数据结构主要教学内容

C++相关:构造函数/析构函数/虚函数/纯虚函数/模板基类的申明/模板派生类的申明/模板类申明内函数定义/模板类申明外函数定义/模板类的实例化/const 变量及 const 函数/传引用调用/异常定义及处理

第 2 章 线性表

2.1 线性表的定义

概念:线性表/元素序号或位置/直接前驱或直接后继/表长/空表/起始结点/终端结点

基本运算:创建空表/清除元素/求表长/插入元素(元素位置有效范围)/删除元素(元素位置有效范围)/搜索元素/访问元素(元素位置有效范围)/遍历元素

2.2 线性表的顺序实现

存储信息:元素数组/数组规模/表长

基本运算实现及其(最好/最差/平均)复杂度

主要缺点:插入或删除元素可能造成大量元素拷贝

特殊运算:doubleSpace

2.3 线性表的链接实现

概念:不带头结点的单链表/带头结点的单链表/不带

头尾结点的双链表/带头尾结点的双链表/单循环链表/双循环链表

存储信息:指向起始结点的指针/指向头结点的指针/指向头尾结点的指针/指向终端结点的指针

基本运算实现及其时间复杂度

特别注意:单链表插入/删除操作需定位到被操作结点的直接前驱结点/头结点/存储指针

特殊运算:兄弟协作法清除元素/就地置逆

2.5 线性表的应用

第 3 章 栈

3.1 栈的定义

概念:栈/栈顶/栈底/栈顶元素/空栈/入栈/进栈/出栈/后进先出

基本运算:创建空栈/进栈/出栈/读栈顶元素/判栈空

出栈序列:构成出栈序列的充要条件/已知出栈序列获得入栈出栈序列/(满足某条件的)出栈序列数目

3.2 栈的顺序实现

概念及思路:顺序栈, 栈底保持不变/栈顶浮动

存储信息:元素数组/数组规模/栈顶位置

特殊运算:doubleSpace

基本运算实现及其(最好/最差/平均)复杂度

3.3 栈的链接实现

概念及思路:链接栈， 不带头结点的单链表， 起始结点为栈顶元素

存储信息:指向栈顶元素的指针

基本运算实现及其(最好/最差/平均)复杂度

3.5 栈的应用

理解函数调用栈的原理

括号匹配算法:遇开括号入栈,遇闭括号出栈及匹配

中缀表达式/前缀表达式/后缀表达式:理解二元运算符的优先级/结合性/操作数/被操作数,理解中缀表达式转后缀表达式的手写方法(操作数及被操作数相对顺序不变)及计算机算法(含鲁棒性检测),理解后缀表达式的求值算法

第 4 章 队列

4.1 队列的定义

概念:队列/队头/队尾/队头元素/队尾元素/队列长度/空队列/入队/出队/先进先出

基本运算:创建空队/入队/出队/读队头元素/判队空

4.2 队列的顺序实现

概念及思路:队头固定/队尾浮动则出队需大量元素移动， 队头/队尾均浮动但需解决空间重复利用问

题形成循环队列

存储信息:元素数组/数组规模/队头元素前一位置/队尾元素位置

特殊运算:doubleSpace

基本运算实现及其(最好/最差/平均)复杂度

空队列/满队列判定条件, 队列长度的计算

4.3 队列的链接实现

概念及思路:链接队列,不带头尾结点的单链表,单链表起始结点为队头元素,单链表终端结点为队尾元素

存储信息:指向队头元素及队尾元素的指针

基本运算实现及其(最好/最差/平均)复杂度

4.5 队列的应用

单服务台公平排队系统的模拟

第6章 树

6.1 树的定义

概念:通用树/有根树/无序树/有序树/位置树/森林/根结点/叶结点/内部结点/结点的度/树的度/子结点/父结点/祖先结点/子孙结点/兄弟结点/结点的层次/结点的高度/树的高度

森林边数与结点数目的关系

6.2 二叉树

概念:满二叉树/完全二叉树/普通二叉树

不同形态二叉树的数目

二叉树的常用性质:每层结点数上限/给定高度二叉树结点上限/叶结点与度为 2 结点数间的关系/完全二叉树结点数与高度的关系/完全二叉树层次遍历顺序编号(0 或 1-based)方案确定父及左右孩子结点的方法

