台州市2025届高三第一次质量评估试题

技　术

第一部分 信息技术（共 50 分）

一、选择题（本大题共 12 题，每题 2 分，共 24 分。每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 3 题：

某市为了优化公共交通体系，引入了无人驾驶出租车。该出租车配备自动驾驶系统，搭载激光雷达、高清摄像头等设备，行驶中实时采集车距、车速和路面图像等路况数据，结合深度学习模型，能够精准识别交通标识、路障、行人车辆等，以应对复杂的交通状况。乘客通过手机 APP 进行预约乘车，体验全程无人驾驶服务。

1. 关于无人驾驶出租车行驶中实时采集的路况数据，下列描述正确的是 D

A. 路况数据只有图像这一种表现形式

B. 未经处理的路况数据是没有价值的

C. 高清摄像头采集的车辆行驶视频是结构化数据　非结构化数据

D. 路况数据需要经过数字化才能被处理　结合阅读材料，是计算机处理

2. 下列有关信息安全与保护的做法，合理的是 A

A. 定期备份无人驾驶出租车的行驶数据　　B. 以明文的方式存储用户的相关信息

C. 将乘车记录发布至社交平台　　D. 乘客将 APP 账号分享给他人使用

3. 关于自动驾驶系统中涉及的人工智能技术，下列描述正确的是 B

A. 系统根据路况规划路线，需要手工构造的知识库和推理引擎

B. 系统通过学习海量数据实现交通标识的精准识别　联结主义

C. 激光雷达自动采集车距、车速，是一种人工智能技术的应用　传感技术

D. 无人驾驶出租车的运营，造成部分出租车司机失业，应限制其发展

阅读下列材料，回答第 4 至 6 题：

某学校食堂引入一款智能结算系统。师生将选好的菜品餐盘放入该系统结算台的识别区，摄像头识别出菜品名称，服务器根据所选的菜品计算价格；师生将校园卡放置在结算台的刷卡机上，即可完成支付；师生在校内外均可通过手机 APP 预订菜品、查看用餐记录和消费明细。

4. 下列关于该智能结算系统组成的说法，正确的是 B

A. 该系统中的用户只有师生

B. 手机 APP 属于该系统的应用软件

C. 识别菜品名称的摄像头是输出设备　输入设备

D. 该系统中的数据仅包含菜品的名称和单价

5. 关于该智能结算系统功能与应用的描述，不正确的是 B

A. 校园卡的刷卡支付功能可采用射频识别技术实现

B. 菜品名称、单价等信息无需事先存储于该系统数据库

C. 可通过分析食堂点餐记录得出各菜品的受欢迎程度

D. 通过手机 APP 查看消费明细属于数据查询功能

6. 下列关于该智能结算系统中网络技术的说法，正确的是 A

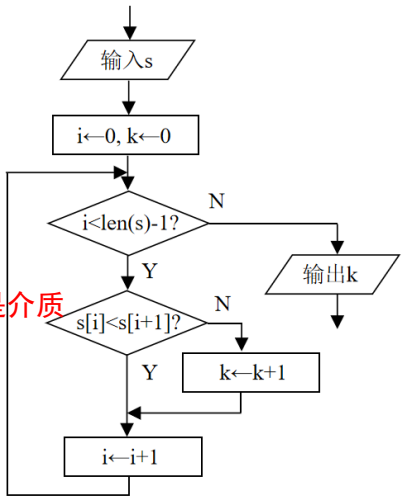
A. 通过手机 APP 查询消费明细需要遵循网络协议

B. 移动终端必须连接学校局域网才能访问该系统

C. 摄像头通过 Wi-Fi 传输菜品信息时不需要传输介质 无线电波也是介质

D. 若该系统的服务器未连接网络，师生也能通过 APP 预订菜品

7. 某算法的部分流程图如第 7 题图所示，执行这部分流程，若输出 k 的值为 3，则输入 s 的值可能是 C



第 7 题图

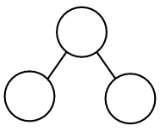
A. "13267459" k=2　　B. "31762594" k=4

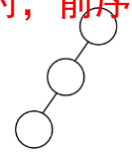
C. "26374859" k=3　　D. "62738495" k=4

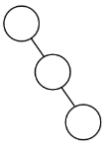
k存放多个递减序列总和

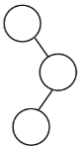
8. 某二叉树的前序遍历序列和中序遍历序列相同，下列各项中符合该二叉树结构的是 C

无左子树时，前序和中序相同

A. 

