

Activities | Text Editor | Aug 22 3:09 PM | eexp5.c | Save | - | + | x

```
1#include <stdio.h>
2#include <stdlib.h>
3#include <malloc.h>
4
5// Defining Structure
6typedef struct node
7{
8    int data;
9    struct node *next;
10} node;
11
12node *createlist();
13node *Insert_beg(node *head, int x);
14node *Insert_end(node *head, int x);
15node *Insert_mid(node *head, int x);
16node *Delete_beg(node *head);
17node *Delete_end(node *head);
18node *Delete_mid(node *head);
19void PrintList(node *head);
20
21// Main function
22void main()
23{
24    int choice, insert_option, delete_option, x;
25    node *head = NULL;
26    printf("Welcome to the implementation of the singly linked list ! \n");
27    do
28    {
29        printf("Please select an operation to perform from the below list \n");
30        printf("1. Create a list \n 2. Insert a node \n 3. Delete a node \n 4. Print the existing list \n 5. Exit \n");
31        printf("Enter your choice: ");
32        scanf("%d", &choice);
33        printf("\n");
34        switch (choice)
35        {
36            case 1:
37                head = createlist();
38                break;
39            case 2:
40                do
41                {
42                    printf("Select a position where you want to insert new node \n");
43                    printf("1. Beginning of the list \n 2. At the end of the list \n 3. Insert in between \n 4. Exit the insert operation \n");
44                    printf("Enter your choice: ");
45                    scanf("%d", &insert_option);
46                    break;
47                } while (insert_option != 4);
48                printf("\n");
49                printf("Enter the data to be inserted: ");
50                scanf("%d", &x);
51                head = Insert_beg(head, x);
52                break;
53            case 3:
54                do
55                {
56                    printf("Select a position where you want to delete the element \n");
57                    printf("1. Beginning of the list \n 2. At the end of the list \n 3. Somewhere in between \n 4. Exit the delete operation \n");
58                    printf("Enter your choice: ");
59                    scanf("%d", &delete_option);
60                    break;
61                } while (delete_option != 4);
62                printf("\n");
63                printf("Enter the data to be inserted: ");
64                scanf("%d", &x);
65                head = Insert_mid(head, x);
66                break;
67            case 4:
68                printf("Insert operation Exit");
69                break;
70            default:
71                printf("Please enter a valid choice: 1, 2, 3, 4");
72                break;
73        }
74    } while (choice != 5);
75    PrintList(head);
76}
```

C | Tab Width: 8 | Ln 5, Col 22 | INS

Activities | Text Editor | Aug 22 3:09 PM | eexp5.c | Save | - | + | x

```
44    scanf("%d", &insert_option);
45    switch (insert_option)
46    {
47        case 1:
48            printf("Enter the data to be inserted: ");
49            scanf("%d", &x);
50            head = Insert_beg(head, x);
51            break;
52        case 2:
53            printf("Enter the data to be inserted: ");
54            scanf("%d", &x);
55            head = Insert_end(head, x);
56            break;
57        case 3:
58            printf("Enter the data to be inserted: ");
59            scanf("%d", &x);
60            head = Insert_mid(head, x);
61            break;
62        case 4:
63            printf("Insert operation Exit");
64            break;
65        default:
66            printf("Please enter a valid choice: 1, 2, 3, 4");
67            break;
68    }
69    while (insert_option != 4);
70    printf("\n");
71    break;
72    case 3:
73    {
74        printf("Select a position from where you want to delete the element \n");
75        printf("1. Beginning of the list \n 2. At the end of the list \n 3. Somewhere in between \n 4. Exit the delete operation \n");
76        printf("Enter your choice: ");
77        scanf("%d", &delete_option);
78        switch (delete_option)
79        {
80            case 1:
81                head = Delete_beg(head);
82                break;
83            case 2:
84                head = Delete_end(head);
85                break;
86            case 3:
87                head = Delete_mid(head);
88                break;
89            case 4:
90                printf("Delete operation Exit");
91                break;
92            default:
93                printf("Please enter a valid choice: 1, 2, 3, 4");
94                break;
95        }
96    } while (delete_option != 4);
97    PrintList(head);
98}
```