二叉树的遍历:层次/前序(先根)/后序(后根)/中序(中根)遍历

根据遍历结果确定二叉树:完全二叉树， 层次/前序/后序+中序

二叉树的实现:顺序实现(层次遍历顺序编号)， 二叉链表(标准存储结构)， 三叉链表(广义的标准存储结构)， 二叉链表的各种遍历实现(递归/非递归)， 二叉链表求结点数/高度的方法(递归/非递归)，二叉树类型(满/完全/普通)的判定

6.3 二叉树的应用：计算表达式

表达式树: 从中缀表达式构造表达式树， 表达式树求值(表达式树的后序遍历即后缀表达式)

6.4 哈夫曼树和哈夫曼编码

前缀编码及二叉树的关系,编码的代价

哈夫曼树的构造及哈夫曼编码的获得

6.5 树和森林

孩子兄弟链表示法

有序树的前序/后序/层次遍历

有序森林的前序/后序遍历

有序森林的前序遍历 vs 孩子兄弟链表示树的前序遍历

有序森林的后序遍历 vs 孩子兄弟链表示树的中序遍历

第 7 章 优先级队列

7.1 基于线性表的优先级队列

在入队时处理优先级/在出队时处理优先级

7.2 基于树的优先级队列

概念:二叉堆/最小化堆/最大化堆

思路:结构性(完全二叉树层次遍历编号规律)及有序性(父结点比孩子结点优先级高)

基本运算:创建空队/入队/出队/读队头元素/判队空

特殊运算:向上过滤/向下过滤/建堆

各种运算实现及其(最好/最差/平均)复杂度

7.6 优先级队列的应用:排队系统的模拟

多服务台公平排队系统的模拟

多服务台优先排队系统的模拟

第 8 章 集合与静态查找表

8.1 集合的定义

概念:集合元素的键值(关键字值), 集合元素的类型

8.2 查找的基本概念

查找的概念:查找表/静态查找表/动态查找表/内部查找/外部查找

8.3 静态查找表

8.4 无序表的查找

顺序查找及哨兵的使用(低位哨兵 vs 高位哨兵)

8.5 有序表的查找

二分查找:执行条件,查找成功/不成功的最多元素比较次数

插值查找:执行条件

分块查找:执行条件

第 9 章 动态查找表

基本运算:查找/插入/删除

9.1 二叉查找树

概念:二叉查找树

性质:中序遍历键值成正序(或逆序)排列

运算实现(递归及非递归): 查找/插入(新插入结点是叶结点)/删除(被删结点为叶结点/度为 1 的结点

/度为 2 的结点-替身选择)

性能: $O(h)$, 最差时为$O(n)$

9.2 AVL 树

概念:平衡因子/平衡树/AVL 树

AVL 树的性质:结点数目与高度间的关系(给定结点数求高度范围, 给定高度求结点数范围)

插入操作: 先按二叉查找树方法插入(若成功则新插入结点为叶结点), 从新插入结点开始沿父结点方向依次检查各祖先结点平衡度, 对首次失衡结点实施 LL/LR(RR+LL)/RL(LL+RR)/RR 调整操作.

删除操作: 先按二叉查找树方法删除(若成功则真实被删结点原所在子树高度降低), 从真实被删结点的父结点开始沿父结点方向依次检查各祖先结点的平衡度及高度降低情况,直至遇到高度不变结点或根结点, 对每个平衡度失衡结点实施 LL/LR/RL/RR 调整操作