B. 

C. 

D. 

9. 有如下 Python 程序段：

```
from random import randint
s = "abcde"
key = "123"
res = ""
for i in range(len(s)):
    k = int(key[i%len(key)])
    if randint(0,1) == 0:          # 随机生成整数 0 或 1
        res += chr((ord(s[i]) - ord("a") + k) % 26 + ord("a"))
    else:
        res += chr((ord(s[i]) - ord("a") - k) % 26 + ord("a"))
```

12312
abcde
+bdfeg
- zzzcc

执行该程序段后，变量 res 的值不可能是 B

A. "bdzec"　　B. "bzfzz"　　C. "zzfcg"　　D. "zdzeg"

10. 小杨编写了一个 Python 程序，部分代码如下：

```
# 输入数据，存储到列表 d 中，代码略
n = len(d)
for i in range(1, n):
    for j in range(i, n):
        if d[j] < d[j - 1]:
            d[j], d[j - 1] = d[j - 1], d[j]
```

从前往后进行的冒泡排序错误表达，违反初定终变规律

执行该程序段，输入下列数据，列表 d 中元素未按升序排列的是 C

A. 1, 2, 4, 3, 5　　B. 4, 1, 2, 3, 5　　C. 5, 2, 1, 4, 3　　D. 5, 1, 2, 3, 4

11. 某括号序列中，若左括号出现的顺序及个数与右括号保持一致，则称该序列中的括号是匹配的。例如序列“())(”中的括号是不匹配的，可将其中第 3、第 4 个括号修改为“()”使其重新匹配。现给出一个长度为偶数的不匹配序列，为使其重新匹配，统计最少需要修改的括号数。实现上述功能的 Python 程序段如下：

```
s = input()      # 输入括号序列，序列中仅包含“(”和“)”两种字符
top = -1; ans = 0
for i in range(len(s)):
    if s[i] == "(":
        top += 1
    else:  碰到“）”时
        if [top == -1]:  如果栈为空，则该“）”需要改成“（”
            top += 1     修改后为“（”并入栈
            ans += 1     修改数+1
        else:  如果栈不为空，则与前面的“）”抵消
            [（2）top-=1]
ans += [（3）] (top+1)//2  遍历完毕没有右括号了，栈内的左括号有一半要改成右括号。top从0开始，所以数量要+1，再除2
```

上述程序段中方框处可选代码为：

①top != -1　②top == -1　③top += 1　④top -= 1　⑤top // 2　⑥(top + 1) // 2

则（1）（2）（3）处代码依次为　D

A. ①③⑥　　B. ①④⑤　　C. ②④⑤　　D. ②④⑥

12. 某二分查找算法的 Python 程序段如下：

```
n = len(a)
L = 0
R = (n - 1) // 2 只在前半截找 语句①
while L <= R:
    m = (L + R) // 2          # 语句②
    if key == a[m]: break
    if key < a[m]:
        R = m - 1
    else:
        L = m + 1
if L <= R or R >= 0 and a[n - R - 1] == key:  a[n-R-1]是后半截与前半截的镜像数据
    print("YES")
else:
    print("NO")
```

执行该程序段后，下列说法正确的是　D

A. 若列表 a 有 8 个元素，则语句②最多被执行 4 次　R=3 L=0,4个元素的二叉树，最多3层

B. 若 a 为[1, 5, 8, 7, 3]，key 为 7，输出结果为“NO”　可以找到7，结果是YES

C. 若要实现 key 的查找，a 中元素应按升序排列

D. 将语句①修改为“R = n - 1”，可能会影响程序输出的结果

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

13. 某路边有一排照明装饰灯，编号依次为 1~n。现发现有多个装饰灯不亮，受维修成本的限制，只对其中的一部分进行维修，维修后保证有 k 个编号连续的装饰灯能够正常照明。编写程序，根据已损坏的装饰灯编号，输出最少需要维修的装饰灯数量。请回答下列问题：

（1）若路边有 10 个照明装饰灯（编号 1~10），其中编号为 1、4、8、10 的装饰灯不亮，维修后需保证有 6 个编号连续的装饰灯能正常照明，则最少需要维修的装饰灯数量是 ▲1 。修理4号灯，1~7就正常

（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

```
# 输入照明装饰灯的总数 n、编号连续的正常照明装饰灯数量 k，代码略
# 读取不亮的装饰灯编号，存入 d，代码略
v = [0] * (n + 1)
c = [0] * (n + 1)
ans = n
for i in ____①____:
    v[d[i]] = 1
for i in range(1, n + 1):
    c[i] = c[i-1] + v[i]
i = 1
while i <= n - k + 1:
    last = ____②____
    num = c[last] - c[i - 1]
    if ____③____:
        ans = num
    i = i + 1
print(ans)
```

range(len(d))

不亮的装饰灯编号存入d中。
在 for 循环中，执行了 v[d[i]]=1语句，该语句功能为将损坏的装饰灯对应的值在v中设置为 1，由于损坏的装饰灯存入d中，在此时仅需遍历d中所有数据即可实现该功能

此循环功能为从第一个装饰灯开始，统计到当前装饰灯位置共损坏了几个装饰灯。

变量 num 为当轮循环中统计出最少修理数量，变量 ans 为最后要求的最小修理数量，此时若 num 小于 ans，则将num 作为最小值

while 循环执行的条件为i<=n-k+1，即根据最少需要点亮的装饰灯数量，统计需要执行几次循环，如k=6时至少需要判断5次，对应 n-k+1

i+k-1

变量 num 用于统计当前连续的照明灯中，至少需要修理几个装饰灯，由于至少需要保证k个装饰灯是连续的,则只需在变量c中去计算 i+k位置与i位置需要需要修理的装饰灯数量的差即可。
根据 num=c[last]-c[i-1]可知，last 对应值为 k+i-1

ans>num

14. 小杨搭建了一个模拟药品仓储环境监测系统的实验，用于监测药品储存区域的温度和湿度。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，服务器根据采集的温湿度数据与设定的相应阈值进行判断，出现异常时，通过智能终端控制执行器发出预警信号。用户可通过浏览器查询历史数据。请回答下列问题：

（1）用户通过浏览器查询的历史数据存储在 B▲ （单选，填字母：A. 传感器 / B. 数据库 / C. 智能终端）

（2）下列功能需要在服务器端程序实现的是 BD▲ （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. 打开或关闭执行器　　B. 处理浏览器访问请求

C. 获取温湿度传感器上的数据　　D. 根据采集的数据和阈值判断异常情况

（3）系统正常工作一段时间后，通过浏览器查看历史数据，发现有多次温湿度数据超过阈值时，执行器均未发出预警信号。下列解决问题的方法合理的是 C ▲ （单选，填字母：A. 更换服务器 / B. 重新设定阈值 / C. 更换执行器）。

（4）不同类别的药品在储存时对温湿度有不同的要求，现要将药品进行分区域储存。为了对不同区域温湿度实现有效监测，从硬件的角度写出一种可行的办法 ▲ ，从软件的角度写出一种可行的办法 ▲ 。

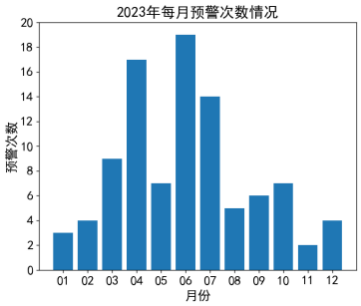
硬件：设置多个监测点，每个监测点配备智能终端和温湿度传感器等设备

软件：每个监测点设置不同的阈值；或 为每个智能终端设置不同的编号，以便在数据发送时能够区分；

（5）小杨将系统中过去一年的湿度数据导出，部分数据如第 14 题图 a 所示（日期格式为“年/月/日”）。若设定的湿度阈值为 70，当采集的湿度值超过阈值时，执行器会发出预警信号。现要统计每月执行器发出预警信号的次数，并绘制如第 14 题图 b 所示的柱形图。

日期	时间	类别	监测值
2023/01/01	00:00	湿度	68
2023/01/01	00:30	湿度	68
2023/01/01	01:00	湿度	67
2023/12/31	22:30	湿度	69
2023/12/31	23:00	湿度	74
2023/12/31	23:30	湿度	51