C | Tab Width: 8 | Ln 5, Col 22 | INS

Activities Text Editor Aug 22 3:09 PM eexp5.c Save

```
87 head = Delete_mid(head);
88 break;
89 case 4:
90     printf("Delete Operation Exit");
91     break;
92     default:
93         printf("Please enter a valid choide: 1, 2, 3, 4");
94     }
95     while (delete_option != 4);
96     printf("\n\n");
97     break;
98 case 1:
99     PrintList(head);
100    break;
101 case 2:
102     printf("Exit: Program Finished !!");
103     break;
104     default:
105         printf("Please enter a valid choide: 1, 2, 3, 4, 5");
106     }
107     while (choice != 5);
108 }
109 // Function to create list
110 node *createList()
111 {
112     node *head, *p;
113     int i, n;
114     head = NULL;
115     printf("Enter the number of nodes: ");
116     scanf("%d", &n);
117     printf("Enter the data: ");
118     for (i = 0; i <= n - 1; i++)
119     {
120         if (head == NULL)
121         {
122             p = head = (node *)malloc(sizeof(node));
123         }
124         else
125         {
126             p->next = (node *)malloc(sizeof(node));
127             p = p->next;
128         }
129         p->next = NULL;
130         scanf("%d", &p->data);
131     }
132     return head;
133 }
```

Activities Text Editor Aug 22 3:09 PM eexp5.c Save

```
130 p->next = NULL;
131 scanf("%d", &(p->data));
132 }
133 printf("\n\n");
134 return head;
135 }
136 // Function to insert element
137 node *Insert_beg(node *head, int x)
138 {
139     node *p;
140     p = (node *)malloc(sizeof(node));
141     p->data = x;
142     p->next = head;
143     head = p;
144     return head;
145 }
146 node *Insert_end(node *head, int x)
147 {
148     node *p, *q;
149     p = (node *)malloc(sizeof(node));
150     p->data = x;
151     p->next = NULL;
152     if (head == NULL)
153         return p;
154     for (q = head; q->next != NULL; q = q->next)
155     {
156     }
157     q->next = p;
158     return head;
159 }
160 node *Insert_mid(node *head, int x)
161 {
162     node *p, *q;
163     int y;
164     p = (node *)malloc(sizeof(node));
165     p->data = x;
166     p->next = NULL;
167     printf("After which element you want to insert the new element ?");
168     scanf("%d", &y);
169     for (q = head; q != NULL && q->data != y; q = q->next)
170     {
171     }
172     if (q != NULL)
173     {
174         p->next = q->next;
175         q->next = p;
176     }
177     return head;
178 }
```

Activities Text Editor Aug 22 3:09 PM eexp5.c Save

```
173 p->next = q->next;
174 q->next = p;
175 }
176 else
177     printf("ERROR !! Data Not Found");
178     return (head);
179 }
180 // Function to delete element
181 node *Delete_beg(node *head)
182 {
183     node *p, *q;
184     if (head == NULL)
185     {
186         printf("Empty Linked List");
187         return (head);
188     }
189     p = head;
190     head = head->next;
191     free(p);
192     return (head);
193 }
194 // Function to delete element
195 node *Delete_end(node *head)
196 {
197     node *p, *q;
198     if (head == NULL)
199     {
200         printf("Empty Linked List");
201         return (head);
202     }
203     p = head;
204     if (head->next == NULL)
205     {
206         head = NULL;
207         free(p);
208         return (head);
209     }
210     for (q = head; q->next->next != NULL; q = q->next)
211         p = q->next;
212     q->next = NULL;
213     free(p);
214     return (head);
215 }
216 node *Delete_mid(node *head)
217 {
```

Activities Text Editor Aug 22 3:09 PM eexp5.c Save

```
218     node *p, *q;
219     int x, i;
220     if (head == NULL)
221     {
222         printf("Empty Linked List");
223         return (head);
224     }
225     printf("Enter the data to be deleted: ");
226     scanf("%d", &x);
227     if (head->data == x)
228     {
229         p = head;
230         head = head->next;
231         free(p);
232         return (head);
233     }
234     for (q = head; q->next->data != x && q->next != NULL; q = q->next)
235         if (q->next == NULL)
236         {
237             printf("ERROR !! Data Not Found");
238             return (head);
239         }
240     p = q->next;
241     q->next = q->next->next;
242     free(p);
243     return (head);
244 }
245 // Function to print the existing list
246 void PrintList(node *head)
247 {
248     node *p;
249     printf("\n");
250     for (p = head; p != NULL; p = p->next)
251     {
252         printf("%d\t", p->data);
253     }
254     printf("\n");
255     printf("\n");
256 }
257
```

