2024 年 11 月绍兴市选考科目诊断性考试

技术参考答案　第2页开始为试卷解析

第一部分 信息技术（共 50 分）

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，只有一个符合题目要求）

1	2	3	4	5	6	7	8	9	10	11	12
C	A	D	C	D	A	C	B	B	B	C	A

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

13. （1）0　（1 分）

（2）①pre = d[t]　（2 分）

②x % 3 - 1 >= 0 或 x % 3 != 0 或 x % 3 > 0 及等价表达式　（2 分）

③abs(d[i] - q[i] % 2)或(d[i] + q[i]) % 2

或 d[i] ^ (q[i] % 2)及等价表达式　（2 分）

14. （1）AE　（2 分）

（2）17　（1 分）

（3）/res　（1 分）

（4）参考答案：　（2 分）

①人体感应模块替代声音传感器，感应到人体热辐射信号时，灯才点亮。

②声音识别模块替代声音传感器，识别到汽车喇叭声时，灯不点亮。

（5）①"日期"　（1 分）

②　dfh["亮度"] <= 110　（2 分）

15. （1）"abb$aa"　（1 分）

（2）①B　（1 分）

②j = b[i] – 1　（2 分）

③不影响，因为比较字符串时，“$”比 data 中所有字符都要小　（2 分）

（3）①x = a[0]　（2 分）

②shift[i] = b[ord(tran_data[a[i]])].pop(0)　（2 分）

2024 年 11 月绍兴市选考科目诊断性考试

1	2	3	4	5	6	7	8	9	10	11	12
C	A	D	C	D	A	C	B	B	B	C	A

第一部分 信息技术（共 50 分）

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、多选、错选均不得分）

阅读下列材料，请回答 1～2 题：

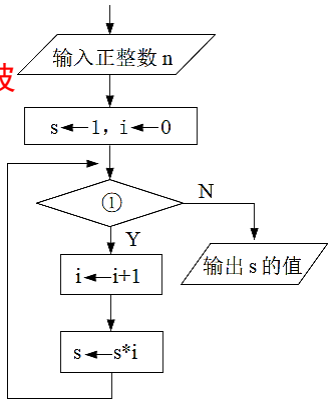
“数字农业大脑”实时监测养殖动物的健康状况，并记录相关数据。由于动物在进食、睡觉和生病等不同状态下被监测到的声音和体温不同，因此，“数字农业大脑”可通过分析监测数据，判断其是否出现异常情况。

1. “数字农业大脑”处理相关监测数据时，做法合理的是 C
   A. 降低单位时间内的监测次数以提升数据的质量
   B. 对监测数据采用单一的呈现形式
   C. 将监测数据分类整理后进行存储
   D. 根据某个特定的监测数据来分析动物的健康状况
2. 下列关于“数字农业大脑”的优势与局限性，说法合理的是A
   A. 可以有效提高工作效率　　B. 不会存在任何安全隐患
   C. 不会引发数字鸿沟问题　　D. 可以完全替代工作人员

阅读下列材料，请回答 3～5 题：

智能门锁系统通过人脸识别和射频识别技术实现无钥匙开锁功能。用户可以刷脸或使用带有射频标签的设备（如手机、手环）解锁，还可以通过手机等设备实现自动化控制，让门锁与灯具、空调等设备联动，并可以登录账号查看使用数据。

3. 以下关于智能门锁系统的说法，不正确的是 D
   A. 系统运行需要传感与控制技术的支持
   B. 射频识别技术传送数据时需要传输介质rfid读卡时：传输介质：无线电波
   C. 门锁与其他设备联动体现了物联功能
   D. 该系统的所有数据都存储在服务器中
4. 为提高智能门锁的人脸识别效果，以下策略不合理的是 C
   A. 提高硬件设备性能　　B. 优化图像特征提取效率
   C. 减少训练数据的规模　　D. 迭代更新相关算法模型
5. 为防止系统敏感信息泄露，下列措施合理的是 D
   A. 对信息采用明文存储　　B. 登陆系统使用初始口令
   C. 系统设置统一登陆账号　　D. 为系统设置防火墙



