

main.c

```
1 #include <stdio.h>
2 #define MAX 10
3 void addFront(int *, int, int *, int *);
4 void addRear(int *, int, int *, int *);
5 int delFront(int *, int *, int *);
6 int delRear(int *, int *, int *);
7 void display(int *);
8 int count(int *);
9 int main() {
10     int arr[MAX];
11     int front, rear, i, n;
12     front = rear = -1;
13     for (i = 0; i < MAX; i++)
14         arr[i] = 0;
15     addRear(arr, 5, &front, &rear);
16     addFront(arr, 12, &front, &rear);
17     addRear(arr, 11, &front, &rear);
18     addFront(arr, 5, &front, &rear);
19     addRear(arr, 6, &front, &rear);
20     addFront(arr, 8, &front, &rear);
21     printf("\nElements in a deque: ");
22     display(arr);
23     i = delFront(arr, &front, &rear);
24     printf("\nremoved item: %d", i);
25     printf("\nElements in a deque after deletion: ");
26     display(arr);
27     addRear(arr, 16, &front, &rear);
28     addRear(arr, 7, &front, &rear);
29     printf("\nElements in a deque after addition: ");
30     display(arr);
31     i = delRear(arr, &front, &rear);
32     printf("\nremoved item: %d", i);
33     printf("\nElements in a deque after deletion: ");
34     display(arr);
35     n = count(arr);
36     printf("\nTotal number of elements in deque: %d", n);
37 }
38 void addFront(int *arr, int item, int *pfront, int *prear) {
39     int i, k, c;
40     if (*pfront == 0 && *prear == MAX - 1) {
41         printf("\nDeque is full.\n");
42         return;
43     }
44     if (*pfront == -1) {
45         *pfront = *prear = 0;
```

```

45     *pfront = *prear = 0;
46     arr[*pfront] = item;
47     return;
48 }
49 if (*prear != MAX - 1) {
50     c = count(arr);
51     k = *prear + 1;
52     for (i = 1; i <= c; i++) {
53         arr[k] = arr[k - 1];
54         k--;
55     }
56     arr[k] = item;
57     *pfront = k;
58     (*prear)++;
59 } else {
60     (*pfront)--;
61     arr[*pfront] = item;
62 }
63 }
64 void addRear(int *arr, int item, int *pfront, int *prear) {
65     int i, k;
66     if (*pfront == 0 && *prear == MAX - 1) {
67         printf("\nDeque is full.\n");
68         return;
69     }
70     if (*pfront == -1) {
71         *prear = *pfront = 0;
72         arr[*prear] = item;
73         return;
74     }
75     if (*prear == MAX - 1) {
76         k = *pfront - 1;
77         for (i = *pfront - 1; i < *prear; i++) {
78             k = i;
79             if (k == MAX - 1)
80                 arr[k] = 0;
81             else
82                 arr[k] = arr[i + 1];
83         }
84         (*prear)--;
85         (*pfront)--;
86     }
87     (*prear)++;
88     arr[*prear] = item;
89 }
90 int delFront(int *arr, int *pfront, int *prear) {

```

```

86 }
87 (*prear)++;
88 arr[*prear] = item;
89 }
90 int delFront(int *arr, int *pfront, int *prear) {
91     int item;
92     if (*pfront == -1) {
93         printf("\nDeque is empty.\n");
94         return 0;
95     }
96     item = arr[*pfront];
97     arr[*pfront] = 0;
98     if (*pfront == *prear)
99         *pfront = *prear = -1;
100     else
101         (*pfront)++;
102     return item;
103 }
104 int delRear(int *arr, int *pfront, int *prear) {
105     int item;
106     if (*pfront == -1) {
107         printf("\nDeque is empty.\n");
108         return 0;
109     }
110     item = arr[*prear];
111     arr[*prear] = 0;
112     (*prear)--;
113     if (*prear == -1)
114         *pfront = -1;
115     return item;
116 }
117 void display(int *arr) {
118     int i;
119     printf("\n front: ");
120     for (i = 0; i < MAX; i++)
121         printf(" %d", arr[i]);
122     printf(" :rear");
123 }
124 int count(int *arr) {
125     int c = 0, i;
126     for (i = 0; i < MAX; i++) {
127         if (arr[i] != 0)
128             c++;
129     }
130     return c;
131 }

```