线性表

线性表的顺序表示

07. 已知在一维数组 $A[m+n]$ 中依次存放两个线性表 $(a_1, a_2, a_3, \cdots, a_m)$ 和 $(b_1, b_2, b_3, \cdots, b_n)$。编写一个函数，将数组中两个顺序表的位置互换，即将 $(b_1, b_2, b_3, \cdots, b_n)$ 放在 $(a_1, a_2, a_3, \cdots, a_m)$ 的前面。

09. 给定三个序列 A、B、C，长度均为 n，且均为无重复元素的递增序列，请设计一个时间上尽可能高效的算法，逐行输出同时存在于这三个序列中的所有元素。例如，数组 A 为 $\{1, 2, 3\}$，数组 B 为 $\{2, 3, 4\}$，数组 C 为 $\{-1, 0, 2\}$，则输出 2。要求：

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）说明你的算法的时间复杂度和空间复杂度。

```
int hash[0xFFFF];
void findRepeated(int A[], int B[], int C[], int n){
    for(int i = 0; i < n; i++){
        hash[A[i]]++;
        hash[B[i]]++;
        hash[C[i]]++;
    }
    for(int i = 0; i < n; i++){
        if(hash[A[i] > 1])
            print(A[i]);
    }
    for(int i = 0; i < n; i++){
        if(hash[B[i] > 1])
            print(B[i]);
    }
    for(int i = 0; i < n; i++){
        if(hash[C[i] > 1])
            print(C[i]);
    }
}
```

```
void findRepeated(int A[], int B[], int C[], int n){
    int i = 0, j = 0, k = 0;
    while(i < n && j < n && k < n){
        if(A[i] == B[j] && B[j] == C[k]){
            print(A[i]);
            i++; j++; k++;
        }
        else{
            i++;
        }
    }
}
```

12. 【2013 统考真题】已知一个整数序列 $A=(a_0, a_1, \cdots, a_{n-1})$，其中 $0 \leqslant a_i < n$（$0 \leqslant i < n$）。若存在 $a_{p1}=a_{p2}=\cdots=a_{pm}=x$ 且 $m>n/2$（$0 \leqslant p_k < n$，$1 \leqslant k \leqslant m$），则称 x 为 A 的主元素。例如 $A=(0, 5, 5, 3, 5, 7, 5, 5)$，则 5 为主元素；又如 $A=(0, 5, 5, 3, 5, 1, 5, 7)$，则 A 中没有主元素。假设 A 中的 n 个元素保存在一个一维数组中，请设计一个尽可能高效的算法，找出 A 的主元素。若存在主元素，则输出该元素；否则输出 -1。要求：

1）给出算法的基本设计思想。

2）根据设计思想，采用 C 或 C++或 Java 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。

```
void findMajor(int arr[], int n){
    int *hash = (int*)malloc(sizeof(int)*(n-1));
    for(int i = 0; i < n; i ++){
        hash[arr[i]] ++;
        if(hash[arr[i]] > n/2){
            print(arr[i]);
            return;
        }
    }
    print(-1);
    return;
}
```

O(n) O(n)

最优法：O(n) O(1)

13. 【2018 统考真题】给定一个含 n（$n \geqslant 1$）个整数的数组，请设计一个在时间上尽可能高效的算法，找出数组中未出现的最小正整数。例如，数组 $\{-5, 3, 2, 3\}$ 中未出现的最小正整数是 1；数组 $\{1, 2, 3\}$ 中未出现的最小正整数是 4。要求：

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。

MissMin的值一定在1到1+n之间

```
int hash[0xFFFF];
void findMissMin(int arr[], int n){
    int max = 0;
    for(int i = 0; i < n; i++){
        if(arr[i] > max)
            max = arr[i];
        if(arr[i] >= 0)
            hash[arr[i]]++;
    }
    for(int i = 1; i <= max; i++){
        if(hash[i] == 0){
            print(i);
            return;
        }
```

```
    }
}
```

