

6b WAP to Implement Single Link List to simulate Stack & Queue Operations.

The screenshot shows two windows of the Dev-C++ IDE. Both windows display the same C code for implementing a singly linked list to simulate stack and queue operations. The code includes functions for creating nodes, pushing and popping from a stack, enqueueing and dequeuing from a queue, and displaying the list. The second window shows the main() function where the user can choose between stack and queue operations and enter data.

```
6b.c  singly.list
1 #include <cslibs.h>
2 #include <stdlib.h>
3
4 struct node {
5     int data;
6     struct node* next;
7 }
8
9 struct node* createnode(int data) {
10    struct node* newnode = (struct Node*)malloc(sizeof(struct Node));
11    newnode->data = data;
12    newnode->next = NULL;
13    return newnode;
14 }
15
16
17 struct node* push(struct Node* top, int data) {
18    struct node* newnode = createnode(data);
19    newnode->next = top;
20    return newnode;
21 }
22
23 struct node* pop(struct Node* top) {
24    if (top == NULL) {
25        printf("Stack Underflow\n");
26        return NULL;
27    }
28
29    struct node* temp = top;
30    printf("Popped: %d\n", temp->data);
31    top = top->next;
32    free(temp);
33    return top;
34 }
35
36 void display(stack(struct Node* top) {
37    printf("Stack: ");
38    while (top != NULL) {
39        printf("%d -> ", top->data);
40        top = top->next;
41    }
42    printf("\nMAX\n");
43 }
44
45
46 struct Node* enqueue(struct Node* front, struct Node** rear, int data) {
47    struct node* newnode = createnode(data);
48
49    if (front == NULL) {
50        front = newnode;
51        rear = &newnode;
52    } else {
53        newnode->next = front;
54        front = newnode;
55    }
56 }
57
58
59 struct Node* dequeue(struct Node* front, struct Node** rear) {
60    if (front == NULL) {
61        printf("Queue Underflow\n");
62        return NULL;
63    }
64
65    struct node* temp = front;
66    printf("Dequeued: %d\n", temp->data);
67    front = front->next;
68
69    if (front == NULL) {
70        free(temp);
71        front = NULL;
72    }
73
74    free(temp);
75    return front;
76 }
77
78 void displayqueue(struct Node* front) {
79    printf("Queue: ");
80    while (front != NULL) {
81        printf("%d -> ", front->data);
82        front = front->next;
83    }
84    printf("\nMAX\n");
85 }
86
87
88 int main() {
89    struct Node *stackTop = NULL;
90    struct Node *queueFront = NULL;
91    struct Node *queueRear = NULL;
92
93    int choice, data;
94
95    while (1) {
96        printf("1. Push (Stack)\n");
97        printf("2. Pop (Stack)\n");
98        printf("3. Display (Stack)\n");
99        printf("4. Enqueue (Queue)\n");
100       printf("5. Dequeue (Queue)\n");
101       printf("6. Display (Queue)\n");
102       printf("7. Exit\n");
103       printf("Enter your choice: ");
104       scanf("%d", &choice);
105
106       switch (choice) {
107           case 1:
108               printf("Enter data: ");
109               scanf("%d", &data);
110               stackTop = push(stackTop, data);
111               break;
112           default:
113               break;
114       }
115   }
116 }
```

File Edit Search View Project Execute Tools AStyle Window Help

TDM-GCC 4.9.2 64-bit Release

Project Classes Debug 0b.c singlyList.c

```
92
93     int choice, data;
94
95     while (1) {
96         printf("1. Push (Stack)\n");
97         printf("2. Pop (Stack)\n");
98         printf("3. Display Stack\n");
99         printf("4. Enqueue (Queue)\n");
100        printf("5. Dequeue (Queue)\n");
101        printf("6. Display Queue\n");
102        printf("7. Exit\n");
103        printf("Enter your choice: ");
104        scanf("%d", &choice);
105
106        switch (choice) {
107            case 1:
108                printf("Enter data: ");
109                scanf("%d", &data);
110                insert(data, stackTop, data);
111                printf("\n");
112                break;
113
114            case 2:
115                stackTop = pop(stackTop); printf("\n");
116                break;
117
118            case 3:
119                displayStack(stackTop); printf("\n");
120                break;
121
122            case 4:
123                printf("Enter data: ");
124                scanf("%d", &data);
125                enqueue(data, queueFront, &queueRear, data); printf("\n");
126                break;
127
128            case 5:
129                queueFront = dequeue(queueFront, &queueRear); printf("\n");
130                break;
131
132            case 6:
133                displayQueue(queueFront); printf("\n");
134                break;
135
136            case 7:
137                exit(0);
138            default:
139                printf("Invalid option!\n");
140        }
141    }
142
143
144 }
```

Compiler Resources Compile Log Debug Find Results Close

Compilation results...

Abort Compilation

- Errors: 0
- Warnings: 0
- Filename: C:\Users\BMSECSE\Documents\ds lab\0b.exe
- Output Size: 131.101625 KiB
- Compilation Time: 0.16s

Line: 91 Col: 35 Sek: 0 Lines: 144 Length: 3183 Insert Done parsing in 0.013 seconds

11:53:29 25-11-2025 ENG US

```
C:\Users\BMSCCSE\Documents> + - x
1. Push (Stack)
2. Pop (Stack)
3. Display Stack
4. Enqueue (Queue)
5. Dequeue (Queue)
6. Display Queue
7. Exit
Enter your choice: 1
Enter data: 12
1. Push (Stack)
2. Pop (Stack)
3. Display Stack
4. Enqueue (Queue)
5. Dequeue (Queue)
6. Display Queue
7. Exit
Enter your choice: 1
Enter data: 23
1. Push (Stack)
2. Pop (Stack)
3. Display Stack
4. Enqueue (Queue)
5. Dequeue (Queue)
6. Display Queue
7. Exit
Enter your choice: 2
Popped: 23
1. Push (Stack)
2. Pop (Stack)
3. Display Stack
4. Enqueue (Queue)
5. Dequeue (Queue)
6. Display Queue
7. Exit
Enter your choice: 3
Stack: 12 --> NULL
1. Push (Stack)
2. Pop (Stack)
3. Display Stack
4. Enqueue (Queue)
5. Dequeue (Queue)
6. Display Queue
7. Exit
Enter your choice: 4
Enter data: 45
1. Push (Stack)

NIFTY -0.03%
ENG US 11:53:40 25-11-2025

C:\Users\BMSCCSE\Documents> + - x
Enter data: 45
1. Push (Stack)
2. Pop (Stack)
3. Display Stack
4. Enqueue (Queue)
5. Dequeue (Queue)
6. Display Queue
7. Exit
Enter your choice: 4
Enter data: 56
1. Push (Stack)
2. Pop (Stack)
3. Display Stack
4. Enqueue (Queue)
5. Dequeue (Queue)
6. Display Queue
7. Exit
Enter your choice: 5
Dequeued: 45
1. Push (Stack)
2. Pop (Stack)
3. Display Stack
4. Enqueue (Queue)
5. Dequeue (Queue)
6. Display Queue
7. Exit
Enter your choice: 6
Queue: 56 --> NULL
1. Push (Stack)
2. Pop (Stack)
3. Display Stack
4. Enqueue (Queue)
5. Dequeue (Queue)
6. Display Queue
7. Exit
Enter your choice: 7
Process exited after 17.39 seconds with return value 0
Press any key to continue . . .

NIFTY -0.03%
ENG US 11:53:45 25-11-2025
```