```c
#include <stdio.h>

#define MAXSIZE 20

typedef enum {
    false = 1,
    true = 0
} boolean;

typedef int ElemenType;     //数组的最大容量
typedef struct {
//    ElemenType *data;
    ElemenType data[MAXSIZE];
    int length;     //当前元素的数量
}Sqlist;

void InitList(Sqlist *L) {
    L->length = 0;
//    L->data = (ElemenType *)malloc(MAXSIZE * sizeof(ElemenType));
//    L->data[MAXSIZE];
}

/**
 * 顺序输入
 * @param list
 * @param e
 * @return
 */
boolean ArraysInputList(Sqlist *list) {
    int e;
    if (list->length == MAXSIZE) {
        puts("list已满! ");
    }
    for (int i = 0; i < 4; i++) {
        printf("输入元素");
        scanf("%d", &e);
        list->data[i] = e;
        list->length++;
    }
    return true;
}

/**
 * 按位置输入
 * @param list
 * @param e
 * @param i
 * @return
 */
boolean addElemtnt(Sqlist *list, int e, int i) {
    int k;
    if (list->length == MAXSIZE) {
        puts("list已满");
        return false;
    }
```

```
    if (i < 1 || i > list->length + 1) {
        puts("i的值超出了list顺序表的最大长度！");
        return false;
    }
    if (i <= list->length) {
        for (k = list->length - 1; k >= i - 1; k--) {
            list->data[k + 1] = list->data[k];
        }
    }

    list->data[i - 1] = e;
    list->length++;
    return true;

}

/**
 * 遍历
 * @param list
 * @return
 */
boolean ViewList(Sqlist list) {

    //从后往前
//    while (list.length != 0) {
//        printf("%d", list.data[list.length - 1]);
//        list.length--;
//    }
//    puts("\n");


//    front --> rear
    for (int i = 0; i < list.length; i ++)
    {
        printf("%d", list.data[i]);
    }

    puts("\n");
    return true;
}

/**
 * 全部删除
 * @param sqlist
 * @return
 */
boolean DeleteListEle(Sqlist *sqlist) {
    sqlist->length = 0;
    return true;
}

/**
 * 按元素位置删除
 * @param sqlist
 * @param i
 * @return
 */
boolean DeletePostListEle(Sqlist *sqlist, int post) {
```

```
    if (sqlist->length == 0) {
        puts("list为空");
        return false;
    }
    if (post > sqlist->length) {
        puts("post大于length");
        return false;
    }

//    for (int j = post - 1; j < sqlist->length - post - 1; j++) {
//        sqlist->data[j - 1] = sqlist->data[j];
//    }

        for (int j = post - 1; j < sqlist->length - post + 1; j ++)
        {
            sqlist->data[j] = sqlist->data[j + 1];
        }

    sqlist->length --;
    return true;
}

/**
 * 按位置查找
 * @param list
 * @param post
 * @return
 */
boolean FindListPostEle(Sqlist list, int post)
{
    if (list.length == 0)
    {
        puts("list为空! ");
        return false;
    }
    if (post > list.length) {
        puts("post大于length");
        return false;
    }
    for (int i = 0; i < list.length; i ++)
    {
        if (i == post)
        {
            printf("post位置的元素为: %d\n", list.data[i - 1]);
        }
    }

    return true;
}

/**
 * 以元素查找
 * @param sqlist
 * @param e
 * @return
 */
boolean FindListELe(Sqlist sqlist, int e)
{
```

```
172
173     if (sqlist.length == 0)
174     {
175         puts("list为空! \n");
176         return false;
177     }
178     for (int i = 0; i < sqlist.length; i ++)
179     {
180         if (e == sqlist.data[i])
181         {
182             printf("元素位置为: %d\n", i + 1);
183             return true;
184         }
185     }
186     return true;
187 }
188
189 int main() {
190
191     Sqlist list;
192
193     InitList(&list);
194     puts("list顺序表初始化成功! ");
195
196     // 插入数据
197     ArraysInputList(&list);
198
199     int post;
200     printf("插入数据的位置\n");
201     scanf("%d", &post);
202     //e为要插入的元素
203     addElemtnt(&list, 6, post);
204
205     puts("遍历元素输出: \n");
206     ViewList(list);
207
208     int post2;
209     puts("请输入要查找元素的位置\n");
210     scanf("%d", &post2);
211     FindListPostEle(list, post2);
212
213     int e;
214     puts("请输入要查找的元素: ");
215     scanf("%d", &e);
216     FindListELe(list, e);
217
218     puts("遍历元素输出: \n");
219     ViewList(list);
220
221     int post1;
222     puts("输入删除的位置\n");
223     scanf("%d", &post1);
224     DeletePostListEle(&list, post);
225
226     puts("遍历元素输出: \n");
227     ViewList(list);
228
229     puts("删除所有元素\n");
```

```
    DeleteListEle(&list);

    puts("遍历元素输出：\n");
    ViewList(list);

    return 0;
}
```