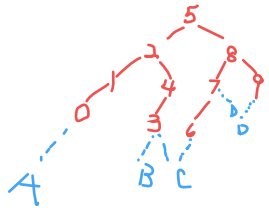


9. 某对分查找法的 Python 程序段如下：

```
i = 0; j = 9
while i <= j:
    m = (i+j+1)//2        #①
    if a[m] > key:
        j = m-1
    else:
        i = m+1
```

数组 a[0]到 a[9]中存放着各不相同且按升序排列的元素，若查找键 key 在以下范围时，则①处语句“m = (i+j+1)//2”被执行次数不同于其他项的是 D

A.key<a[0]　　B.a[2]<=key<=a[3]　　C.key=a[5]　　D.a[7]<=key<=a[8]



10.如下 Python 程序段，实现十进制数转十六进制数:

```
def cal(dec):
    chars = "0123456789ABCDEF"
    if dec == 0:
        return ""
    else:
        return [          ]
n = int(input("请输入待转换的十进制数:"))
print(cal(n))
```

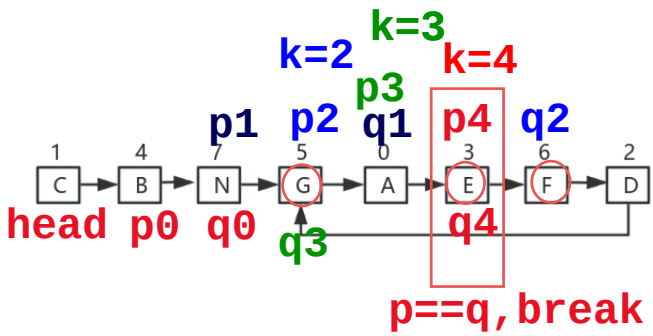
方框中应填入的正确代码为 **A**

A.cal(dec//16)+chars[dec%16]

B.chars[dec//16]+cal(dec%16)

C.cal(dec%16)+chars[dec//16]

D.chars[dec%16]+cal(dec//16)



11.有如下 Python 程序段：

```
from random import randint
k=randint(2,5)
lst = [["A",3],["C",4],["D",5],["E",6],["B",7],["G",0],["F",2],["N",5]]
head = 1; p = q = head; i=0
while  i <= k:
    p = lst[p][1]; q = lst[lst[q][1]][1]
    if p == q:
        break
    i += 1
```

运行后，lst[q][0]的值不可能是 **B**

k=3时，lst[q][0]='G'　　k=4时，lst[q][0]='E'　　k=2时，lst[q][0]='F'

A."G"　　B."A"　　C."E"　　D."F"

12.“气温跨度”定义为这一天之前连续多少天的最高气温低于或等于这一天的最高气温。

	周一	周二	周三	周四	周五	周六	周日
最高气温	10℃	17℃	13℃	11℃	11℃	15℃	16℃
气温跨度	1	2	1	1	2	4	5

实现该功能的 Python 程序段如下，方框中应填入的正确代码为 **D**

```
d = [10,17,13,11,11,15,16]          #存放每天的最高气温
k = [1 for i in range(len(d))]      #存放每天的气温跨度
s = [0 for i in range(len(d))]
top = -1
for i in range(1, len(d)):
    while top != -1 and d[s[top]] <= d[i]:
        top -= 1
    [          ]
print(k)
```

对于第i天（从0开始），找到第一个比它气温高的那天的下标prev，则跨度为 i - prev；
若前面所有天的气温都 ≤ 第i天，则跨度为i+1（因为包含自身，共i+1天）。

弹出栈中气温 ≤ 当前天i的下标（因为这些天的气温不影响 “第一个更高气温” 的判断）。
弹出后，需根据栈的状态计算k[i]，再将i入栈。

d：存储每天的最高气温；
k：存储每天的气温跨度（初始值为 1）；
s：单调栈（存储气温递减的天数下标）；
top：栈顶指针（top=-1表示栈空）。

弹出后分两种情况：栈空（top == -1）：前面所有天的气温都≤第i天，跨度为i+1）。
栈非空：栈顶d[s[top]] > d[i]，则跨度为 i - s[top]（如周六i=5，弹出后栈顶是i=1（气温 17），跨度5-1=4，与表格一致）。
再结合 “入栈操作”（计算后需将i压入栈）

top+=1
s[top]=i
if top==-1:
  k[i]=i+1
else:
  k[i]=i-s[top]
	if top==-1:
	  k[i]=i+s[top]
	else:
	  k[i]=i-1
	top+=1
	s[top]=i
		if top==-1:
		  k[i]=i-s[top]
		else:
		  k[i]=i+1
		top+=1
		s[top]=i
			if top==-1:
			  k[i]=i+1
			else:
			  k[i]=i-s[top]
			top+=1
			s[top]=i

