

# 线性表

同学们要自己动手，锻炼动手能力!

## 顺序表

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 typedef struct vector {
5     int *data;
6     int size, cap;
7 } vector;
8
9 vector *init(int cnt) {
10     vector *p = (vector *)malloc(sizeof(vector));
11     p->data = (int *)malloc(sizeof(int) * cnt);
12     p->size = 0;
13     p->cap = cnt;
14     return p;
15 }
16
17 void delete_vector(vector *p) {
18     free(p->data);
19     free(p);
20 }
21
22 void show_vector(vector *v) {
23     printf("----- size = %d, cap = %d-----\n", v->size, v->cap);
24     for (int i = 0; i < v->size; i++) {
25         printf("%d ", v->data[i]);
26     }
27     printf("\n-----\n");
28 }
29
30 int insert_ele(vector *v, int ind, int val) {
31     if (ind > v->size) {
32         return 1;
33     }
34     if (v->size == v->cap) {
35         v->cap *= 2;
```

```

36     v->data = (int *)realloc(v->data, sizeof(int) * v->cap);
37 }
38 for (int i = v->size; i > ind; i--) {
39     v->data[i] = v->data[i - 1];
40 }
41 v->data[ind] = val;
42 v->size++;
43 return 0;
44 }
45
46 int delete_ele(vector *v, int ind) {
47     if (v->size <= ind) {
48         return 1;
49     }
50     for (int i = ind; i < v->size - 1; i++) {
51         v->data[i] = v->data[i + 1];
52     }
53     v->size--;
54     return 0;
55 }
56
57 int main() {
58     int n, cnt;
59     scanf("%d%d", &n, &cnt);
60     vector *v = init(cnt);
61     for (int i = 0; i < n; i++) {
62         int a, b;
63         scanf("%d", &a);
64         if (a == 0) {
65             scanf("%d%d", &a, &b);
66             insert_ele(v, a, b);
67         } else if (a == 1) {
68             scanf("%d", &a);
69             delete_ele(v, a);
70         }
71         show_vector(v);
72     }
73     delete_vector(v);
74     v = NULL;
75     return 0;
76 }

```

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 typedef struct node {
5     int data;
6     struct node *next;
7 } node;
8
9 typedef struct list {
10     int size;
11     struct node *head;
12 } list;
13
14 node *get_new_node(int val) {
15     node *p = (node *)malloc(sizeof(node));
16     p->data = val;
17     p->next = NULL;
18     return p;
19 }
20
21 void delete_list(list *p) {
22     node *q = p->head;
23     for (int i = 0; i <= p->size; i++) {
24         node *t = q->next;
25         free(q);
26         q = t;
27     }
28     free(p);
29 }
30
31 list *init() {
32     list *p = (list *)malloc(sizeof(list));
33     p->head = get_new_node(0);
34     p->size = 0;
35     return p;
36 }
37
38 void show_list(list *l) {
39     printf("----- size = %d -----\\n", l->size);
40     for (node *p = l->head->next; p != NULL; p = p->next) {
41         printf("%d->", p->data);
42     }
43     printf("NULL\\n-----\\n");
44 }
```

```
45
46 int insert_ele(list *l, int ind, int val) {
47     if (ind > l->size) {
48         return 1;
49     }
50     node *p = l->head;
51     for (int i = 0; i < ind; i++) {
52         p = p->next;
53     }
54     node *q = get_new_node(val);
55     q->next = p->next;
56     p->next = q;
57     l->size++;
58     return 0;
59 }
60
61 int delete_ele(list *l, int ind) {
62     if (l->size <= ind) {
63         return 1;
64     }
65     node *p = l->head;
66     for (int i = 0; i < ind; i++) {
67         p = p->next;
68     }
69     node *q = p->next;
70     p->next = q->next;
71     free(q);
72     l->size--;
73     return 0;
74 }
75
76 int main() {
77     int n;
78     scanf("%d", &n);
79     list *l = init();
80     for (int i = 0; i < n; i++) {
81         int a, b;
82         scanf("%d", &a);
83         if (a == 0) {
84             scanf("%d%d", &a, &b);
85             insert_ele(l, a, b);
86         } else if (a == 1) {
87             scanf("%d", &a);
88             delete_ele(l, a);
89         }
90     }
```

```
89     }
90     show_list(l);
91 }
92 delete_list(l);
93 l = NULL;
94 return 0;
95 }
```