O(n) O(1)

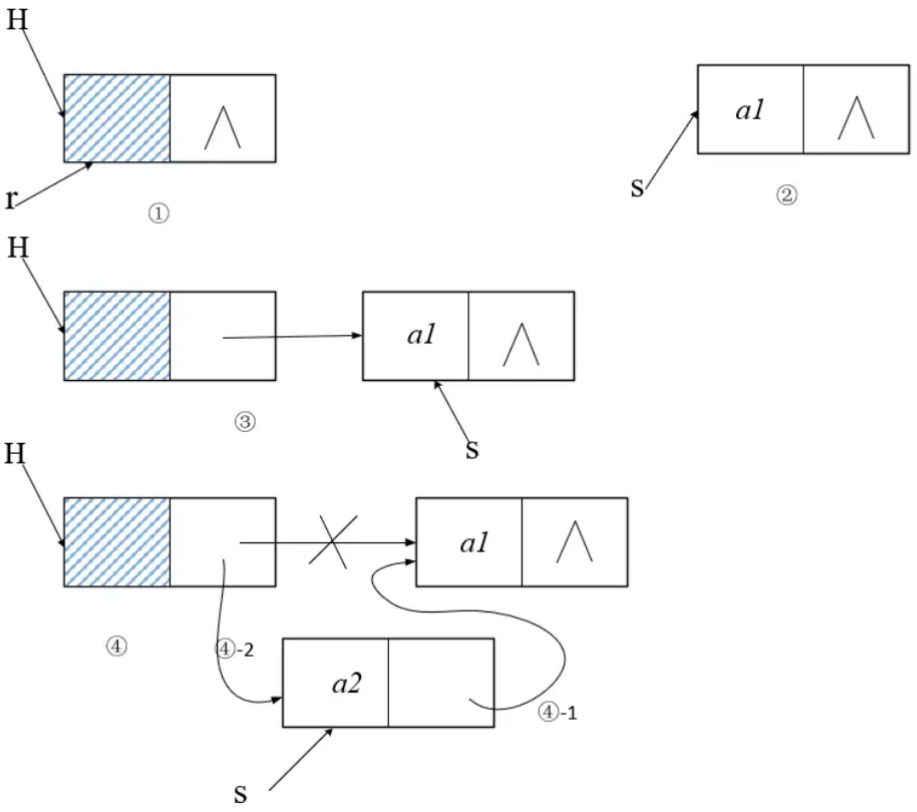
线性表的链式表示

① 头插法

该算法的官方描述为：从一个空表开始，重复读入数据，生成新结点，将读入数据存放到新结点的数据域中，然后将新结点插入到当前链表的表头结点之后。

这里的重点就是：每次生成的新结点都是要与头结点相连接的，每个新结点都插在了原来第一个节点的前面。通过这种方法建立的链表是后来居前的，也就是链表是逆序的。因此，当有题目让我们实现线性表的逆序表示，就应该首先考虑头插法。

图示为：



01. 在带头结点的单链表 L 中，删除所有值为 x 的结点，并释放其空间，假设值为 x 的结点不唯一，试编写算法以实现上述操作。

```
void del(Node* head, int x){
    Node* ptr = head->next;
    Node* pre = head;
    Node* q;
    while(ptr!=NULL){
        if(ptr->data == x){
            q = ptr;
            pre->next = q->next;
            ptr = q->next;
            free(q);
```

```
        }
        else{
            ptr = ptr->next;
            pre = pre->next;
        }
    }
}
```

02. 试编写在带头结点的单链表 L 中删除一个最小值结点的高效算法（假设该结点唯一）。

```
void delMin(Node* head){
    Node* ptr = head->next;
    Node* pre = head;
    Node* minPtr = head->next;
    Node* minPre;
    while(ptr!=NULL){
        if(ptr->data < minPtr->data){
            minPtr = ptr;
            minPre = pre;
        }
        ptr = ptr->next;
        pre = pre->next;
    }
    minPre->next = minPtr->next;
    free(minPtr);
}
```

