|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Ch7&8** | | | | | | | |
| 班级 | 01 | 学号 | 2022040906023 | 姓名 | 梁书恺 | 成绩 |  |
| 一、选择题（每空5分，共50分）  1. 以下（ C ）方法适合动态查找。  A 顺序查找 B 折半查找 C 散列查找 D 随机查找  2. 在以下数据结构中，（ C ）查找效率最低。  A 有序顺序表 B 二叉排序树 C 单链表 D 散列表  3. 假定查找成功与不成功的可能性相同，在查找成功的情况下每个记录的查找概率相同，则顺序查找的平均查找长度为（ D ）。  A 0.5(*n*+1) B 0.25(*n*+1) C 0.5(*n*-1) D 0.75*n*+0.25  4. 对100个元素进行折半查找，在查找成功的情况下，比较次数最多是（ D ）。  A 25 B 50 C 10 D 7  5. 二叉排序树中，最小值结点的（ A ）。  A 左指针一定为空 B 右指针一定为空  C 左、右指针均为空 D 左、右指针均不为空  6. 下述排序方法中，时间性能与待排序记录的初始状态无关的是（ C ）。  A插入排序和快速排序 B归并排序和快速排序  C选择排序和归并排序 D插入排序和归并排序  7. 设要将序列（Q，H，C，Y，P，A，M，S，R，D，F，X）的ASCII码按升序排列，则（ D ）是起泡排序一趟扫描的结果，（ E ）二路归并排序一趟扫描的结果，（ A ）是以第一个元素为基准的快速排序一趟扫描的结果。  A（F，H，C，D，P，A，M，Q，R，S，Y，X）  B（P，A，C，S，Q，D，F，X，R，H，M，Y）  C（A，D，C，R，F，Q，M，S，Y，P，H，X）  D（H，C，Q，P，A，M，S，R，D，F，X，Y）  E（H，Q，C，Y，A，P，M，S，D，R，F，X）  8. （ A ）在某趟排序结束后不一定能选出一个元素放到其最终位置上。  A 选择排序 B 起泡排序 C 归并排序 D 快速排序  二、简答题（共30分）  1. **假定对有序表：（3，4，5，7，24，30，42，54，63，72，87，95）进行折半查找，回答下列问题：**  **① 画出描述折半查找过程的判定树；**  **② 若查找元素54，需依次与哪些元素比较？**  **③ 若查找元素90，需依次与哪些元素比较？**  **④ 假定每个元素的查找概率相等，求查找成功时的平均查找长度。**    ② 需要与30、63、42、54比较  ③ 需要与30、63、87、95比较  ④ 平均查找长度为：  **2. 已知如下长度为12的表：（Jan, Feb, Mar, Apr, May, June, July, Aug, Sep, Oct, Nov, Dec）,元素按字典顺序排序。**  **① 试按表中元素的顺序依次插入一棵初始为空的二叉排序树，画出插入完成之后的二叉排序树，并求其在等概率的情况下查找成功的平均查找长度。**  **② 若Hash(x)=（x的字母ASCII码之和）%12，分别求出以拉链法和二次探测法得到的散列表。求在等概率的情况下查找成功的平均查找长度。**  **①**    平均查找长度为：  **②**      平均查找长度为  **3. 对*n*=10，给出快速排序一个最好情况和最坏情况的初始排列的实例，说明比较次数可能达到的最大值和最小值分别是多少？**  最好情况：4 1 0 2 3 7 5 6 8 9  最坏情况：9 8 7 6 5 4 3 2 1 0  比较次数最大值为45次  比较次数最小值为19次  三、算法设计（共20分）  N个元素用单链表存储，每个结点包括元素值、后继指针、查找频率；查找频率初始化为1，以后元素每查找一次，其频率加一。编写算法，使得查找频率高的元素能更快找到，从而提高总查找效率。  void InsertList(List \*L, int x)  {  Node \*p = L->head->next;  Node \*q = L->head;  while (p != L->tail)  {  if (p->data == x)  {  p->freq++;  if (p->freq > q->freq)  {  q->next = p->next;  p->next = L->head->next;  L->head->next = p;  }  return;  }  q = p;  p = p->next;  }  Node \*s = (Node \*)malloc(sizeof(Node));  s->data = x;  s->freq = 1;  s->next = L->head->next;  L->head->next = s;  } | | | | | | | |
|  | | | | | | | |