```c
//Double ended queue(Deque) implementation

#include <stdio.h>
#define size 5
int deque[size];
int f = -1, r = -1;
//  insert_front function will insert the value from the fr
void enqueueF(int x)
{
    if((f==0 && r==size-1) || (f==r+1))
    {
        printf("Overflow");
    }
    else if((f==-1) && (r==-1))
    {
        f=r=0;
        deque[f]=x;
    }
    else if(f==0)
    {
        f=size-1;
        deque[f]=x;
    }
    else
    {
        f=f-1;
        deque[f]=x;
    }
}

// insert_rear function will insert the value from the rear
void enqueueR(int x)
{
    if((f==0 && r==size-1) || (f==r+1))
    {
        printf("Overflow");
    }
    else if((f==-1) && (r==-1))
    {
```

```
40:         r=0;
41:         deque[r]=x;
42:     }
43:     else if(r==size-1)
44:     {
45:         r=0;
46:         deque[r]=x;
47:     }
48:     else
49:     {
50:         r++;
51:         deque[r]=x;
52:     }
53:
54: }
55:
56: // display function prints all the value of deque.
57: void display()
58: {
59:     int i=f;
60:     printf("\nElements in a deque are: ");
61:
62:     while(i!=r)
63:     {
64:         printf("%d ",deque[i]);
65:         i=(i+1)%size;
66:     }
67:      printf("%d",deque[r]);
68: }
69:
70: // getfront function retrieves the first value of the deque
71: void getfront()
72: {
73:     if((f==-1) && (r==-1))
74:     {
75:         printf("Deque is empty");
76:     }
77:     else
78:     {
```

```
 79:         printf("\nThe value of the element at front is: %d", deque
 80:     }
 81:
 82: }
 83:
 84: // getrear function retrieves the last value of the deque.
 85: void getrear()
 86: {
 87:     if((f==-1) && (r==-1))
 88:     {
 89:         printf("Deque is empty");
 90:     }
 91:     else
 92:     {
 93:         printf("\nThe value of the element at rear is %d", deque[r
 94:     }
 95:
 96: }
 97:
 98: // delete_front() function deletes the element from the fro
 99: void dequeueF()
100: {
101:     if((f==-1) && (r==-1))
102:     {
103:         printf("Deque is empty");
104:     }
105:     else if(f==r)
106:     {
107:         printf("\nThe deleted element is %d", deque[f]);
108:         f=-1;
109:         r=-1;
110:
111:     }
112:      else if(f==(size-1))
113:      {
114:          printf("\nThe deleted element is %d", deque[f]);
115:          f=0;
116:      }
117:      else
```

```
118:      {
119:           printf("\nThe deleted element is %d", deque[f]);
120:           f=f+1;
121:      }
122: }
123:
124: // delete_rear() function deletes the element from the rear
125: void dequeueR()
126: {
127:     if((f==-1) && (r==-1))
128:     {
129:         printf("Deque is empty");
130:     }
131:     else if(f==r)
132:     {
133:         printf("\nThe deleted element is %d", deque[r]);
134:         f=-1;
135:         r=-1;
136:
137:     }
138:      else if(r==0)
139:      {
140:          printf("\nThe deleted element is %d", deque[r]);
141:          r=size-1;
142:      }
143:      else
144:      {
145:           printf("\nThe deleted element is %d", deque[r]);
146:           r=r-1;
147:      }
148: }
149:
150: int main()
151: {
152:     enqueueF(20);
153:     enqueueF(10);
154:     enqueueF(30);
155:     enqueueR(50);
156:     enqueueR(80);
```

```
157:     display();  // Calling the display function to retrieve the va
158:     getfront();  // Retrieve the value at front-end
159:     getrear();  // Retrieve the value at rear-end
160:     dequeueF();
161:     dequeueR();
162:     display(); // calling display function to retrieve values afte
163:     return 0;
164: }
```