

## LAB(2-12-2025)

### INPUT

```
File Edit Selection View Go Run Terminal Help
Welcome X C PRG.c X C prgm.c
C:\Users\BMSGECSE\Desktop> 1BF24C3243 > LAB(2-12-2025) > C PRG.c > @main()
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 struct Node {
5     int data;
6     struct Node *prev, *next;
7 };
8
9 struct Node *head = NULL, *tail = NULL;
10
11 // Create list with n nodes
12 void createlist(int n) {
13     int i, data;
14     struct Node *newNode;
15     for (i = 1; i <= n; i++) {
16         printf("Enter data for node %d: ", i);
17         scanf("%d", &data);
18
19         newNode = (struct Node*)malloc(sizeof(struct Node));
20         newNode->data = data;
21         newNode->prev = newNode->next = NULL;
22
23         if (head == NULL) {
24             head = tail = newNode;
25         } else {
26             tail->next = newNode;
27             newNode->prev = tail;
28             tail = newNode;
29         }
30     }
31 }
32
33 // Insert at front
34 void insertAtFront(int data) {
35     struct Node *newNode = (struct Node*)malloc(sizeof(struct Node));
36     newNode->data = data;
37     newNode->prev = NULL;
38     newNode->next = head;
39
40     if (head == NULL) {
41         head = tail = newNode;
42     } else {
43         head->prev = newNode;
44         head = newNode;
45     }
46 }
```

```
File Edit Selection View Go Run Terminal Help
Welcome X C PRG.c X C prgm.c
C:\Users\BMSGECSE\Desktop> 1BF24C3243 > LAB(2-12-2025) > C PRG.c > @main()
33 void insertAtFront(int data) {
34     struct Node *newNode = (struct Node*)malloc(sizeof(struct Node));
35     newNode->data = data;
36     newNode->prev = NULL;
37     newNode->next = head;
38     head = newNode;
39 }
40
41 // Insert at end
42 void insertAtEnd(int data) {
43     struct Node *newNode = (struct Node*)malloc(sizeof(struct Node));
44     newNode->data = data;
45     newNode->prev = NULL;
46     newNode->next = tail;
47
48     if (tail == NULL) {
49         head = tail = newNode;
50     } else {
51         tail->next = newNode;
52         tail = newNode;
53     }
54 }
55
56 // Insert at left
57 void insertAtLeftOfPosition(int data, int pos) {
58     int i;
59     struct Node *newNode, *temp = head;
60     if (pos == 1) {
61         insertAtFront(data);
62         return;
63     }
64     for (i = 1; i < pos - 1 && temp != NULL; i++) {
65         temp = temp->next;
66     }
67     if (temp == NULL) {
68         printf("Invalid position\n");
69         return;
70     }
71     newNode = (struct Node*)malloc(sizeof(struct Node));
72     newNode->data = data;
73     newNode->prev = temp->prev;
74     newNode->next = temp;
75     temp->prev->next = newNode;
76     temp->prev = newNode;
77 }
```

```
File Edit Selection View Go Run Terminal Help
C:\Users> BMSCICSE > Desktop > 1BF24CS243 > LABQ-12-2025 > C PRG.c > main()
62 void insertLeftOfPosition(int data, int pos) {
63     newNode = (struct Node*)malloc(sizeof(struct Node));
64     newNode->data = data;
65     newNode->next = temp->next;
66     newNode->prev = temp;
67     if (temp->next != NULL)
68         temp->next->prev = newNode;
69     temp->next = newNode;
70     if (newNode->next == NULL)
71         tail = newNode;
72 }
73 //Insert at right
74 void insertRightOfPosition(int data, int pos) {
75     int i;
76     struct Node *newNode, *temp = head;
77     for (i = 1; i < pos && temp != NULL; i++) {
78         temp = temp->next;
79     }
80     if (temp == NULL) {
81         printf("Invalid position\n");
82         return;
83     }
84     newNNode = (struct Node*)malloc(sizeof(struct Node));
85     newNNode->data = data;
86     newNNode->next = temp->next;
87     newNNode->prev = temp;
88     if (temp->next != NULL)
89         temp->next->prev = newNNode;
90     else
91         tail = newNNode;
92     temp->next = newNNode;
93 }
```

```
File Edit Selection View Go Run Terminal Help
C:\Users> BMSCICSE > Desktop > 1BF24CS243 > LABQ-12-2025 > C PRG.c > main()
91 void insertRightOfPosition(int data, int pos) {
110     newNNode->next = temp->next;
111     newNNode->prev = temp;
112     if (temp->next != NULL)
113         temp->next->prev = newNNode;
114     else
115         tail = newNNode;
116     temp->next = newNNode;
117 }
118 //delete by value
119 void deleteByValue(int value) {
120     struct Node *temp = head;
121     if (head == NULL) {
122         printf("List is empty\n");
123         return;
124     }
125     while (temp != NULL && temp->data != value)
126         temp = temp->next;
127     if (temp == NULL) {
128         printf("Value not found\n");
129         return;
130     }
131     if (temp == head) {
132         head = head->next;
133         if (head == NULL)
134             head->prev = NULL;
135         else
136             tail = head;
137     }
