

绝密★考试结束前

2024年6月浙江省普通高校招生选考科目考试 技 术

姓名：_____ 准考证号：_____

考生须知：

1. 答题前，请务必将自己的姓名、准考证号用黑色字迹的签字笔或钢笔分别填写在试题卷和答题纸规定的位置上。
2. 答题时，请按照答题纸上“注意事项”的要求，在答题纸相应的位置上规范作答，在本试题卷上的作答一律无效。
3. 非选择题的答案必须使用黑色字迹的签字笔或钢笔写在答题纸上相应区域内，作图时可先使用2B铅笔，确定后必须用黑色字迹的签字笔或钢笔描黑。

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，回答第1至4题：

某校基于线下校史馆创建在线数字校史馆，将学校发展历史及校友的代表性成果、活动影像等资料，以文本、图像、视频等格式存储。校友可以用手机、电脑等终端登录数字校史馆查阅资料，也可以向在线问答机器人咨询学校相关信息。

1. 关于该数字校史馆中数据的叙述，正确的是 **A**
 - A. 数字校史馆中的数据有助于学校传承与发展，体现了数据的价值性
 - B. 不同格式的数据必须保存在不同的存储设备中
 - C. 学校的发展历史只能以同一种数据表现形式呈现
 - D. 文本、图像、视频都是结构化数据
2. 下列有关信息安全与保护的做法，合理的是 **A**
 - A. 定期备份数字校史馆数据
 - B. 未经校友同意发布其资料
 - C. 随意剪辑校友的活动影像
 - D. 以明文方式保存校友的注册信息
3. 为使问答机器人更准确地回答校史相关问题，下列方法可行的是 **C**
 - A. 增加校友的最新作品
 - B. 提高咨询所用终端的性能
 - C. 完善语料库中的校史资料
 - D. 提升数字校史馆的访问速度
4. 下列对校史馆资料的处理方式，不合理的是 **B**
 - A. 为了方便预览，为高清图像生成缩略图
 - B. 为了节省存储空间，将JPEG格式的图像转换成BMP格式
 - C. 为了方便传输，对高清视频进行压缩
 - D. 为了便于检索，将纸质文稿扫描成图像后识别出文字一并保存

阅读下列材料，回答第5至6题：

某小区智能回收箱可通过刷卡、扫码等方式开启箱门，箱内的传感器能识别可回收物的种类，当容量达到上限时，系统通知清运人员及时处理。居民可通过手机APP查看本人投递记录。

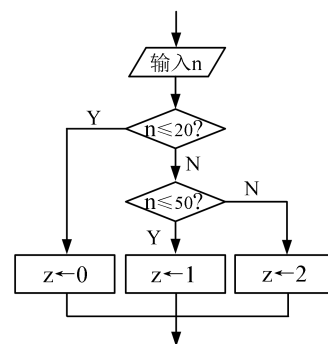
5. 关于该系统功能与软件设计的描述，正确的是 **B**
- A. 系统数据处理都可由传感器完成 B. 在设计系统时需考虑数字鸿沟问题
- C. 系统中的软件不包括手机APP D. 系统的软件升级是指增加新功能

6. 下列技术中，不能用于智能回收箱接入互联网的是 **D**

A. 5G B. Wi-Fi C. 光纤通信 D. RFID

7. 某同学根据第7题图所示流程图编写的Python程序段如下：

```
n = int(input())
if n <= 20:
    z = 0
if n <= 50:
    z = 1
else:
    z = 2
```



第 7 题图

用下列输入数据测试程序段与流程图，两者得到的z值不同的是 **D**

A. 60 B. 50 C. 30 D. 10

8. 某完全二叉树包含5个节点，其根节点在后序遍历序列、中序遍历序列中的位置序号分别记为x,y，则x-y的值为 **B**

A. 0 B. 1 C. 2 D. 3

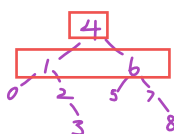
9. 栈初始为空，经过一系列入栈、出栈操作后，栈又为空。若元素入栈的顺序为“生”“旦”“净”“末”“丑”，则所有可能的出栈序列中，以“旦”结尾的序列个数为 **C**

A. 3 B. 4 C. 5 D. 6

10. 某二分查找算法的Python程序段如下：

```
i, j = 0, len(d) - 1
while i <= j:
    m = (i + j) // 2      # 语句①
    if key == d[m]:
        break
    elif key < d[m]:
        j = m - 1
    else:
        i = m + 1
```

