## 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>
4 typedef struct vector {
      int *data;
       int size, cap;
 7 } vector;
 8
9 vector *init(int cnt) {
      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);
       for (int i = 0; i < v->size; i++) {
24
           printf("%d ", v->data[i]);
25
26
27
      printf("\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) {
           v->cap *= 2;
```

```
v->data = (int *)realloc(v->data, sizeof(int) * v->cap);
36
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) {</pre>
48
           return 1;
49
       for (int i = ind; i < v->size - 1; i++) {
50
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) {
               scanf("%d%d", &a, &b);
65
66
                insert_ele(v, a, b);
67
           } else if (a == 1) {
               scanf("%d", &a);
68
69
               delete_ele(v, a);
70
71
           show_vector(v);
72
73
       delete vector(v);
74
       v = NULL;
75
       return 0;
76 }
```

## 单链表

```
1 #include <stdio.h>
 2 #include <stdlib.h>
 3
 4 typedef struct node {
      int data;
 6
      struct node *next;
 7 } node;
9 typedef struct list {
     int size;
10
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;
      for (int i = 0; i <= p->size; i++) {
23
24
           node *t = q->next;
25
          free(q);
26
           q = t;
27
28
      free(p);
29 }
30
31 list *init() {
      list *p = (list *)malloc(sizeof(list));
32
      p->head = get_new_node(0);
33
34
      p->size = 0;
35
      return p;
36 }
37
38 void show list(list *l) {
      printf("---- size = %d ----\n", l->size);
39
      for (node *p = l->head->next; p != NULL; p = p->next) {
40
           printf("%d->", p->data);
41
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;
       for (int i = 0; i < ind; i++) {
51
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) {</pre>
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;
       scanf("%d", &n);
78
       list *l = init();
79
       for (int i = 0; i < n; i++) {
80
81
           int a, b;
           scanf("%d", &a);
82
83
           if (a == 0) {
                scanf("%d%d", &a, &b);
84
85
                insert_ele(l, a, b);
86
           } else if (a == 1) {
87
                scanf("%d", &a);
88
                delete_ele(l, a);
```