第 7 题图

6. 某未经压缩的无声影片帧频为 30fps(帧/秒)，帧图像存储容量为 24MB，则其 1 分钟视频的数据量约为A　若有声视频，+声音存储容量　24MB*30*60/1024 约等于 A选项
   A. 42. 2GB　　B. 20. 9GB　　C. 1440MB　　D. 720MB
7. 如第 7 题图所示的流程图描述了“计算 s=1*2*3*…*n”的算法，则图中①处应填入的内容是 C
   A. i＜n-1　　B. i≠n-1　　C. i≠n　　D. i≤n

暂认为优先级 越大则优先发送

8. 某设备按优先级 1 至 4 依次发送数据，利用队列 S 组织数据并模拟发送过程，初始队首至队尾数据依次为：S1、S4、S3、S2，各数据优先级如第 8 题图所示。可利用操作 M（出队后再入队）调整数据发送顺序，为将数据发送完毕，M 的执行次数至少为 B

数据	优先级
S1	3
S2	4
S3	1
S4	2

第 8 题图

A. 4　　B. 5　　C. 6　　D. 7

阅读下列材料，回答第 9 至 10 题：

马尔科夫链在人工智能中有许多的应用实例。如某竞猜活动，每局猜中与否的概率均为 50%，猜中赢得 1 积分，猜不中输掉 1 积分。初始积分为 A，竞猜持续进行，直到积分为 0 或达到预期积分 B（A、B 为正整数，B>A）时停止。此过程可以通过马尔科夫链分析，前两轮结果如第 9 题图所示。

代码中条件为B>=A

9. 数组元素 d[0]至 d[6]保存了如第 9 题图所示二叉树的中序遍历序列，有如下程序段：

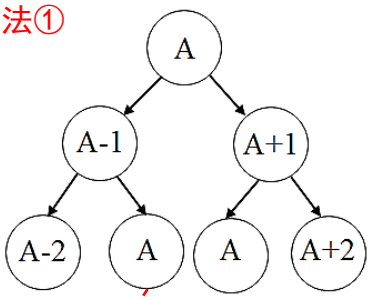
```
i, j = 0, 6
key = A
while i <= j:
    m = (i + j) // 2
    if key <= d[m]:
        j = m - 1
    else:
        i = m + 1
print(i, j)
```

方法2

[A-2,A-1,A,A,A,A+1,A+2]

j i

方法①



第 9 题图

该程序段执行结束后，输出结果为 B

A. 1　0　　B. 2　1　　C.　3　2　　D. 4　3

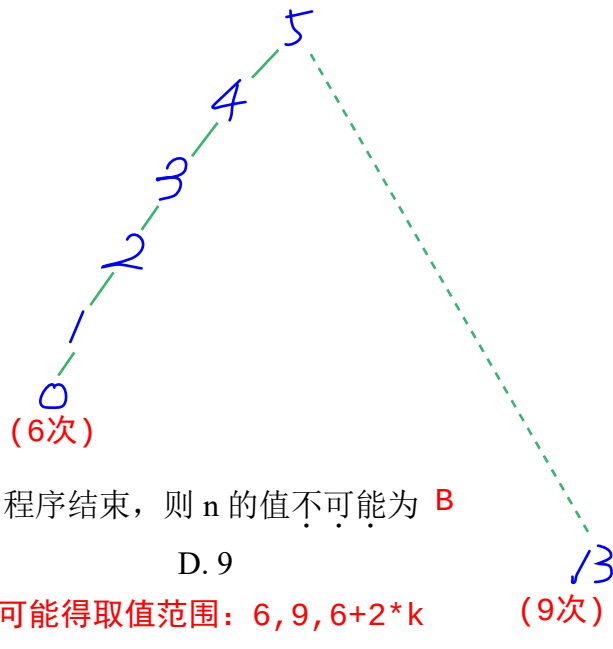
赌徒 常见释义

英 ['gæmblə(r)]

