Exam 01

2025 年 3 月 21 日

目录

依据考纲，数据结构和数据库分数，各占 50%, 以下为大题的考点：

0.1 数据结构

在数据结构上，容易考的主要有树的前中序遍历，哈夫曼树的构造，图到树的转化。代码主要在二分查找，插入、选择、冒泡排序。

0.2 数据库

数据库主要考的是关系 E-R 模型，主码判定，SQL 语句。

---

1 数据结构

1.1 树的遍历

Example 1. 给定一棵二叉树的前序遍历序列为 $ABDEGCFH$,中序遍历序列为 $DBGEACFH$, 请：

1. 画出这棵二叉树
2. 写出这棵二叉树的后序遍历序列
3. 用 C 语言实现该二叉树的构建和后序遍历

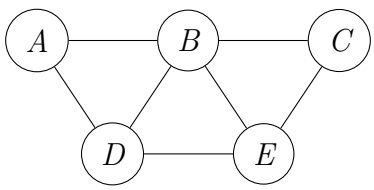
1.2 哈夫曼树

Example 2. 给定以下字符及其出现频率：$A(5),\ B(29),\ C(7),\ D(8),\ E(14),\ F(23),\ G(3),\ H(11)$

1. 构造哈夫曼树
2. 给出每个字符的哈夫曼编码
3. 计算平均编码长度

1.3 图与树的转换

Example 3. 给定如下的无向图 G:



1. 从顶点 A 开始，构造该图的深度优先生成树

2. 从顶点 A 开始，构造该图的广度优先生成树

1.4 查找与排序

Example 4. 请实现二分查找算法，并分析其时间复杂度。

```
int binarySearch(int arr[], int n, int key) {
    // 请完成代码
}
```

Example 5. 给定数组 *[64, 34, 25, 12, 22, 11, 90]*:

1. 用冒泡排序对其进行排序，写出每一趟排序后的结果

2. 用选择排序对其进行排序，写出每一趟排序后的结果

3. 用插入排序对其进行排序，写出每一趟排序后的结果

Example 6. 请用 C 语言实现冒泡排序、选择排序和插入排序算法，并比较它们的时间复杂度和空间复杂度。

```
void bubbleSort(int arr[], int n) {
    // 请完成代码
}

void selectionSort(int arr[], int n) {
    // 请完成代码
}

void insertionSort(int arr[], int n) {
    // 请完成代码
}
```

1.5 栈与队列应用

Example 7. 请用 C 语言实现一个栈，并利用该栈实现表达式求值算法。以下是一个简化版本，仅考虑整数、加减乘除运算和括号。

```
typedef struct {
    int data[100];
    int top;
} Stack;

int evaluateExpression(char* expression) {
```

```
    // 请完成代码
}
```

Example 8. 使用队列实现二叉树的层序遍历。

```
typedef struct BiTNode {
    char data;
    struct BiTNode *lchild, *rchild;
} BiTNode, *BiTree;

void levelOrderTraversal(BiTree T) {
    // 请完成代码
}
```

1.6 线性表操作

Example 9. 实现单链表的逆置操作。

```
typedef struct LNode {
    int data;
    struct LNode *next;
} LNode, *LinkList;

void reverseList(LinkList *L) {
    // 请完成代码
}
```

Example 10. 给定一个有序链表，请删除链表中的重复元素，使每个元素只出现一次。

```
typedef struct LNode {
    int data;
    struct LNode *next;
} LNode, *LinkList;

void removeDuplicates(LinkList L) {
    // 请完成代码
}
```

1.7 更多树的操作

Example 11. 给定二叉树的前序遍历序列为 $GDAFEMHZ$，中序遍历序列为 $ADEFGHMZ$。请：

1. 画出该二叉树

2. 计算该二叉树的深度

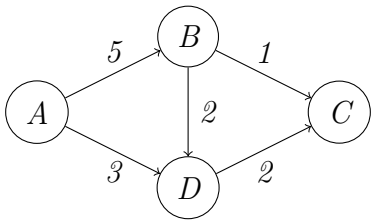
3. 求出所有叶子节点

Example 12. 编写函数统计二叉树中度为 2 的结点个数。

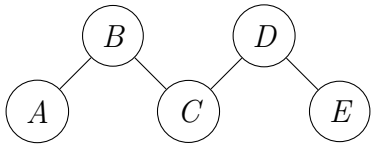
```c
typedef struct BiTNode {
    char data;
    struct BiTNode *lchild, *rchild;
} BiTNode, *BiTree;

int countNodes2Degree(BiTree T) {
    // 请完成代码
}
```