性能: $O(h), O(\log n)$

9.6 散列表

概念:散列表/散列函数/冲突(碰撞)/冲突解决办法

常用散列函数: 直接定址/除留余数/数字分析/平方取中/折叠法

除留余数法中模数的选择: 不超过存储数组规模的最

大素数

冲突解决办法： 闭散列表(线性探测法/二次探测法/再散列法及延迟删除)，开散列表

性能：平均$O(1)$

第 10 章 排序

10.1 排序的基本概念

排序的概念:排序/稳定性/内排序/外排序

10.2 插入排序

直接插入/二分插入/希尔排序： 过程/稳定性/时间复杂度/输入分布对复杂度的影响

插入排序思想： 找到当前元素的插入位置后移动必要元素

10.3 选择排序

直接选择/堆排序： 过程/稳定性/时间复杂度/输入分布对复杂度的影响

直接排序思想： 找到当前最小(大)元素， 如有必要与其最终位置当前元素交换

堆排序思想：建大顶堆/出队元素并缩小堆规模

10.4 交换排序

冒泡排序/快速排序： 过程/稳定性/时间复杂度/输入分布对复杂度的影响

冒泡排序思想: 起泡使得当前位置元素是前面所有位置元素的最大值, 记录最后发生交换的位置

快速排序思想: 设[low,high]为当前轮要分置的范围,随机选定一个基准元素(如有必要与low位置元素交换),将比基准元素小的元素移到low位置,将比基准元素大的元素移到 high 位置,直至 low 与 high 相遇后将基准元素放入相遇位置

10.5 归并排序

归并排序: 过程/稳定性/时间复杂度/输入分布对复杂度的影响/空间复杂度

归并排序的思想: 利用两(多)个读取位置及一个写入位置将读取位置当前最小值拷贝到写入位置

10.6 基数排序

基数排序: 执行要求/过程/稳定性/时间复杂度/输入分布对复杂度的影响/空间复杂度

基数排序思想: 从低位向高位依次按当前位的顺序入口袋及出口袋并缝合

第 11 章 外部查找与排序

11.1 主存储器与外存储器

主存储器(内存)与外存储器(外存)各自的优缺点

11.2 B 树

概念：B 树

记录(关键字)数目与高度间的关系(给定记录数求高度范围，给定高度求记录数范围)

B 树的查找

B 树的插入：找到插入位置(叶结点的上一层结点)并插入，若插入成功，从插入结点开始沿着父结点方向检查结点关键字数目是否超标：若不超标则结束,否则分裂超标结点并将分裂产生的分割记录插入到超标结点的父结点中并检查父结点的超标情况

B 树的删除：找到删除元素并删除该元素或其替身，若删除成功，从真实被删结点开始沿着父结点方向检查关键字数是否过低：若不过低则结束；若过低且兄弟有记录可借则借(移动所借记录及父结点中的分割记录)；若过低且兄弟无记录可借则与兄弟结点及父结点分割记录合并，同时将分割记录从父结点中删除并检查父结点的关键字数

参数m的确定：根据分块/关键字/记录/地址大小信息确定最佳参数m

11.3 B+树

概念：B+树

B 树与 B+树的关系：B+树可解决 B 树快速按序全列记

录的问题； B+树的索引部分是一棵只存关键字不存其余记录数据的特殊 B 树； B+树的叶结点保存含关键字在内的记录数据并形成单链表

记录(关键字)数目与索引树高度间的关系(给定记录数求高度范围，给定高度求记录数范围)

B+树的查找

B+树的插入： 找到插入位置(叶结点)并插入， 若插入成功则检查叶结点关键字数目是否超标： 若不超标则结束,否则分裂叶结点并将新叶结点的最小关键字插入到索引树相应结点中， 之后按索引树(B 树)插入方法继续后续操作

B+树的删除： 若能找到删除位置(叶结点)则删除该元素,若在查找被删元素时在索引树中找到了被删关键字， 则先将被删叶结点删除前的第 2 小关键字替换到索引树中找到的被删关键字. 检查被删叶结点关键字数是否过低： 若不过低则结束； 若过低且兄弟有记录可借则借(移动所借记录， 更新索引树父结点中的分割关键字)； 若过低且兄弟无记录可借则与兄弟结点合并， 同时将分割关键字从索引树父结点中删除并按索引树(B 树)删除方法继续后续操作

参数m,l的确定： 根据分块/关键字/记录/地址大小信

息确定最佳参数m,l

11.4 外排序

概念：外排序/置换选择/多阶段归并

置换选择生成一个有序片段的过程： 建最小化堆/出队元素并输出到输出缓存/从输入缓存读下一记录/根据下一记录关键字与当前出队关键字大小进行堆规模调整(保持不变或减 1)…直至输入缓存无法读下一记录或堆规模变成 0