1、4下标位置不变，执行次数分别为2、1
下标6位置上的值从查找2次变为4次



当d为[6, 12, 15, 18, 22, 25, 28, 35, 46]时，运行该程序段查找key，语句①的执行次数小于等于2；若将d修改为[6, 12, 15, 18, 22, 25, 28, 35, 46, 58]，重新运行该程序段，查找同一key值，则语句①的执行次数不可能为 **C**

A. 1 B. 2 C. 3 D. 4

11. 有如下Python程序段：

```
for i in range(0, len(a) - 1, 2):
    if i > 0 and a[i] < a[i - 1]:
        a[i], a[i - 1] = a[i - 1], a[i]
    if a[i] < a[i + 1]:
        a[i], a[i + 1] = a[i + 1], a[i]
```

$a[i-1] < a[i] > a[i+1]$ 状态

列表a有6个元素，运行该程序段后，a可能的值是 **D**

A. [2, 9, 8, 6, 9, 3] B. [9, 9, 8, 6, 3, 2] C. [9, 3, 6, 2, 8, 9] D. [6, 3, 9, 2, 9, 8]

当节点p和节点i交换数据区域后，节点p应该插入到下面两个位置中的其中一个，保证原数据的链表顺序保持不变



12. 使用列表d模拟链表结构（节点数 $n>0$ ），如第12题图a所示，每个节点包含数据区域和指针区域，h为头指针。现要按链表顺序将这n个节点中的数据依次存放到 $d[0][0]$ 、 $d[1][0] \dots d[n-1][0]$ 中，最终保持节点链接关系不变，结果如第12题图b所示。实现上述功能的Python程序段如下，方框中应填入的正确代码为

p, i = h, 0

while p != -1:

tp = d[p][1] **i表示物理顺序下标**

if p == i: **p是链表遍历下标**

i += 1 **p==i,表示不用调整**

elif p > i: **p<i则前面已经处理过**

d[i][0], d[p][0] = d[p][0], d[i][0]

i += 1

p = tp

调整头指针h及指针区域，保持节点链接关系不变，代码略

A. d[i][1] = d[p][1]

B. d[p][1] = d[i][1]

C. d[i][1] = p

D. d[p][1] = i

d[p][1] = i

d[i][1] = p

d[p][1] = d[i][1]

d[i][1] = d[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10分，第15小题9分，共26分）

13. 某监控设备可定时采集红绿灯信号灯状态数据，数据格式记为[a,b]，其中a、b分别为红灯和绿灯的状态值，0表示灯灭，1表示灯亮，如[0,1]表示红灯灭、绿灯亮。

现要编写程序，每隔1秒采集并检测信号灯是否存在如下异常状态：第一类，红绿灯同亮或同灭；第二类，红灯或绿灯超时，即保持同一状态时长大于上限值（如300秒）。检测到异常状态就发送相应信息。请回答下列问题：

(1) 若检测到“红绿灯同亮”异常，则采集到的数据是 **D**（单选，填字母）。

A. [0, 0]

B. [0, 1]

C. [1, 0]

D. [1, 1]

(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。

tlimit = 300

设置信号灯保持同一状态时长上限值

pre = [-1, -1]

t = [0, 0]

t[0]、t[1]分别记录红灯、绿灯保持同一状态的时长

while True:

接收一次采集到的状态数据，存入d，代码略

if **d[0]==d[1] 或 d[0]+d[1]!=1**:

if d[0] == 1:

发送“红绿灯同亮”信息，代码略

else:

发送“红绿灯同灭”信息，代码略

for i in **range(2) 或 range(len(d))**:

if d[i] == pre[i]:

t[i] += 1

if **t[i]>tlimit**:

if i == 0:

发送“红灯超时”信息，代码略

else:

