

			a) WAP to simulate the working of a queue of integers using an array. Provide the following operations: Insert, Delete, Display The program should print appropriate messages for queue empty and queue overflow conditions
3	2	5	b) WAP to simulate the working of a circular queue of integers using an array. Provide the following operations: Insert, Delete & Display The program should print appropriate messages for queue empty and queue overflow conditions

The screenshot shows a C code editor interface with a toolbar at the top and several tabs labeled "Start here", "3.c", "4.c", "5.c", and "6.c". The main code area contains the following C code:

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 /* Definition of node */
5 struct node {
6     int data;
7     struct node *next;
8 };
9
10 struct node *head = NULL;
11
12 /* Create Linked List */
13 void create() {
14     int n, i, val;
15     struct node *newnode, *temp;
16
17     printf("Enter number of nodes: ");
18     scanf("%d", &n);
19
20     for (i = 0; i < n; i++) {
21         newnode = (struct node*)malloc(sizeof(struct node));
22         printf("Enter data: ");
23         scanf("%d", &val);
24
25         newnode->data = val;
26         newnode->next = NULL;
27
28         if (head == NULL) {
29             head = newnode;
30             temp = head;
31         } else {
32             temp->next = newnode;
33             temp = newnode;
34         }
35     }
36 }
37
```

The screenshot shows a C code editor interface with a toolbar at the top and a code editor window below. The code editor window has tabs for "Start here", "3.c", "4.c", "5.c", and "6.c". The code itself is a C program with three main functions: `insert_begin()`, `insert_end()`, and `insert_pos()`. The code uses a linked list structure where each node contains an integer value and a pointer to the next node. It includes memory allocation using `malloc` and input handling via `scanf`.

```
37  /* Insert at beginning */
38  void insert_begin() {
39      struct node *newnode;
40      newnode = (struct node*)malloc(sizeof(struct node));
41
42      printf("Enter data to insert at beginning: ");
43      scanf("%d", &newnode->data);
44
45      newnode->next = head;
46      head = newnode;
47  }
48
49  /* Insert at end */
50  void insert_end() {
51      struct node *newnode, *temp;
52      newnode = (struct node*)malloc(sizeof(struct node));
53
54      printf("Enter data to insert at end: ");
55      scanf("%d", &newnode->data);
56
57      newnode->next = NULL;
58
59      if (head == NULL) {
60          head = newnode;
61      } else {
62          temp = head;
63          while (temp->next != NULL)
64              temp = temp->next;
65          temp->next = newnode;
66      }
67  }
68
69  /* Insert at any position */
70  void insert_pos() {
71      int pos, i = 1;
72      struct node *newnode, *temp;
```

The screenshot shows a code editor window with a tab bar at the top labeled "Start here X 3.c X 4.c X 5.c X 6.c X". The main area displays the following C code:

```
70  /* Insert at any position */
71  void insert_pos() {
72      int pos, i = 1;
73      struct node *newnode, *temp;
74
75      printf("Enter position: ");
76      scanf("%d", &pos);
77
78      newnode = (struct node*)malloc(sizeof(struct node));
79      printf("Enter data: ");
80      scanf("%d", &newnode->data);
81
82      if (pos == 1) {
83          newnode->next = head;
84          head = newnode;
85          return;
86      }
87
88      temp = head;
89      while (i < pos - 1 && temp != NULL) {
90          temp = temp->next;
91          i++;
92      }
93
94      if (temp == NULL) {
95          printf("Invalid position!\n");
96      } else {
97          newnode->next = temp->next;
98          temp->next = newnode;
99      }
100 }
101
102 /* Display Linked List */
103 void display() {
104     struct node *temp = head;
105
106     if (head == NULL) {
```

A vertical green line on the left margin indicates the current line of execution or focus. The code uses standard C syntax with structures, pointers, and basic control flow statements.

C:\Users\HP\Documents\cj.ex + ▾

- 3. Insert at Any Position
- 4. Insert at End
- 5. Display
- 6. Exit

Enter your choice: 1

Enter number of nodes: 2

Enter data: 1

Enter data: 1

--- SINGLY LINKED LIST MENU ---

- 1. Create Linked List
- 2. Insert at Beginning
- 3. Insert at Any Position
- 4. Insert at End
- 5. Display
- 6. Exit

Enter your choice: 2

Enter data to insert at beginning: 1

--- SINGLY LINKED LIST MENU ---

- 1. Create Linked List
- 2. Insert at Beginning
- 3. Insert at Any Position
- 4. Insert at End
- 5. Display
- 6. Exit

Enter your choice: 5

Linked list contents:

1 -> 1 -> 1 -> NULL

56.c - Code::Blocks 25.03

File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help

Management x Start here x c.j.c x 56.c x

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 /* Node structure */
5 struct node {
6     int data;
7     struct node *next;
8 };
9
10 struct node *head = NULL;
11
12 /* Create Linked List */
13 void create() {
14     int n, i, val;
15     struct node *newnode, *temp;
16
17     printf("Enter number of nodes: ");
18     scanf("%d", &n);
19
20     for (i = 0; i < n; i++) {
21         newnode = (struct node*)malloc(sizeof(struct node));
22         printf("Enter data: ");
23         scanf("%d", &val);
24
25         newnode->data = val;
26         newnode->next = NULL;
27
28         if (head == NULL) {
29             head = newnode;
30             temp = head;
31         } else {
32             temp->next = newnode;
33             temp = newnode;
34         }
35     }
36 }
37
```

Logs & others x

C:\Users\HP\Documents\56.c C/C++ Windows (CR+LF) WINDOWS-1252 Line 168, Col 1, Pos 3703 Insert Read/Write default ENG IN 21:39 28-12-2025