(2 路)多阶段归并： 在置换选择阶段按斐波那契数列顺序生成有序片段$(\rightarrow F_0;\rightarrow F_1;\cdots F_{i-1}\rightarrow F_{i+1}\cdots)$，执行首次归并拷贝轮使得磁带有序片段数分别为$F_i,F_{i+1}$，依斐波那契数列执行归并

第 13 章 图

13.1 图的定义

概念： 图/顶(结)点集/边集/无向边/有向边/无向图/有向图/边的权值(代价)/加权图/简单图/完全图/邻接/度/入度/出度/子图/路径/简单路径/回路(环)/路径长度/连通/有向无环图(DAG)/连通图/连通分量/强连通图/强连通分量/弱连通分量/生成树

图的通用操作：插入边/删除边/查找边/顶点数/边数

图中结点度与边数的关系

13.2 图的存储

邻接矩阵: 存储结点值的数组/存储边的邻接矩阵/表示无边的特殊值(noEdge); 邻接矩阵保存的图的通用操作及获得结点度/入度/出度的方法, 相应的时间复杂度

邻接表: 存储结点值及边链表的数组; 邻接表保存的图的通用操作及获得结点度/入度/出度的方法, 相应的时间复杂度

13.3 图的遍历

深度优先搜索(DFS)/深度优先生成森林

广度优先搜索(BFS)/广度优先生成森林

13.4 图的遍历的应用

无向图的连通性: 深度/广度优先生成森林的每棵树对应无向图的一个连通分量

有向图的强连通性:

1)对有向图执行深度优先搜索获得对应生成森林

2)对 1)中森林结点按后续遍历顺序从小到大编号

3)按 2)中编号从大到小顺序对有向图的逆图执行深度优先搜索获得对应生成森林, 则此森林的每棵树对应有向图的一个强连通分量

欧拉回路:

连通无向图存在欧拉回路的充要条件：没有度为奇数的顶点

连通无向图存在欧拉路径的充要条件：度为奇数的顶点数不超过 2

笔画问题：若连通无向图中度为奇数的顶点数为$2n$,则最少可通过n笔在平面中画出该无向图

解决思路：设连通无向图中度为奇数的顶点数为$2n$：使用n条虚拟边分别连接原图中顶点为奇数的顶点；从任一顶点出发沿着连接该顶点还未走过的边走到其邻接顶点，持续此过程直到走到的顶点所有连接边已被走过(最后的顶点一定是此轮的始发顶点)；若当前行走路径中还有顶点有连接边未走过，从该顶点出发沿着连接该顶点还未走过的边走到其邻接顶点，持续此过程直到走到的顶点所有连接边已被走过(最后的顶点一定是此轮的始发顶点)…持续本轮操作直到当前行走路径中没有顶点有连接边未走过；将当前行走路径中的虚拟边删除后得到的路径段即为相应的笔画

拓扑排序：

概念:拓扑排序/AOV 网

有向图存在拓扑排序的充要条件：有向图是无环图

拓扑排序/DAG 判定的求解算法：依次将图中入度为 0 的结点删除并输出，直到图中没有入度为 0 的结点或剩下空图. 若最后剩下的是空图则原图是 DAG 且按序输出的结果为其一个拓扑排序；若最后剩下的图不是空图，则原图不是 DAG 且剩下的图恰对应原图中的环上顶点集

会写拓扑排序/DAG 判定的实现代码

关键路径:

概念:关键路径/AOE 网/关键事件/关键活动/源点(起点)/汇点(收点)

有向图(设源点入度及汇点出度均为 0)存在关键路径的充要条件：能到达汇点的顶点集 W 是源点能到达的顶点集 U 的子集，且有向图的恰含顶点集 W 的最大子图是 DAG

关键路径求解算法框架：对恰含顶点集 W 的最大子图执行操作：初始所有顶点的最早完成时间为 0，按任一拓扑排序顺序确定顶点的最早完成时间并更新当前顶点的邻接到顶点的最早完成时间，初始所有顶点的最晚完成时间为汇点的最早完成时间，按之前拓扑排序的逆序确定顶点的最晚完成时间并更新当前顶点的邻接顶点的最晚完成时间，根据顶点的完成时间获得所

有边的最早及最晚启动时间，确定关键顶点及关键活动