第 14 题图 a



第 14 题图 b

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。

```
import pandas as pd
import matplotlib.pyplot as plt
df = pd.read_excel("data.xlsx")
df.insert(1, "月份", "")                    # 插入列
for i in df.index:
    m = df.at[i, "日期"]
    df.at[i, "月份"] = m[5:7]
[          ]
plt.bar(df1.月份, df1.监测值)
# 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__②③__（选填 2 项，填数字序列，少选、多选、错选或次序错均不得分）

① df1 = df1[df1["监测值"] > 70]

② df1 = df[df["监测值"] > 70]

③ df1 = df1.groupby("月份", as_index = False).count()

④ df1 = df1.groupby("月份", as_index = False).sum()

15. 某计算机中，CPU 通过与 Cache（高速缓存）交换数据，减少直接访问主存的次数，从而提升数据的读取速度，CPU 通过主存单元编号来实现数据的访问。当 CPU 访问的主存单元数据在 Cache 中时，访问结果为命中，无需从主存调入数据；当 CPU 访问的主存单元数据不在 Cache 中时，访问结果为缺失，需要从主存中将被访问的主存单元数据调入 Cache，若 Cache 容量已满，则需要进行替换操作。

替换操作如下：根据 CPU 依次访问主存单元数据的编号，优先替换 Cache 中后续不再被访问的数据；若 Cache 中的数据在后续均会被访问，则选择最晚被访问到的数据进行替换。

例如，某 Cache 的容量为 3，且其中已有编号为 10 和 20 的主存单元数据，CPU 依次访问的主存单元编号为“10、30、40、50、10、30、40”，其过程如下表所示：

步骤	Cache 状态	访问主存单元编号	访问结果	Cache 调度操作
①	10，20	10	命中	无
②	10，20	30	缺失	将编号为 30 的主存单元数据调入
③	10，20，30	40	缺失	替换编号为 20 的主存单元数据
④	10，40，30	50	缺失	替换编号为 40 的主存单元数据
……	……	……	……	……

编写程序，根据 CPU 依次访问主存单元数据的编号（编号范围为 0~1023），计算访问结果为缺失的次数。请回答下列问题：

缺 缺 缺 缺

（1）若 Cache 容量为 2，且初始状态为空，CPU 对编号为“1，2，3，2，3，1”的主存单元数据依次进行访问，此过程中访问结果为缺失的次数为 ▲4 。

（2）定义 getpos(info)函数，参数 info 列表元素表示 CPU 依次访问主存单元数据的编号。

步骤	Cache 状态	访问主存单元编号	访问结果	Cache 调度操作
①	空	1	缺失	将编号为 1 的主存单元数据调入
②	1	2	缺失	将编号为 2 的主存单元数据调入
③	1，2	3	缺失	替换编号为 1 的主存单元数据
④	3，2	2	命中	无
⑤	3，2	3	命中	无
⑥	3，2	1	缺失	替换编号为 3 的主存单元数据

```
def getpos(info):
    n = len(info)
    last = [-1] * 1024 #用于存储数据最后一次出现的索引
    next = [n] * 1024 #存储同一数据后一次出现的索引
    for i in range(n):
        k = info[i]
        if last[k] != -1:          #若该数据已被访问过
            next[last[k]] = i  #更新该数据上一次出现的后继指针指向当前数据
        last[k] = i    #更新该数据最后一次出现的索引
    return next        #函数返回值为每个主存单元数据的后继元素
```

若 info 为[1,2,3,2,3,1]，执行语句 nxt = getpos(info)后，nxt[0]的值为 5▲ 。nx=[5,3,4,6……]

（3）定义 update(head, data, val, num)函数，函数功能是在链表 data 中插入一个新的节点，并保持链表节点按照 val 的值降序排列。请在划线处填入合适的代码。

```
def update(head, data, val, num):
    pre = -1; p = head
    while p != -1 and ▲data[p][0] > val:
        pre = p
        p = data[p][2]
    data.append([val, num, p])   # 为 data 追加一个元素[val, num, p]
    if pre == -1:
        head = len(data) - 1
    else:
        data[pre][2] = len(data) - 1
    return head, data
```

在pre 和p之间插入val元素

（4）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 Cache 的容量存入变量 m，代码略
  读取 CPU 依次访问主存单元数据的编号存入 info 列表，代码略 '''
```

```
data = []
vis = [False] * 1024
ans, cnt, head = 0, 0, -1
next = getpos(info)
for i in range(len(info)):
    if ①:
        ans += 1
        vis[info[i]] = True
        if cnt < m:
            cnt += 1
        else:
            vis[data[head][1]] = False
            ②
        head, data = update(head, data, next[i], info[i])
print(ans)
```

#标记当前遍历到的主存单元数据不在 Cache 中

① vis[info[i]] == False 或 not vis[info[i]]

#更新缺失次数

#标记该数据状态，将其调入或替换入 Cache 中

#若当前 Cache 容量未满，则将其调入 Cache 中

#更新 Cache 已用容量

#若当前 Cache 容量已满，则需进行替换操作

#根据处理规则，将被替换数据在 Cache 中的状态更新为 False

② head = data[head][2]

#删除表头元素，更新被替换链的表头指针