发送“绿灯超时”信息，代码略

else:

t[i] = 1

pre = d

延时1秒，代码略

两盏灯相同状态有两种：同亮、同灭

18->15->19->12->29->23

p tp i

	数据区域	指针区域
0	15	4
1	18	0
2	12	5
3	23	-1
4	19	2
5	29	3

第12题图a

	数据区域	指针区域
0	18	1
1	15	2
2	19	3
3	12	4
4	29	5
5	23	-1

第12题图b

此时d[i]节点是链表未遍历过的节点，删除i节点，换到p前面

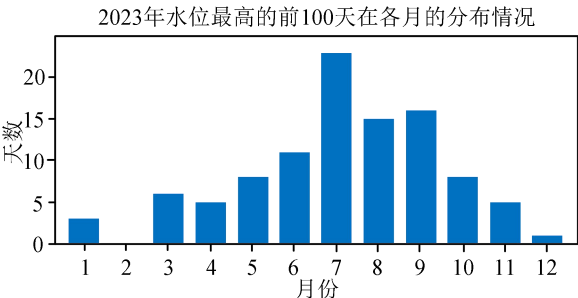
B

14. 某研究小组拟采集某水域水位及周边土壤含水量等数据，进行地质灾害监测。该小组在实验室搭建了一个模拟系统，该系统的智能终端获取传感器数据，并通过无线通信方式将数据传输到 Web 服务器，服务器根据数据判断出异常情况后，通过智能终端控制执行器发出预警信号。请回答下列问题。

- (1) 该模拟系统中的传感器和执行器 ▲^B (单选，填字母：A. 必须连接在不同智能终端 / B. 可以连接在同一智能终端)。
- (2) 水位传感器和土壤水分传感器连接在同一智能终端，服务器能正常获取土壤含水量数据，但不能正常获取水位数据，以下故障与该现象无关的是 ▲^C (单选，填字母)。
- A. 水位传感器故障
B. 水位传感器与智能终端连接故障
C. 智能终端无法与服务器通信
- (3) 下列关于该系统设计的说法，正确的有 ▲^{BC} (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. 水位、土壤含水量等数据的采集时间间隔不能相同
B. 水位、土壤含水量等数据可用数据库存储 **传输数据时添加监测点或智能终端或传感器的信息，可以是设备的编号、名称、IP、Mac 等，能唯一识别该设备即可。**
C. 可以基于 Flask Web 框架编写服务器程序
D. 系统获取数据的程序可以只部署在服务器端
- (4) 现场实地测试时需要设置多个监测点，每个监测点配备一个智能终端。为使服务器能区分出数据的监测点来源，从智能终端的角度写出一种可行的解决方法。
- (5) 研究小组整理出近几年的水位(日平均)数据，部分数据如第 14 题图 a 所示(图中水位单位为“米”)。现要统计 2023 年水位最高的前 100 天在各月的分布情况，并绘制如第 14 题图 b 所示的柱形图。

年	月	日	水位
2020	1	1	2.11
2020	1	2	2.09
2020	1	3	2.1
2023	12	28	2.21
2023	12	29	2.23
2023	12	30	2.28
2023	12	31	2.26

第 14 题图 a



第 14 题图 b

实现上述功能的部分 Python 程序如下：

```
import pandas as pd
import matplotlib.pyplot as plt
df=pd.read_excel("data.xlsx")
df=df[df["年"]== 2023] # 筛选出 2023 年的数据
```



```
plt.bar(df1["月"],df1["水位"]) # 绘制柱形图
#设置绘图参数，显示如图第 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 241 (选 3 项，填数字序列，少选、多选、错选或次序错均不得分)。
```

- ①df1 = df1.groupby("月",as_index = False).count() # 分组计数
②df1 = df.sort_values("水位",ascending = False) # 降序排序
③df1 = df1.sort_values("水位",ascending = False) # 降序排序
④df1 = df1.head(100) # 获取前 100 条数据

- (6) 观察第 14 题图 b 可知，2023 年水位最高的前 100 天分布在 7 月的天数最多。现要筛选出 2023 年 7 月的水位数据以便进一步分析，可在第 (5) 小题处理结果的基础上，再运行如下语句，请在划线处填入合适的代码。

```
df2 = df[▲]
df["月"]==7
```

年	月	日	水位
2020	1	1	2.11
2020	1	2	2.09
2020	1	3	2.1
2023	12	28	2.21
2023	12	29	2.23
2023	12	30	2.28
2023	12	31	2.26

15. 某数据序列 `data` 中的元素均为小于 127 的正整数。现在要对 `data` 进行加密，处理过程分“变换”和“重排”两步。

“变换”处理方法是使用指定的 n 组序列 $R_0, R_1, \dots, R_{n-1}$ 依次对 `data` 进行变换。利用 R_i 对 `data` 进行变换的过程是：在 `data` 中查找所有与 R_i 相同的子序列，将找到的每个子序列中的元素值加上 R_i 的长度值 L_i ，并在各子序列前插入一个标记元素（值为 $127+L_i$ ），这些子序列及标记元素不再参与后续的变换。

如 `data` 为 `[3, 5, 1, 6, 3, 8, 7, 5, 1, 8, 7]`，指定的两组序列为 `[5, 1]`、`[3, 8, 7]`，“变换”处理后的 `data` 为 `[3, 129, 7, 3, 6, 130, 6, 11, 10, 129, 7, 3, 8, 7]`。

对 `data` “重排”处理通过给定的 `shuff` 函数实现。

请回答下列问题：

(1) 若 `data` 为 `[3, 5, 1, 6, 3, 8, 7, 5, 1, 8, 7]`，指定的两组序列为 `[5, 1]`、`[8, 7]`，经过“变换”处理后，`data` 中插入的标记元素个数为 ▲ 4。

(2) “重排”处理的 `shuff` 函数如下：

```
def shuff(data, c):    # 根据列表 c 对列表 data 进行重排
    # 若列表 data 的长度不是列表 c 长度的整数倍，则用 0 补足，代码略
    m = len(c)
    s = [0] * m
    k = 0
    while k < len(data):
```

```
        for i in range(m):
```

```
            s[i] = data[k + i]
```

将 `data` 中的数据分组存入 `s[i]`

```
        for i in range(m):
```

```
            data[k + i] = s[c[i]]
```

再将 `s[c[i]]` 中的数据按分组存回到 `data` 中

`c` 是索引数组 `c[0]=1` 表示 0 位置上放 `data[1]` 的值

```
        k += m
```

若 `data` 为 `[3, 129, 7, 3, 130, 6, 11, 10]`，`c` 为 `[1, 3, 0, 2]`，调用 `shuff(data, c)` 后，`data` 的最后一个元素值为 ▲ 11。
6, 10, 130, 11

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

```
def compare(data, i, r):
    # 函数功能：返回 data 从索引 i 位置、r 从索引 0 位置开始的连续相等元素的个数
    # 例如 r 为 [7, 3, 6], data 从索引 i 位置开始的元素依次为 7, 6, 7, 3, ...，函数返回 1
    j = 0
    while j < len(r) and i + j < len(data):
        if r[j] != data[i + j]:
            break
        else:
            j += 1
    return j
```

`j` 是匹配的长度，所以 `j == len(r)` 表示完全匹配

```

def trans(data, r, segs):    记录匹配情况
    newsegs = []
    for s in segs:
        if s[0] == 0:    未变换
            h = i = s[1]    开始位置
            m = len(r)
            while i + m <= s[2] + 1:    结束位置+1
                if compare(data, i, r) == m:    从i位置开始匹配上了
                    if i > h:    处理i之前不匹配段
                        newsegs.append([0, h, i-1])    # 为 newsegs 追加一个元素
                    newsegs.append([m, i, i + m - 1])    处理i开始的匹配段
                    i += m
                    h = i    ② 更新不匹配段的开始
                else:
                    i += 1
            if h <= s[2]:    最后一段不匹配
                newsegs.append([0, h, s[2]])
            else:
                newsegs.append(s)
    return newsegs

def update(data, segs):
    for s in segs:
        if s[0] != 0:
            data.append(0)
            p = len(data) - 1    最后一个位置开始
            for i in range(len(segs) - 1, -1, -1):    从后往前
                for j in range(segs[i][2], segs[i][1] - 1, -1):    对需要处理的部分从后往前每个值加m
                    data[p] = data[j] + segs[i][0]
                    p -= 1
            if segs[i][0] > 0:
                data[p] = 127 + segs[i][0]
                p -= 1
# 读取待加密数据存入 data,读取指定的若干组用于变换的序列存入 rs, 代码略
'''
列表 segs 用于记录 data 的变换信息, segs[i] 包含三个元素, segs[i][0]、segs[i][1]、segs[i][2] 分别表示
data 中一个子序列的状态、起始位置和结束位置, 如果 segs[i][0] 为 0, 则表示该子序列未经过变
换。
'''
segs = [[0, 0, len(data) - 1]]
for r in rs:    指定的两组序列为[5, 1]、[3, 8, 7],
    segs = trans(data, r, segs)    # 根据 r 更新 segs
update(data, segs)    # 利用 segs 完成对 data 的变换操作
c = [1, 3, 0, 2]
shuff(data, c)
# 输出加密后的 data 序列, 代码略

```