138     else if (temp == tail) {
139         tail = tail->prev;
140         tail->next = NULL;
141     }
142     else {
143         temp->prev->next = temp->next;
144     }
145 }
```

```
File Edit Selection View Go Run Terminal Help
C PRG.c X C prg.c
C:\Users> BMSGCCSE > Desktop > 1BF24CS243 > LABQ2-12-2025 > C PRG.c > main()
124 void deleteByValue(int value) {
125     if (temp == NULL) {
126         tail = tail->prev;
127         tail->next = NULL;
128     }
129     else {
130         temp->prev->next = temp->next;
131         temp->next->prev = temp->prev;
132     }
133     free(temp);
134     printf("Node with value %d deleted.\n", value);
135 }
136
137 //delete by pos
138 void deleteAtPosition(int pos) {
139     struct Node *temp = head;
140     int i;
141
142     if (head == NULL) {
143         printf("List is empty!\n");
144         return;
145     }
146
147     if (pos == 1) {
148         head = head->next;
149         if (head != NULL)
150             head->prev = NULL;
151         else
152             tail = NULL;
153         free(temp);
154         return;
155     }
156     for (i = 1; i < pos && temp != NULL; i++) {
157         temp = temp->next;
158     }
159     if (temp == NULL) {
160         printf("Invalid position!\n");
161         return;
162     }
163     if (temp == tail) {
164         tail = tail->prev;
165     }
166     temp->prev->next = temp->next;
167     temp->next->prev = temp->prev;
168     free(temp);
169     printf("Node at position %d deleted.\n", pos);
170 }
171
172 // Display forward
173 void displayForward() {
174     struct Node *temp = head;
175     printf("List (Forward): ");
176     while (temp != NULL) {
177         printf("%d <-> ", temp->data);
178         temp = temp->next;
179     }
180     printf("NULL\n");
181 }
182
183 // Display backward
184 void displayBackward() {
185     struct Node *temp = tail;
186     printf("List (Backward): ");
187     while (temp != NULL) {
188         printf("%d <-> ", temp->data);
189         temp = temp->prev;
190     }
191     printf("NULL\n");
192 }
193
194 int main() {
195     int choice, n, data, pos, value;
196 }
```

```
File Edit Selection View Go Run Terminal Help
C PRG.c X C prg.c
C:\Users> BMSGCCSE > Desktop > 1BF24CS243 > LABQ2-12-2025 > C PRG.c > main()
141 void deleteAtPosition(int pos) {
142     for (i = 1; i < pos && temp != NULL; i++) {
143     }
144     if (temp == NULL) {
145         printf("Invalid position!\n");
146         return;
147     }
148     if (temp == tail) {
149         tail = tail->prev;
150         tail->next = NULL;
151     }
152     else {
153         temp->prev->next = temp->next;
154         temp->next->prev = temp->prev;
155     }
156     free(temp);
157     printf("Node at position %d deleted.\n", pos);
158 }
159
160 // Display forward
161 void displayForward() {
162     struct Node *temp = head;
163     printf("List (Forward): ");
164     while (temp != NULL) {
165         printf("%d <-> ", temp->data);
166         temp = temp->next;
167     }
168     printf("NULL\n");
169 }
170
171 // Display backward
172 void displayBackward() {
173     struct Node *temp = tail;
174     printf("List (Backward): ");
175     while (temp != NULL) {
176         printf("%d <-> ", temp->data);
177         temp = temp->prev;
178     }
179     printf("NULL\n");
180 }
181
182 int main() {
183     int choice, n, data, pos, value;
184 }
```

```
File Edit Selection View Go Run Terminal Help
C:\Users> BMSCEICSE > Desktop > 1BF24CS243 > LAB02-12-2025 > C PRGc > (main)
222 int main() {
223     int choice, n, data, pos, value;
224
225     while (1) {
226         printf("\n--- Doubly Linked List Menu ---\n");
227         printf("1. Create List\n");
228         printf("2. Insert to Left of a Node (Position)\n");
229         printf("3. Insert to right of a Node (Position)\n");
230         printf("4. Delete Node by Value\n");
231         printf("5a. Delete Node by position\n");
232         printf("6a. Display forward\n");
233         printf("7a. Display backward\n");
234         printf("8. Exit\n");
235         printf("Enter choice: ");
236         scanf("%d", &choice);
237
238         switch (choice) {
239             case 1:
240                 printf("Enter number of nodes: ");
241                 scanf("%d", &n);
242                 createlist(n);
243                 break;
244             case 2:
245                 printf("Enter data to insert: ");
246                 scanf("%d", &data);
247                 printf("Insert to the left of position: ");
248                 scanf("%d", &pos);
249                 insertleftOfPosition(data, pos);
250                 break;
251             case 3:
252                 printf("Enter data to insert: ");
253                 scanf("%d", &data);
254                 printf("Insert to the left of position: ");