10. 模拟竞猜过程的 gambler 函数定义如下：

```
from random import randint
def gambler(A, B):
    if A == 0:
        return "lose"
    if A >= B:
        return "win"
    p = [-1, 1]
    i = randint(0, 1)      # 随机取整数 0 或 1
    return gambler(A + p[i], B)
```

int((randint(0,1)-0.5)*2)



执行语句 v = gambler (5, 13)，函数 gambler 被调用 n 次后程序结束，则 n 的值不可能为 B

A. 6　　B. 7　　C. 8　　D. 9

X

可能得取值范围：6,9,6+2*k

11. 有如下程序段：

```
p = 0
for i in range(len(a)):
    if not ("a" <= a[i] <= "z"):
        j = i; pt = a[i]
        while j > p:
            a[j] = a[j - 1]
            j = j - 1
        p = j; a[j] = pt
print(a[4])
```

将所有非小写字母字符向前移动，到所有小写字母以前，且小写字母相对位置不发生改变。

运行结果 a=['-','-','-','p','y','t','h','o','n']

a[4] ='y'

若列表 a=["-","p","y","-","t","h","o","n","-"]，则运行该程序段后，输出的值为 C

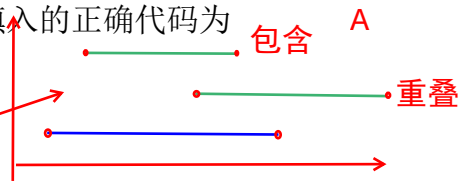
A. "-"　　B. "p"　　C. "y"　　D. "t"

```
            j = j - 1
        p = j; a[j] = pt
print(a[4])
```

若列表 a=["-","p","y","-","t","h","o","n","-"]，则运行该程序段后，输出的值为

A. "-"　　B. "p"　　C. "y"　　D. "t"

12. 数组 lst 的元素[a, b]，表示一个整数序列区间(a, b 为整数，且 a≤b)，如[2,4]表示整数 2,3,4。合并数组 lst 中所有的重叠区间，输出合并结果。如[[1, 5], [9, 15], [6, 10], [20, 25], [12, 18]]可以合并为[[1, 18], [20, 25]]。实现该功能的程序段如下，方框中应填入的正确代码为



1.lst中区间可能乱序
2.两个区间存在重叠，才需要合并
3.重叠的情况

```
i = 0; j = len(lst) - 1
while i <= j:
    p = i + 1
    while p <= j:
        if lst[i][0] <= lst[p][0]:
            [          ]
        else:
            lst[i],lst[p] = lst[p],lst[i]
    i = i + 1
print(lst[0:i])
```

[[1, 5], [9, 15], [6, 10], [20, 25], [12, 18]]

i　p　j

p区间起始位置小于i区间起始位置，互换，类似冒泡
没有这句，就可能死循环

A.
```
if lst[i][1] + 1 >= lst[p][0]:
    lst[i][1] = max(lst[p][1], lst[i][1])
    lst[p] = lst[j]
    j = j - 1
    p = i + 1
else:
    p = p + 1
```

存在重叠或包含
p已并到i中，p要删除
方法：将最后区间(j)覆盖到p
p重新从i+1开始扫描

B.
```
if lst[i][1] + 1 >= lst[p][0]:
    lst[i][1] = max(lst[p][1], lst[i][1])
    lst[p] = lst[j]
    j = j - 1
    i = p + 1
else:
    p = p + 1
```

C.
```
if lst[i][1] + 1 >= lst[p][0]:
    lst[i][1] = max(lst[p][1], lst[i][1])
    lst[p] = lst[j]
    j = j - 1
else:
    p = p + 1
```

D.
```
if lst[i][1] + 1 >= lst[p][0]:
    lst[i][1] = max(lst[p][1], lst[i][1])
    lst[p] = lst[j]
    j = j - 1
i = p + 1
p = p + 1
```

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

13. 编号为 0~8 的开关组成一个 3 行 3 列的开关阵列，如第 13 题图所示。每个开关状态为“开”(1)或“关”(0)，每操作某个开关 1 次，将导致自身和所有相邻的开关改变状态。例如，操作开关 4 将导致开关 1、3、4、5、7 改变状态。

