

```

1  #include <stdio.h>
2  #include <stdlib.h>
3
4  struct node
5  {
6      int info;
7      struct node *ptr;
8  }*front,*rear,*temp,*front1;
9
10
11 void enq(int data);
12 void deq();
13
14 void display();
15 void create();
16
17

```

```

19
20 int main()
21 {
22     int no, ch, e;
23
24     printf("\n 1 - Enqueue");
25     printf("\n 2 - Dequeue");
26     printf("\n 3 - Display");
27     printf("\n 4 - Exit");
28     create();
29     while (1)
30     {
31         printf("\n Enter choice : ");
32         scanf("%d", &ch);
33         switch (ch)
34         {
35             case 1:
36                 printf("Enter data : ");
37                 scanf("%d", &no);
38                 enq(no);
39                 break;
40             case 2:
41                 deq();
42                 break;
43             case 3:
44                 display();
45                 break;
46             case 4:
47                 exit(0);
48             default:
49                 printf("Wrong choice, Please enter correct choice ");
50                 break;
51         }
52     }
53     return 0;
54 }
55

```

```

56
57 void create()
58 {
59     front = rear = NULL;
60 }
61
62
63 void enq(int data)
64 {
65     if (rear == NULL)
66     {
67         rear = (struct node *)malloc(1*sizeof(struct node));
68         rear->ptr = NULL;
69         rear->info = data;
70         front = rear;
71     }
72     else
73     {
74         temp=(struct node *)malloc(1*sizeof(struct node));
75         rear->ptr = temp;
76         temp->info = data;
77         temp->ptr = NULL;
78
79         rear = temp;
80     }
81 }
82
83

```

```

ButtonList TextFieldDemo.java x threads.java x
85 void display()
86 {
87     front1 = front;
88
89     if ((front1 == NULL) && (rear == NULL))
90     {
91         printf("Queue is empty");
92         return;
93     }
94     while (front1 != rear)
95     {
96         printf("%d ", front1->info);
97         front1 = front1->ptr;
98     }
99     if (front1 == rear)
100         printf("%d", front1->info);
101 }
102
103

```

```

103
104 void deq()
105 {
106     front1 = front;
107
108     if (front1 == NULL)
109     {
110         printf("\n queue is empty");
111         return;
112     }
113     else
114         if (front1->ptr != NULL)
115         {
116             front1 = front1->ptr;
117             printf("\n Dequed value : %d", front->info);
118             free(front);
119             front = front1;
120         }
121     else
122     {
123         printf("\n Dequed value : %d", front->info);
124         free(front);
125         front = NULL;
126         rear = NULL;
127     }
128 }
129
130
131
132
133

```

```

1 - Enque
2 - Deque
3 - Display
4 - Exit

```

Enter choice : 1

Enter data : 22

Enter choice : 1

Enter data : 55

Enter choice : 1

Enter data : 33

Enter choice : 2

Dequed value : 22

Enter choice : 3

55 33

Enter choice : 4

...Program finished with exit code 0

Press ENTER to exit console.

```

1  #include<stdio.h>
2  #include<stdlib.h>
3  struct node
4  {
5      int info;
6      struct node *ptr;
7  }*top,*top1,*temp;
8
9  void push(int data);
10 void pop();
11 void display();
12 void create();
13
14
15
16 int main()
17 {
18     int no, ch, e;
19
20     printf("\n 1 - Push");
21     printf("\n 2 - Pop");
22     printf("\n 3 - Dipslay");
23     printf("\n 4 - Exit");
24
25     create();
26
27     while (1)
28     {
29         printf("\n Enter choice : ");
30         scanf("%d", &ch);
31
32         switch (ch)
33         {
34             case 1:
35                 printf("Enter data : ");
36                 scanf("%d", &no);
37                 push(no);
38                 break;
39             case 2:
40                 pop();
41                 break;
42             case 3:

```

```

42         break;
43     case 3:
44         display();
45         break;
46     case 4:
47         exit(0);
48     default :
49         printf(" Wrong choice, Please enter correct choice ");
50     }
51 }
52
53
54 void create()
55 {
56     top = NULL;
57 }
58
59
60 void push(int data)
61 {
62     if (top == NULL)
63     {
64         top =(struct node *)malloc(1*sizeof(struct node));
65         top->ptr = NULL;
66         top->info = data;
67     }
68     else
69     {
70         temp =(struct node *)malloc(1*sizeof(struct node));
71         temp->ptr = top;
72         temp->info = data;
73         top = temp;
74     }
75 }
76
77

```

```

75 }
76 }
77
78
79 void display()
80 {
81     top1 = top;
82     if (top1 == NULL)
83     {
84         printf("Stack is empty");
85         return;
86     }
87
88     while (top1 != NULL)
89     {
90         printf("%d ", top1->info);
91         top1 = top1->ptr;
92     }
93 }
94
95 void pop()
96 {
97     top1 = top;
98     if (top1 == NULL)
99     {
100         printf("\n Error : Trying to pop from empty stack");
101         return;
102     }
103     else
104     {
105         top1 = top1->ptr;
106         printf("\n Popped value : %d", top->info);
107         free(top);
108         top = top1;
109     }
110 }
111
112
113
114
115

```

```

1 - Push
2 - Pop
3 - Display
4 - Exit
Enter choice : 1
Enter data : 22

Enter choice : 1
Enter data : 55

Enter choice : 1
Enter data : 33

Enter choice : 2

Popped value : 33
Enter choice : 3
5 22
Enter choice : 4

..Program finished with exit code 0
Press ENTER to exit console.

```