

05/19/18 06:16:05 D:\git-repos\data-structure-homework\09\e25.cpp

```
1  #include <stdio>
2  #include <cstring>
3  #include <iostream>
4  #define MXN 1007
5  using namespace std;
6  typedef int eletype;
7  struct node {
8      int cnt;
9      eletype data;
10     node *prev, *nxt;
11 } buffer[MXN], *root, *tail, *tot;
12 node *newnode(eletype dta) {
13     tot->cnt = 0;
14     tot->data = dta;
15     tot->prev = tot->nxt = NULL;
16     return tot++;
17 }
18 void init() {
19     root = tail = buffer;
20     tot = buffer + 1;
21     root->data = -1;
22     root->cnt = 0x7fffffff;
23     root->prev = root->nxt = NULL;
24 }
25 void adddata(eletype dta) {
26     tail->nxt = newnode(dta);
27     tail->nxt->prev = tail;
28     tail = tail->nxt;
29 }
30 void pull(node *x) {
31     node *ox = x->prev->prev, *oy = x->nxt, *oa = x->prev;
32     if (ox) ox->nxt = x;
33     x->prev = ox;
34     x->nxt = oy;
35     oy->prev = x;
36     oy->nxt = oy;
37     if (oy) oy->prev = oa;
38     if (tail == x) tail = oa;
39 }
40 bool access(eletype data) {
41     node *x;
42     for (x = root->nxt; x; x = x->nxt) {
43         if (x->data == data) {
44             x->cnt++;
45             while (x->prev->cnt < x->cnt) pull(x);
46             return true;
47         }
48     }
49     return false;
50 }
51 int main() {
52     init();
53     int n;
54     int x;
55     scanf("%d", &n);
56     for (int i = 0; i < n; ++i) {
57         scanf("%d", &x);
58         adddata(x);
59     }
60     puts("-----");
61     while (~scanf("%d", &x)) {
62         if (access(x)) {
63             puts("Successful");
```

```

64         for (node *x = root->nxt; x; x = x->nxt)
65             printf("%d ", x->data);
66         putchar('\n');
67     } else {
68         puts("Failed");
69     }
70 }
71 return 0;
72 }
73 /**
74  root ► ... ► git-repos ► data-structure-homework ► 09 ► g++ e25.cpp      ◀ master
75  root ► ... ► git-repos ► data-structure-homework ► 09 ► ./a.out        ◀ master
76  5 1 2 3 4 5
77  -----
78  2
79  Successful
80  2 1 3 4 5
81  5
82  Successful
83  2 5 1 3 4
84  5
85  Successful
86  5 2 1 3 4
87  6
88  Failed
89  0
90  Failed
91  2
92  Successful
93  5 2 1 3 4
94  1
95  Successful
96  5 2 1 3 4
97  1
98  Successful
99  5 2 1 3 4
100 1
101 Successful
102 1 5 2 3 4
103 2
104 Successful
105 1 2 5 3 4
106 2
107 Successful
108 2 1 5 3 4
109 ^Z
110 [1] + 123 suspended ./a.out
111 root ► ... ► git-repos ► data-structure-homework ► 09 ► 1 ►
112 */

```