03. 试编写算法将带头结点的单链表就地逆置，所谓“就地”是指辅助空间复杂度为 $O(1)$。

```
// 头插法
void reverse(Node* head){
    Node* ptr = head->next;
    Node* temp;
    Node* tail = head;
    while(ptr!=NULL){
        temp = ptr;
        head->next = temp;
        if(tail == head){
            temp->next = NULL;
            tail = temp;
        }
        else{
            temp->next = tail;
            tail = temp;
        }
    }
}
```

04. 设在一个带表头结点的单链表中，所有结点的元素值无序，试编写一个函数，删除表中所有介于给定的两个值（作为函数参数给出）之间的元素（若存在）。

```

```

05. 给定两个单链表，试分析找出两个链表的公共结点的思想（不用写代码）。

两个单链表有公共结点，即两个链表从某一结点开始，它们的 next 都指向同一结点。由于每个单链表结点只有一个 next 域，因此从第一个公共结点开始，之后的所有结点都是重合的，不可能再出现分叉。所以两个有公共结点而部分重合的单链表，拓扑形状看起来像 Y，而不可能像 X。

本题极容易联想到“蛮”方法：在第一个链表上顺序遍历每个结点，每遍历一个结点，在第二个链表上顺序遍历所有结点，若找到两个相同的结点，则找到了它们的公共结点。显然，该算法的时间复杂度为 $O(\text{len1}\times\text{len2})$。

06. 设 $C=\{a_1,b_1,a_2,b_2,\cdots,a_n,b_n\}$ 为线性表，采用带头结点的单链表存放，设计一个就地算法，将其拆分为两个线性表，使得 $A=\{a_1,a_2,\cdots,a_n\}$，$B=\{b_n,\cdots,b_2,b_1\}$。

```
void disJoin(Node* head){
    Node* head1 = head; Node* head2 = head->next;
    Node* tail1 = head; Node* tail2 = head->next;
    Node* ptr = head2->next;
    int flag = 1;
    while(ptr!=NULL){
        if(flag == 1){
            tail1->next = ptr;
            tail1 = ptr;
        }
        if(flag == -1){
            tail2->next = ptr;
            tail2 = ptr;
        }
        flag *= -1;
        ptr = ptr->next;
    }
    tail1->next = NULL;
    tail2->next = NULL;
}
```

栈

03. 栈的初态和终态均为空，以 I 和 O 分别表示入栈和出栈，则出入栈的操作序列可表示为由 I 和 O 组成的序列，可以操作的序列称为合法序列，否则称为非法序列。

1）下面所示的序列中哪些是合法的？

A. IOIIOIOO　　B. IOOIOIIO　　C. IIIOIOIO　　D. IIIOOIOO

2）通过对 1）的分析，写出一个算法，判定所给的操作序列是否合法。若合法，返回 true，否则返回 false（假定被判定的操作序列已存入一维数组中）。

1. AD 确保最终栈是空的

```
int judge(char opr[]){
    int i, j = 0;
    while(opr[i]!='\0'){
        if(opr[i] == 'I')
            j++;
        else{
            j--;
            if(j<0){
                printf("X_X...");
```

```
                return 0;
            }
        }
    }
    if(j!=0){
        printf("x_X...");
        return 0;
    }
    return 1;
}
```

04. 设单链表的表头指针为 L，结点结构由 data 和 next 两个域构成，其中 data 域为字符型。试设计算法判断该链表的全部 n 个字符是否中心对称。例如 xyx、xyyx 都是中心对称。

```
int isMiddleConjunct(Node* L, int n){
    int indx = 1;
    Node* ptr = L;
    char s[n/2];
    if(n%2 == 1){
        // xxyxx
        while(indx <= n/2){
            ptr = ptr->next;
        }

    }
}
```