指定一组操作对象序列 P，序列元素 P_i 为开关编号。对 P 中开关依次各执行 1 次操作后，求开关阵列的状态，并判断其中某个开关 S 的状态是否变化。请回答问题：

（1）若 P 为[0,2,5,6,7]，执行操作后，开关阵列中状态发生变化的开关编号是__▲__（填数字）。

（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。

```
while True:
    #操作对象序列、开关阵列状态数据存入 P、d，d[0]~d[8]表示开关 0~8 的状态，代码略
    t = int(input("请输入开关 S 的编号："))
    pre=d[t]
    q = [0] * 9
    for x in P:
        q[x] += 1
        if x%3!=0或x%3>0或x%3>=1:
            q[x - 1] += 1
        if x - 3 >= 0:                    #上
            q[x - 3] += 1
        if x % 3 < 2:                     #右
            q[x + 1] += 1
        if x + 3 <= 8:                    #下
            q[x + 3] += 1
    for i in range(9):                    #更新开关阵列的状态
        d[i] = (d[i]+q[i]) % 2
    if pre == d[t]:
        print("开关 S 的状态未改变")
    else:
        print("开关 S 的状态改变")
```

0	1	2
3	4	5
6	7	8

第 13 题图

14. 为减少能源浪费，提高系统巡检和故障定位效率，科创小组拟在实验室搭建“智慧路灯”模拟系统。该系统中含有 15 盏路灯，每盏路灯配备一个智能终端，连接 5 个传感器。智能终端获取传感器数据，并通过无线通信将数据传输至 Web 服务器。请回答下列问题。

（1）科创小组整理了如下功能需求，这些功能中必须在智能终端实现的是 ▲AE（多选，填字母）。（注：全部选对的得 2 分，选对但不全的得 1 分，不选或有错的得 0 分）

A. 支持北斗定位，能上传故障点位置 智能终端 上传 服务器

B. 提供警报功能，在路灯发生故障时可短信通知检修人员 服务器

C. 对所有路灯实施分时段的不同控制策略 服务器

D. 通过 Web 平台可视化监控和调取路灯信息 服务器

E. 实时采集路灯工作状态和环境温湿度、光亮度数据✓

F. 支持历史数据统计分析和查询，构建校园路灯画像

（2）为方便数据传输，该系统采用的传感器二进制编码方案是：路灯 5 位编号+传感器 3 位编号，如 01001011。若后期系统扩容需要，至多可增加 17 （填数字）盏路灯。

17=2**5-15（已有数量）

（3）该系统服务器端程序采用 Flask Web 框架编写，部分网页设置代码如下：

```
@app. route("/")
def data():
    #查看历史数据，过程略
@app. route("/res",methods=['POST', 'GET'])
def app_res():
    #接收上传数据，过程略
```

①人体感应模块替代声音传感器，感应到人体热辐射信号时，灯才点亮
②语音识别模块替代声音传感器，识别到汽车喇叭声时，灯不点亮。
将智能终端（服务器）触发开灯的声音阈值设置为合理范围，避免汽车喇叭错误触发

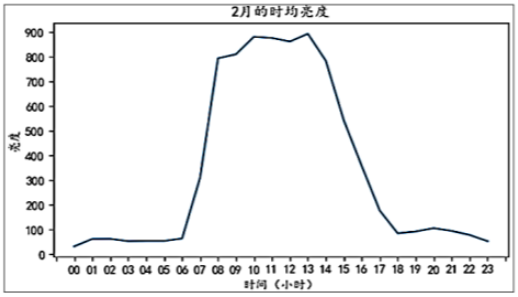
则上传数据时访问的页面路由为 /res 。

（4）在黑暗环境中，系统使用声控方式控制路灯是否点亮。实验发现，当室外有汽车喇叭声时，灯也可能点亮。请完善该设计方案（要求：文字描述不超过 35 字）。

（5）科创小组整理出该校的一年历史采集数据，部分数据如第 14 题图 a 所示。现要统计各月中每小时的平均亮度的分布情况并绘制线形图，例如 2 月的时均亮度线形图如第 14 题图 b 所示。

日期	时间	温度	湿度	亮度
2023-2-28	00:59:16	19.9884	37.0933	45.08
2023-2-28	01:03:16	19.3024	38.4629	45.08
2023-2-28	01:06:16	19.1652	38.8039	45.08
2023-3-6	09:33:08	24.4278	34.3273	250.24
2023-3-6	09:33:33	24.467	34.2577	250.24
2023-3-6	09:34:03	24.5062	34.188	250.24

第 14 题图 a



第 14 题图 b

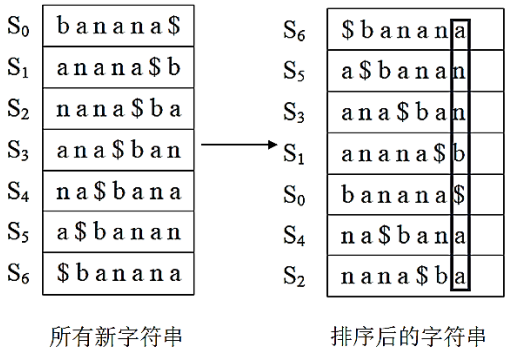
科创小组发现，时均亮度不高于 110 的“小时”应开灯，为了更好地为每个月选择合适的开启路灯时长，编写的部分 Python 程序如下，请在划线处补充代码。

```
import pandas as pd
import matplotlib. pyplot as plt
dft = pd. read_csv("data. csv")   # 读取文件 data. csv 中的数据
# 为 dft 插入"小时"、"月份"列，代码略
for i in dft. index:
    t = dft. at[i, "时间"]          # 通过行标签和列标签选取单个值
    dft. at[i, "小时"] = t[0:2]
    m = dft. at[i, '日期']
    p = m. split("-")[1]           # 获取月份数据
    dft. at[i, "月份"] = int(p)
for i in range(12):
    dfm = dft[dft["月份"] == i + 1]
    dfh = dfm. groupby("小时", as_index=False). mean()   # 分组求均值
    dfn = dfh[    ②    ]["小时"]. count()                  # 筛选求时长
    plt. plot(dfh["小时"], dfh["亮度"])                    # 绘制线形图
    #设置绘图参数，显示如图第 14 题图 b 所示的线形图，代码略
```

dfh.亮度<=110 或 dfh['dfh.亮度<=110']<=110

15. 某英文字母序列 data 的加密、解密过程如下：

“加密”过程为：先将 data 加上后缀“$”（“$”比 data 中所有字符都要小)得到字符串 S_0，取出 S_0 首字符放到 S_0 末尾得到新字符串 S_1，重复上述操作，一轮结束后得到所有新字符串；对这些字符串进行升序排序后，依次取它们的末尾字符组合得到加密结果。例如对“banana”加密，结果为“annb$aa”，具体过程如第 15 题图 a 所示。



第 15 题图 a

“解密”则通过还原加密结果得到 data。例如，先将“annb$aa”还原为“banana$”，再去掉“$”得到“banana”。编写程序实现上述功能，程序中用到的列表函数与方法如第 15 题图 b 所示：

函数与方法	功能
w. append(x)	在列表 w 末尾添加元素 x
x = w. pop(0)	在列表 w 的首元素赋值给 x，并将其从 w 中删除

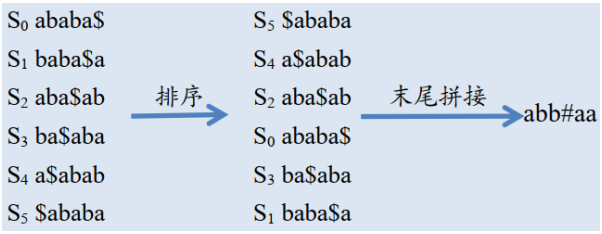
第 15 题图 b

请回答下列问题：

（1）若 data 为“ababa”，经过“加密”处理后结果为 abb$aa ▲ 。

（2）“加密”处理的相关函数如下：

```
def compute (data):
    a = []; b = []
    n = len(data)
    for i in range(n):
        a. append( [i, data[i:n] + data[0:i]] )
    # 对 a 数组中的所有字符串升序排序，代码略
    for i in a:
        b. append(i[0])
    return b
def trans(data):
    data = data + "$"
    b = compute(data)
    n = len(b); s = ""
    for i in range(n):
        j=b[i]-1
        if j < 0:
            j = j + n
        s += data[j]
    return s
```