255                 scanf("%d", &pos);
256                 insertrightOfPosition(data, pos);
257                 break;
258             case 4:
259                 printf("Enter value to delete: ");
260                 scanf("%d", &value);
261                 deletebyValue(value);
262                 break;
263             case 5:
264                 printf("Enter position to delete: ");
265                 scanf("%d", &pos);
266                 deletebyPosition(pos);
267                 break;
268             case 6:
269                 displayforward();
270                 break;
271             case 7:
272                 displaybackward();
273                 break;
274             case 8:
275                 printf("Exiting...\n");
276                 exit(0);
277             default:
278                 printf("Invalid choice! Try again.\n");
279         }
280     }
281     return 0;
282 }
```

```
File Edit Selection View Go Run Terminal Help
C:\Users> BMSCEICSE > Desktop > 1BF24CS243 > LAB02-12-2025 > C PRGc > (main)
222 int main() {
223     while (1) {
224         switch (choice) {
225             case 1:
226                 scanf("%d", &data);
227                 printf("Insert to the left of position: ");
228                 scanf("%d", &pos);
229                 insertrightOfPosition(data, pos);
230                 break;
231             case 4:
232                 printf("Enter value to delete: ");
233                 scanf("%d", &value);
234                 deletebyValue(value);
235                 break;
236             case 5:
237                 printf("Enter position to delete: ");
238                 scanf("%d", &pos);
239                 deletebyPosition(pos);
240                 break;
241             case 6:
242                 displayforward();
243                 break;
244             case 7:
245                 displaybackward();
246                 break;
247             case 8:
248                 printf("Exiting...\n");
249                 exit(0);
250             default:
251                 printf("Invalid choice! Try again.\n");
252         }
253     }
254     return 0;
255 }
```

## OUTPUT

```
PS C:\Users\UMGCECE> cd "c:\Users\UMGCECE\Desktop\1BF24C5243\LAB(2-12-2025)\output"
PS C:\Users\UMGCECE\Desktop\1BF24C5243\LAB(2-12-2025)\output> .\PMS.exe

--- Doubly Linked List Menu ---
1. Create List
2. Insert to Left of a Node (Position)
3. Insert to right of a Node (Position)
4. Delete Node by Value
5. Delete Node by position
6. Display forward
7. Display backward
8. Exit
Enter choice: 1
Enter number of nodes: 2
Enter data for node 1: 10
Enter data for node 2: 20

--- Doubly Linked List Menu ---
1. Create List
2. Insert to Left of a Node (Position)
3. Insert to right of a Node (Position)
4. Delete Node by Value
5. Delete Node by position
6. Display forward
7. Display backward
8. Exit
Enter choice: 2
Enter data to insert: 30
Insert to the left of position: 2

--- Doubly Linked List Menu ---
1. Create List
2. Insert to Left of a Node (Position)
3. Insert to right of a Node (Position)
4. Delete Node by Value
5. Delete Node by position
6. Display forward
7. Display backward
8. Exit
Enter choice: 3
Enter data to insert: 40
Insert to the left of position: 2

--- Doubly Linked List Menu ---
1. Create List
2. Insert to Left of a Node (Position)
3. Insert to right of a Node (Position)
4. Delete Node by Value
5. Delete Node by position
6. Display forward
7. Display backward
8. Exit
Enter choice: 4
Enter value to delete: 20
Node with value 20 deleted.
```

```
Enter value to delete: 20
Node with value 20 deleted.

--- Doubly Linked List Menu ---
1. Create List
2. Insert to Left of a Node (Position)
3. Insert to right of a Node (Position)
4. Delete Node by Value
5. Delete Node by position
6. Display forward
7. Display backward
8. Exit
Enter choice: 5
Enter position to delete: 3
Node at position 3 deleted.

--- Doubly Linked List Menu ---
1. Create List
2. Insert to Left of a Node (Position)
3. Insert to right of a Node (Position)
4. Delete Node by Value
5. Delete Node by position
6. Display forward
7. Display backward
8. Exit
Enter choice: 6
List (forward): 10 <-> 30 <-> NULL

--- Doubly Linked List Menu ---
1. Create List
2. Insert to Left of a Node (Position)
3. Insert to right of a Node (Position)
4. Delete Node by Value
5. Delete Node by position
6. Display forward
7. Display backward
8. Exit
Enter choice: 7
List (Backward): 30 <-> 10 <-> NULL

--- Doubly Linked List Menu ---
1. Create List
2. Insert to Left of a Node (Position)
3. Insert to right of a Node (Position)
4. Delete Node by Value
5. Delete Node by position
6. Display forward
7. Display backward
8. Exit
Enter choice: 8
Exiting...
PS C:\Users\UMGCECE\Desktop\1BF24C5243\LAB(2-12-2025)\output>
```