A.	B.	C.	D.

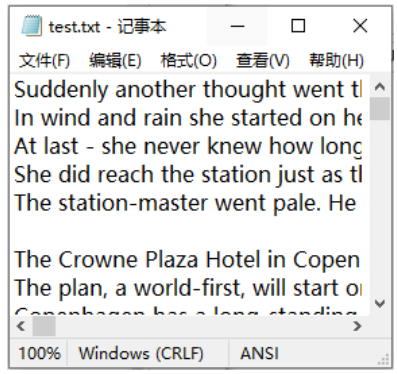
二、非选择题（本大题共 3 小题，第 13 小题 7 分，第 14 小题 10 分，第 15 小题 9 分，共 26 分。）

13. 小明整理了数篇高考英语阅读理解的文章，如第 13 题图所示，统计这些文章中词汇首字母出现的频率，最终输出出现频率最高首字母，若存在多个频率相同字母一并输出。

（1）若小写字母 a 到 f 为首字母的出现频率依次为 88、85、75、70、88、87，则最终选出的首字母为（填字母，逗号分隔）：▲ a，e

（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。

```
# 读取所有文章内容，均转为小写字母后，存储于变量 line 中
d = {}; i = 0; word = ""
while i <= len(line)-1:
    ch = line[i]
    if "a" <= ch <= "z":
        word += ch
    else:
        if len(word) > 0:
            if word not in d:
                d[word]=0          ①
            d[word] += 1
        word = ""
    i += 1
s = [0 for i in range(26)]
for i in d:
    s[ord(i[0])-97]+=d[i]          ②
m = 0; sign = []
for i in range(26):
    if s[i] > m:
        m = s[i]; sign = [chr(i+97)]
    elif s[i]==m                   ③ 或if s[i]==m
        sign.append(chr(i+97))
print("出现频率最高首字母是",sign)
```



第 13 题图

14. 某超市运用了“智能冷鲜系统”，该系统的智能终端能获取冷藏区内的温度数据，通过无线通信方式将采集到的温度数据实时传输至 Web 服务器。工作人员利用移动终端扫描商品条码，以此完成商品存放。服务器会根据货品存放情况，对临期商品进行降价处理并提醒。工作人员通过浏览器查询数据并管理。请回答下列问题。

（1）工作人员通过浏览器查询的数据存储在__C▲__（单选，填字母：A.智能终端 / B.移动终端 / C.数据库）

（2）下列关于该信息系统的说法，正确的有__AC▲__（多选，填字母）。（注：全部选对的得 2 分，选对但不全的得 1 分，不选或有选错的得 0 分）

A.监测结果可用图形化方式在浏览器界面上呈现

B.系统中的所有硬件设备需连接服务器后才能正常工作

C.若冷藏区无线通信故障，工作人员无法查看实时监测数据

D.传感器数据不需要经过服务器处理，直接推送给工作人员的移动终端

（3）服务器会根据商品库存情况，提醒选取了临期商品的顾客。以下适合的提醒方式是__C▲__（单选，填字母：A.冷藏区安装警铃 / B.向顾客发送短信 / C.顾客结账时提醒）。

（4）不同类别的冷鲜产品在储存时对温度要求有所不同，为了服务器对不同区域实现温控。请分别从软件和硬件的角度写出一种完整可行的办法

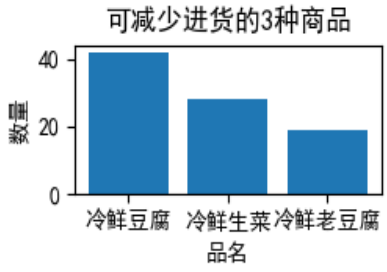
从硬件角度：为多个区域设置监测点，配备传感器及智能终端，为各监测点编号▲

从软件角度：在服务器为不同编号的设备设置不同的阈值▲

工作人员将系统中近一周（截止 2025 年 12 月 6 日）冷藏区货品库存数据导出，部分数据如第 14 题图 a 所示。现要对这些数据进行分析，请回答（5）、（6）两小题：

编号	品名	入柜日期	质保天数	单价	数量
8040105552	冷鲜大闸蟹	2025/12/1	1	70	6
8040105432	冷鲜豆腐	2025/12/1	2	4	21
8040105534	冷鲜老豆腐	2025/12/1	3	3	19
8040105441	冷鲜土层斑	2025/12/2	3	12	9
8040105471	冷鲜带鱼	2025/12/6	2	40	7
8040105582	冷鲜老豆腐	2025/12/6	2	3	19

第 14 题图 a



第 14 题图 b

（5）系统将质保天数的三分之一时间设为临期时限，离保质期只剩下临期时限的产品为临期产品，超过保质期为过期，其余为正常。实现该功能的 Python 程序段如下：

```
#导入相关模块，代码略
df = pd.read_excel("data.xls")
df['临期时限'] = df['质保天数'] / 3
today = datetime.today()     #获取当天日期
for i in df.index:
    x = (today - df.at[i,'入柜日期']).days        #计算入柜日期到当天的天数存入 x 中
    y = df.at[i,'质保天数']
    z = df.at[i,'临期时限']
    [          ]
    df.at[i,'状态'] = df_s
```

方框中可填入的正确代码有____▲ AD____（多选，填字母）。（注：全部选对的得 2 分，选对但不全的得 1 分，不选或有选错的得 0 分）

A.	df_s='正常'
	if y<x:
	df_s='过期'
	if y-z<x<=y:
	df_s='临期'
		B.	if y<x:
			df_s='过期'
			if y-z<x<=y:
			df_s='临期'
			else:
			df_s='正常'
				C.	if x>y:
					df_s='过期'
					elif x<z:
					df_s='正常'
					else:
					df_s='临期'
						D.	if x>y:
							df_s='过期'
							else:
							df_s='临期'
							if x<=y-z:
							df_s='正常'

（6）现从临期、过期商品中统计可减少进货的前 3 种商品，生成如第 14 题图 b 所示的柱形图。实现该功能的部分 Python 程序如下：

```
df1 = df          #备份数据以进行以下操作
[          ]
plt.bar(df2.品名,df2.数量)
#设置绘图参数，显示如第 14 题图 b 所示的柱形图，代码略
```

方框中应填入的语句依次为____▲ EDBF____（选 4 项，填数字序列，少选、多选、错选或次序错均不得分）

A. df1 = df1.sort_values('数量',ascending = True)　　#升序排序
B. df1 = df1.sort_values('数量',ascending = False)　　#降序排序
C. df1 = df1.groupby('状态',as_index = False).sum()　　#分组求和
D. df1 = df1.groupby('品名',as_index = False).sum()　　#分组求和
E. df1 = df1[df1['状态']!="正常"]
F. df2 = df1.head(3)
G. df2 = df1.tail(3)

15. 某数据序列 data 中的元素均为正整数。现在要对 data 进行处理，处理过程分“去重”“识别”和“合并”三步。对 data“去重”处理通过给定的 dedup 函数实现。“识别”指在一组数据中先识别出已升序的自然子序列，如 data 为[3,1,6,8,7,5]，可识别出 3 个，分别是[3,6,8],[1,7],[5]。在“合并”过程中，算法会优先合并那些长度较短的子序列，最终得到一个完全有序的完整序列。

（1）若 data 为[5,1,6,3,8,7,4,9]，则可依次识别出的自然子序列为：____3 ▲____个。

（2）“去重”处理的 dedup 函数如下，请在划线处填入合适的代码。

```
def dedup(a):
    i = 0; n = len(a)
    while i < n:
        r = i + 1
        for j in range(i + 1, n):
            if  a[i] != a[j] :
                a[r] = a[j]; r += 1
        n = r; i += 1
    return a[:n]
```

此处函数作用为去重，由 n=r可知，r为不重复元素的索引

（3）实现处理功能的部分 Python 程序如下，请在划线处填入合适的代码。

```
def sort_data(data):
    k = 1
    while k < len(data):
        tmp = data[k]; j = k - 1
        while j>=0 and data[j][2]>tmp[2]:  ①
            data[j+1] = data[j]
            j -= 1
        data[j+1] = tmp
        k += 1
def merge(i):
    qa = pa = ha = runs[i][0]
    qb = pb = hb = runs[i+1][0]
    while pb != -1:
        while pa!=-1 and nodes[pa][1] < nodes[pb][1]:
            qa = pa; pa = nodes[pa][2]
        qb = pb; pb = nodes[pb][2]
        nodes[qb][2] = pa
        if pa == ha:
            ha = hb; qa = qb
        else:
            nodes[qa][2] = qb
            qa=nodes[qa][2]  ②
        if ha!=-1:
            end = runs[i][1]
        else:
            end = runs[i+1][1]
    return [ha,end,runs[i][2]+runs[i+1][2]]
# 读取数据存入 data,进行去重处理，n 为去重后 data 数据个数，代码略
nodes = [[0,data[0],-1]]
runs = [[0,0,1]]
for i in range(1,n):
    k=0
    while k<len(runs) and data[i]<nodes[runs[k][1]][1]:
        k+=1
    if k<len(runs):
        nodes[runs[k][1]][2]=i  ③
        runs[k][1]=i; runs[k][2]+=1
    else:
        runs.append([i,i,1])
    nodes.append([i,data[i],-1])
i = 0
while i<len(runs)-1:
    sort_data(runs)
    runs.append(merge(i))
    i += 2
    # 输出处理后的 data 序列，代码略
```

sort_data(data)的作用是将 runs 中的内容进行插入排序，故①处答案为 while j >= 0 and data[j][2] > tmp[2]

merge(i)函数的作用是nodes 中的数据按runs升序结果进行链表合并操作，将b链表中的头节点插入到 a链表中，故 nodes[qa][2] = qb，且qa 需要指向下一个节点，故②处答案为qa = nodes[qa][2]

runs=[链头，链尾，个数]

找到data[i]可以加入的链队

链队已经存在

新开一个链队

链队代码

主程序中，需要将 data中的数据分别存成链表 nodes，nodes[i][0]为data[i]的索引位置，nodes[i][1]为data[i]的值，nodes[i][2] 为该节点的指针，runs中每个元素分别存放自然子序列的头索引，尾索引，以及元素个数，故先找到需要链上去的 k，runs[k][1]尾索引指向i，更新nodes中runs[i][1]对应节点的指针，故 ③处答案为 nodes[runs[k][1]][2]=i。