[索引， 字符]

[0 , "ababa$"]

[索引-1]

在 compute 函数中应当选择数据项 ▲B （单选，填字母：A. a[i][0] /B. a[i][1]）作为排序关键字对 a 数组升序排序。

#每次取出的新字符串中的尾字符与该串的首字符在原字符串中为相邻字符，由于python 中字符串正索引、负索引的特殊性，原字符串中的首字符和尾字符也可以视作相邻关系。综上分析，每次产生的新字符串的最后一个字符与首字符在原字符串中的对应位置关系为：最后一个字符的索引值为首字符的索引值减1

③ 在 compute 函数中，加框处代码若修改为“[i, data[i:n]]”是否影响该函数返回结果？请说明原因。

③不影响。因为比较字符串时，“$”比data中所有字符都要小

因为在拼接的时候是根据$进行的，而$比所有元素都要小，因此拼接在其后面的部分本身在进行比较的时候也不会比较到。

① 在 compute 函数中应当选择数据项__▲B__（单选，填字母：A. a[i][0] /B. a[i][1]）作为排序关键字对 a 数组升序排序。

② 执行语句 tran_data = trans(data)即可对 data 加密，结果存 tran_data。为实现该功能，请在 trans 函数中划线处补充代码。

③ 在 compute 函数中，加框处代码若修改为“[i, data[i:n]]”是否影响该函数返回结果？请说明原因。

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

```
def invert(tran_data):
    n = len(tran_data)
    a = []
    for i in range(n):   #存储密文每个字符索引，生成索引数组a
        a. append(i)
    for i in range(n - 1):
        for j in range(n - i - 1):
            if tran_data[a[j]] > tran_data[a[j + 1]]:
                a[j],a[j + 1] = a[j + 1],a[j]
shift = [0] * n  #用于记录相邻字符的跳转关系，最终形成单循环链
x=a[0]     #记录最小字符$的索引，以此为跳转的起点
b = [[] for i in range(128)]      #创建列表 b，共 128 个元素，每个元素均为空列表
for i in range(n):    #利用列表b记录各个字符在尾字符序列中的索引
    b[ord(tran_data[i])]. append(i)
for i in range(n):
    shift = b[ord(tran_data[a[i]])].pop(0)
decoded = ""
for i in range(n):
    x = shift[x]  #根据 shift中存储的跳转关系，逐步恢复原始字符串
    decoded = decoded + tran_data[x]
return decoded
#读取英文字母序列存入 data，代码略
tran_data = trans(data)
print("加密后的结果为 :", tran_data)
original = invert(tran_data)
print("解密后的结果为 :",original[0:-1])      # original[0:-1]表示 original 去掉末尾字符
```

#按照字符大小进行索引排序。结合加密过程可知，对尾字符序列进行升序排序即可得到对应的首字符序列。

#根据尾字符序列的字符找到首字符序列中的相同字符，然后得到首字符序列的尾字符，并将该字符的索引记录下来，形成跳转关系

去掉后缀"$"，形成原文

[解密过程算法分析]依据密文以及密文的组成由来恢复原始字符串。分析可知，某次新产生字符串的尾字符是其变换后的下一个字符串的首字符。在首字符序列中找到和尾字符序列遍历字符相同的字符，就可以找到该字符作为首字符对应生成的字符串。在原始字符串中，该字符序列的最后一个字符排在尾字符序列的前面。例如，如下图示例中，尾字符序列的第一个字符是“c”，我们可以由此对应到其在首字符序列中的位置，对应得到字符“c”的上一个字符为“b”；再让“b”作为新的尾字符序列，并对应到其在首字符序列中的位置，得到上一个字符为“a”...不断重复这个过程，就可以恢复原始字符串。