1.8 图的应用

Example 13. 请用表示下图，使用 *Dijkstra* 算法描述从顶点 A 到其他各顶点的最短路径，计算过程。



Example 14. 判断下图是否为一棵树，如果不是，给出理由；如果是，画出对应的树结构。



1.9 高级排序算法

Example 15. 给定数组 *[38, 27, 43, 3, 9, 82, 10]*，请演示快速排序的详细过程，写出每一趟排序后的结果。

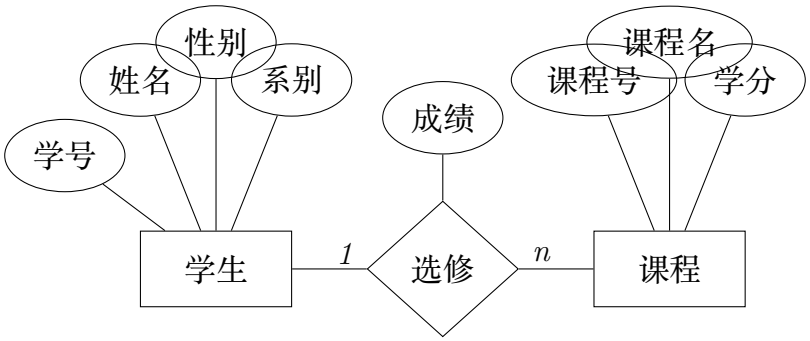
Example 16. 请用 *C* 语言实现快速排序算法。

```c
void quickSort(int arr[], int low, int high) {
    // 请完成代码
}
```

2 数据库

2.1 关系模型与 E-R 图

Example 17. 将下述 *E-R* 图转换为关系模式，并标明主码、外码。



Example 18. 根据以下描述，画出 *E-R* 图，然后转换为关系模式：
一个图书管理系统需要管理图书馆的图书和读者信息。每本图书有唯一的编号、书名、作者、出版社、出版日期和价格。每位读者有唯一的借书证号、姓名、性别、单位和电话。读者可以借阅多本图书，每本图书也可以被多位读者借阅（但不能同时）。系统需要记录借阅日期和归还日期。

Example 19. 给定关系模式 *R(A,B,C,D,E,F)*，其中函数依赖集为：

$$F = \{A \to B, BC \to D, AE \to F, F \to A\}$$

1. 求出 R 的所有候选码
2. 判断该关系模式属于哪种范式

Example 20. 给定关系模式 *S(A,B,C,D,E)*，其中函数依赖集为：

$$F = \{A \to B, B \to C, D \to E, C \to D\}$$

1. 求出 S 的候选码
2. 该关系模式满足哪一级范式?

2.2 SQL 语句

Example 21. 假设有以下关系模式：

学生：$Student(\underline{Sno}, Sname, Ssex, Sage, Sdept)$

课程：$Course(\underline{Cno}, Cname, Cpno, Ccredit)$

学生选课：$SC(\underline{Sno}, \underline{Cno}, Grade)$

请用 *SQL* 语句完成以下操作：

1. 查询计算机系（*CS*）年龄在 *20* 岁以下的学生的学号和姓名
2. 查询选修了课程名为 *"* 数据库 *"* 的学生学号和成绩
3. 查询没有选修任何课程的学生姓名
4. 查询选修了全部课程的学生姓名

Example 22. 假设有以下关系模式：

$$\begin{aligned}
&\text{部门}：Department(\underline{Dno}, Dname, Dlocation)\\
&\text{员工}：Employee(\underline{Eno}, Ename, Esalary, Dno)\\
&\text{项目}：Project(\underline{Pno}, Pname, Budget, Dno)\\
&\text{员工参与项目}：Works_on(\underline{Eno}, \underline{Pno}, Hours)
\end{aligned}$$

请用 *SQL* 语句完成以下查询：

1. 查询月薪超过 *3000* 元的员工姓名及其部门名称
2. 查询参与了名为 *"* 数据库开发 *"* 项目的员工姓名及工作时间
3. 查询平均月薪最高的部门编号、名称及平均月薪
4. 查询至少参与了 *3* 个项目的员工姓名
5. 查询没有员工参与的项目名称

Example 23. 针对上述学生-课程关系模式，编写 *SQL* 语句完成以下操作：

1. 将 *"* 数据库 *"* 课程的学分增加 *1* 学分
2. 删除没有选修任何课程的学生记录
3. 创建一个新表存储各系的学生人数
4. 为 *Student* 表的 *Sage* 字段添加检查约束，要求年龄在 *16* 到 *30* 之间