22°C Mostly clear Search

```
Start here x c.j.c x 56.c x
37  /* Delete first element */
38  void delete_first() {
39      struct node *temp;
40
41      if (head == NULL) {
42          printf("List is empty. Cannot delete.\n");
43          return;
44      }
45
46      temp = head;
47      head = head->next;
48      printf("Deleted element: %d\n", temp->data);
49      free(temp);
50  }
51
52
53  /* Delete last element */
54  void delete_last() {
55      struct node *temp, *prev;
56
57      if (head == NULL) {
58          printf("List is empty. Cannot delete.\n");
59          return;
60      }
61
62      if (head->next == NULL) {
63          printf("Deleted element: %d\n", head->data);
64          free(head);
65          head = NULL;
66          return;
67      }
68
69      temp = head;
70      while (temp->next != NULL) {
71          prev = temp;
72          temp = temp->next;
73      }

```

Logs & others

C:\Users\Windows\Documents\GitHub\c-journal	Windows Terminal	WINDOWS 10 PRO	Line 150 Col 1 Dec 27 2023	Insert	Read Only	default
---	------------------	----------------	----------------------------	--------	-----------	---------

The screenshot shows a code editor window with a tab bar at the top containing "Start here", "cjc.c", and "56.c". The main area displays the following C code:

```
73     }
74
75     prev->next = NULL;
76     printf("Deleted element: %d\n", temp->data);
77     free(temp);
78 }
79
80 /* Delete specified element */
81 void delete_specified() {
82     int key;
83     struct node *temp, *prev;
84
85     if (head == NULL) {
86         printf("List is empty. Cannot delete.\n");
87         return;
88     }
89
90     printf("Enter element to delete: ");
91     scanf("%d", &key);
92
93     /* If first node is the key */
94     if (head->data == key) {
95         temp = head;
96         head = head->next;
97         printf("Deleted element: %d\n", temp->data);
98         free(temp);
99         return;
100    }
101
102    temp = head;
103    while (temp != NULL && temp->data != key) {
104        prev = temp;
105        temp = temp->next;
106    }
107
108    if (temp == NULL) {
109        printf("Element not found.\n");
```

56.c - Code::Blocks 25.03

File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help

<global>

Management x Start here x cj.c x 56.c x

```
109     printf("Element not found.\n");
110 } else {
111     prev->next = temp->next;
112     printf("Deleted element: %d\n", temp->data);
113     free(temp);
114 }
115
116 /* Display Linked List */
117 void display() {
118     struct node *temp = head;
119
120     if (head == NULL) {
121         printf("Linked list is empty.\n");
122         return;
123     }
124
125     printf("Linked list contents:\n");
126     while (temp != NULL) {
127         printf("%d -> ", temp->data);
128         temp = temp->next;
129     }
130     printf("NULL\n");
131 }
132
133 /* Main Function */
134 int main() {
135     int choice;
136
137     do {
138         printf("\n--- SINGLY LINKED LIST MENU ---");
139         printf("\n1. Create Linked List");
140         printf("\n2. Delete First Element");
141         printf("\n3. Delete Specified Element");
142         printf("\n4. Delete Last Element");
143         printf("\n5. Display");
144         printf("\n6. Exit");
145     }
```

Logs & others

C:\Users\HP\Documents\56.c C/C++ Windows (CR+LF) WINDOWS-1252 Line 168, Col 1, Pos 3703 Insert Read/Write default

56.c - Code::Blocks 25.03

File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help

<global>

Management Resources Resources

Start here x cjc x 56.c x

```
132 }  
133  
134 /* Main Function */  
135 int main() {  
136     int choice;  
137  
138     do {  
139         printf("\n--- SINGLY LINKED LIST MENU ---");  
140         printf("\n1. Create Linked List");  
141         printf("\n2. Delete First Element");  
142         printf("\n3. Delete Specified Element");  
143         printf("\n4. Delete Last Element");  
144         printf("\n5. Display");  
145         printf("\n6. Exit");  
146         printf("\nEnter your choice: ");  
147         scanf("%d", &choice);  
148  
149         switch (choice) {  
150             case 1: create();  
151                 break;  
152             case 2: delete_first();  
153                 break;  
154             case 3: delete_specified();  
155                 break;  
156             case 4: delete_last();  
157                 break;  
158             case 5: display();  
159                 break;  
160             case 6: printf("Exiting program.\n");  
161                 break;  
162             default: printf("Invalid choice!\n");  
163         }  
164     } while (choice != 6);  
165  
166     return 0;  
167 }  
168 }
```

Logs & others

C:\Users\HP\Documents\56.c C/C++ Windows (CR+LF) WINDOWS-1252 Line 168, Col 1, Pos 3703 Insert Read/Write default

C:\Users\HP\Documents\56.e

--- SINGLY LINKED LIST MENU ---

1. Create Linked List
2. Delete First Element
3. Delete Specified Element
4. Delete Last Element
5. Display
6. Exit

Enter your choice: 1

Enter number of nodes: 2

Enter data: 2

Enter data: 4

--- SINGLY LINKED LIST MENU ---

1. Create Linked List
2. Delete First Element
3. Delete Specified Element
4. Delete Last Element
5. Display
6. Exit

Enter your choice: 2

Deleted element: 2

--- SINGLY LINKED LIST MENU ---

1. Create Linked List
2. Delete First Element
3. Delete Specified Element
4. Delete Last Element
5. Display
6. Exit

Enter your choice: 5

Linked list contents:

4 -> NULL

--- SINGLY LINKED LIST MENU ---

1. Create Linked List
2. Delete First Element
3. Delete Specified Element
4. Delete Last Element
5. Display
6. Exit

Enter your choice: