数据结构与算法 算法题 例题

李明哲　计算学部金牌讲师团

仅供参考，实际答题请以考试题目要求为准。

前言

算法题一般有三个方向：线性表、树、图。基本都是在一个设定的背景或问题下，通过使用常见的算法（排序算法、查找算法、双指针、树的遍历、DFS、BFS）来解题。其中树和图的遍历需要掌握递归和非递归两种形式。考试时需要根据题目要求作答，每道题一般会以小问形式分为三部分：

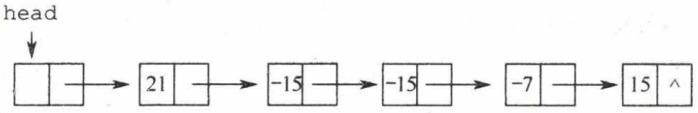
1. 算法的基本设计思想
2. 算法的代码描述
3. 时间和空间复杂度

其中算法的代码描述部分最为重要，但在因种种原因写不出代码时一定要把算法的基本设计思想部分写清楚，这样也可以获得一定的分数。

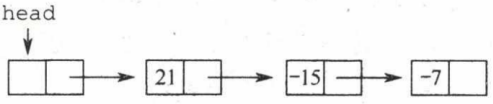
算法题经常会要求你"设计一个尽可能高效的算法"，这时需要时间复杂度和空间复杂度都优；也可能只要求你"设计一个时间复杂度尽可能高效的算法"（如下面给出的考研题），这时只需考虑时间复杂度即可，可以尝试用"时间换空间"。

有些题目有额外要求，作答时以题目描述为准，例如2015年考研题还要求给出单链表结点的数据类型定义：

24. 【2015 统考真题】用单链表保存 m 个整数，结点的结构为 [data][link]，且|data|≤n（n 为正整数）。现要求设计一个时间复杂度尽可能高效的算法，对于链表中 data 的绝对值相等的结点，仅保留第一次出现的结点而删除其余绝对值相等的结点。例如，若给定的单链表 head 如下：



则删除结点后的 head 为



要求：

1）给出算法的基本设计思想。

2）使用 C 或 C++语言，给出单链表结点的数据类型定义。

3）根据设计思想，采用 C 或 C++语言描述算法，关键之处给出注释。

4）说明你所设计算法的时间复杂度和空间复杂度。

24.【解答】

1）算法的基本设计思想：

- 算法的核心思想是用空间换时间。使用辅助数组记录链表中已出现的数值，从而只需对链表进行一趟扫描。
- 因为|data|≤n，故辅助数组 q 的大小为 $n+1$，各元素的初值均为 0。依次扫描链表中的各结点，同时检查 q[|data|]的值，若为 0 则保留该结点，并令 q[|data|]=1；否则将该结点从链表中删除。

2）使用 C 语言描述的单链表结点的数据类型定义：

```
typedef struct node {
   int           data;
   struct node   *link;
}NODE;
Typedef NODE *PNODE;
```

3）算法实现如下：

```
void func (PNODE h,int n){
    PNODE p=h,r;
    int *q,m;
    q=(int *)malloc(sizeof(int)*(n+1));  //申请 n+1 个位置的辅助空间
    for(int i=0;i<n+1;i++)               //数组元素初值置 0
       *(q+i)=0;
    while(p->link!=NULL){
       m=p->link->data>0? p->link->data:-p->link->data;
       if(*(q+m)==0){                    //判断该结点的 data 是否已出现过
          *(q+m)=1;                      //首次出现
          p=p->link;                     //保留
       }
       else{                             //重复出现
          r=p->link;                     //删除
          p->link=r->link;
          free(r);
       }
    }
    free(q);
}
```

4）参考答案所给算法的时间复杂度为 $O(m)$，空间复杂度为 $O(n)$。

下面我出了一道有关图的例题，难度为期末考试算法题第三题难度，供参考。

例题

H 市冬季旅游火爆，你的好朋友小明喜爱冰雪，也想前往 H 市旅游，因此他在网络上找到了一份特殊的H市旅游地图。在这份特殊的地图中，不同景点间的旅游路径都是单向的，即对于任意的两个景点 A 和 B，若能从 A 旅游至 B，则一定不能从 B 旅游至 A。现在小明急切地向你求助：他想在这份旅游地图中找到一个景点 S，使得他从 S 出发可以游览这份地图中标注的任一景点。现在小明把这份地图交给你，请你设计一个算法来帮助他找到这个景点 S。

1）给出算法的基本设计思想。

2）根据设计思想，采用 C 或 C++ 语言描述算法，关键之处给出注释。

3）说明你所设计算法的时间复杂度和空间复杂度。

解析

问题转化：

将这张“旅游地图”构造成数据结构中的图，其中图中的顶点是旅游地图中的景点，图中的边是旅游地图中的路径。题目中对这份旅游地图给出了“不同景点间的旅游路径都是单向的”这一限制，因此我们构造的图是一个有向无环图。

即问题转化为：**有向无环图中是否存在一点可以通达图中其他任一结点。**

思路简析：

首先，需要注意的是可能不存在满足条件的顶点。

接着，若存在满足条件的顶点，则一定唯一，否则若存在两个不同的顶点 v_1, v_2 满足上述条件，那么就存在一条从 v_1 到 v_2 的路和一条从 v_2 到 v_1 的路，显然图中出现了环，与题意不符。

然后，我们开始分析这个问题。根据“有向无环图”我们可以联想到拓扑排序，因为我们知道拓扑排序是一个有向无环图（DAG）的所有顶点的线性序列。进一步分析拓扑排序和这道题目的关系，我们发现拓扑排序中的第一个顶点就是我们要求的顶点。

最后，根据拓扑排序的思想，我们知道这个顶点满足的条件是“入度为0”。

参考答案：

答：

（1）将这张“旅游地图”构造成数据结构中的图，记为图 $G=(V,E)$。G 中的顶点是旅游地图中的景点，G 中的边是旅游地图中的路径，G 是一个有向无环图。读取 G 中所有的边，计算 G 中顶点的入度。若仅有一个入度为 0 的顶点，记该顶点为 v，则 v 就是题中所求的旅游景点 S。

（2）

```
int FindStart(Graph G)  // 图G中顶点以序号表示，范围为[0, G.numVertices - 1]
{
    int degree[G.numVertices] = {0};                    // 记录入度
    for (int i = 0; i < G.numEdges; i++) {
        degree[G.Edge[i][1]]++;
    }

    start = -1;
    for (int i = 0; i < G.numVertices; i++) {
        if (degree[i] == 0) {
            if (start != -1) {                          // 有两个以上顶点入度为0
                start = -1;
                break;
            }
            start = i;
        }
    }
    return start;                                       // -1表示不存在满足条件的顶点
```

```
}
```

(3) 时间复杂度：$O(max\{m,n\})$，空间复杂度：$O(n)$，其中 n 为图中顶点数，m 为图中边数。