SINGLY LINKED LISTS

#include<stdio.h>

 #include<stdlib.h>

 struct node

 {

     int data;

     struct node \*next;

 };

 struct node \*head;



 void beginsert ();

 void lastinsert ();

 void randominsert();

 void begin\_delete();

 void last\_delete();

 void random\_delete();

 void display();

 void search();

 void main ()

 {

     int choice =0;

     while(choice != 9)

     {

         printf("\n\n\*\*\*\*\*\*\*\*\*Main Menu\*\*\*\*\*\*\*\*\*\n");

         printf("\nChoose one option from the following list ...\n");

         printf("\n===============================================\n");

         printf("\n1.Insert in begining\n2.Insert at last\n3.Insert at any random location\n4.Delete from Beginning\n

         5.Delete from last\n6.Delete node after specified location\n7.Search for an element\n8.Show\n9.Exit\n");

         printf("\nEnter your choice?\n");

         scanf("\n%d",&choice);

         switch(choice)

         {

             case 1:

             beginsert();

             break;

             case 2:

             lastinsert();

             break;

             case 3:

             randominsert();

             break;

             case 4:

             begin\_delete();

             break;

             case 5:

             last\_delete();

             break;

             case 6:

             random\_delete();

             break;

             case 7:

             search();

             break;

             case 8:

             display();

             break;

             case 9:

             exit(0);

             break;

             default:

             printf("Please enter valid choice..");

         }

     }

 }

 void beginsert()

 {

     struct node \*ptr;

     int item;

     ptr = (struct node \*) malloc(sizeof(struct node \*));

     if(ptr == NULL)

     {

         printf("\nOVERFLOW");

     }

     else

     {

         printf("\nEnter value\n");

         scanf("%d",&item);

         ptr->data = item;

         ptr->next = head;

         head = ptr;

         printf("\nNode inserted");

     }



 }

 void lastinsert()

 {

     struct node \*ptr,\*temp;

     int item;

     ptr = (struct node\*)malloc(sizeof(struct node));

     if(ptr == NULL)

     {

         printf("\nOVERFLOW");

     }

     else

     {

         printf("\nEnter value?\n");

         scanf("%d",&item);

         ptr->data = item;

         if(head == NULL)

         {

             ptr -> next = NULL;

             head = ptr;

             printf("\nNode inserted");

         }

         else

         {

             temp = head;

             while (temp -> next != NULL)

             {

                 temp = temp -> next;

             }

             temp->next = ptr;

             ptr->next = NULL;

             printf("\nNode inserted");



         }

     }

 }

 void randominsert()

 {

     int i,loc,item;

     struct node \*ptr, \*temp;

     ptr = (struct node \*) malloc (sizeof(struct node));

     if(ptr == NULL)

     {

         printf("\nOVERFLOW");

     }

     else

     {

         printf("\nEnter element value");

         scanf("%d",&item);

         ptr->data = item;

         printf("\nEnter the location after which you want to insert ");

         scanf("\n%d",&loc);

         temp=head;

         for(i=0;i<loc;i++)

         {

             temp = temp->next;

             if(temp == NULL)

             {

                 printf("\ncan't insert\n");

                 return;

             }



         }

         ptr ->next = temp ->next;

         temp ->next = ptr;

         printf("\nNode inserted");

     }

 }

 void begin\_delete()

 {

     struct node \*ptr;

     if(head == NULL)

     {

         printf("\nList is empty\n");

     }

     else

     {

         ptr = head;

         head = ptr->next;

         free(ptr);

         printf("\nNode deleted from the begining ...\n");

     }

 }

 void last\_delete()

 {

     struct node \*ptr,\*ptr1;

     if(head == NULL)

     {

         printf("\nlist is empty");

     }

     else if(head -> next == NULL)

     {

         head = NULL;

         free(head);

         printf("\nOnly node of the list deleted ...\n");

     }



     else

     {

         ptr = head;

         while(ptr->next != NULL)

         {

             ptr1 = ptr;

             ptr = ptr ->next;

         }

         ptr1->next = NULL;

         free(ptr);

         printf("\nDeleted Node from the last ...\n");

     }

 }

 void random\_delete()

 {

     struct node \*ptr,\*ptr1;

     int loc,i;

     printf("\n Enter the location of the node after which you want to perform deletion \n");

     scanf("%d",&loc);

     ptr=head;

     for(i=0;i<loc;i++)

     {

         ptr1 = ptr;

         ptr = ptr->next;



         if(ptr == NULL)

         {

             printf("\nCan't delete");

             return;

         }

     }

     ptr1 ->next = ptr ->next;

     free(ptr);

     printf("\nDeleted node %d ",loc+1);

 }

 void search()

 {

     struct node \*ptr;

     int item,i=0,flag;

     ptr = head;

     if(ptr == NULL)

     {

         printf("\nEmpty List\n");

     }

     else

     {

         printf("\nEnter item which you want to search?\n");

         scanf("%d",&item);

         while (ptr!=NULL)

         {

             if(ptr->data == item)

             {

                 printf("item found at location %d ",i+1);

                 flag=0;

             }

             else

             {

                 flag=1;

             }

             i++;

             ptr = ptr -> next;

         }

         if(flag==1)

         {

             printf("Item not found\n");

         }

     }



 }



 void display()

 {

     struct node \*ptr;

     ptr = head;

     if(ptr == NULL)

     {

         printf("Nothing to print");

     }

     else

     {

         printf("\nprinting values . . . . .\n");

         while (ptr!=NULL)

         {

             printf("\n%d",ptr->data);

             ptr = ptr -> next;

         }

     }

 }

OUTPUT

\*\*\*\*\*\*\*\*\*Main Menu\*\*\*\*\*\*\*\*\*

Choose one option from the following list ...

===============================================

1.Insert in begining

2.Insert at last

3.Insert at any random location

4.Delete from Beginning

5.Delete from last

6.Delete node after specified location

7.Search for an element

8.Show

9.Exit

Enter your choice?

1

Enter value

1

Node inserted

\*\*\*\*\*\*\*\*\*Main Menu\*\*\*\*\*\*\*\*\*

Choose one option from the following list ...

===============================================

1.Insert in begining

2.Insert at last

3.Insert at any random location

4.Delete from Beginning

5.Delete from last

6.Delete node after specified location

7.Search for an element

8.Show

9.Exit

Enter your choice?

2

Enter value?

2

Node inserted

\*\*\*\*\*\*\*\*\*Main Menu\*\*\*\*\*\*\*\*\*

Choose one option from the following list ...

===============================================

1.Insert in begining

2.Insert at last

3.Insert at any random location

4.Delete from Beginning

5.Delete from last

6.Delete node after specified location

7.Search for an element

8.Show

9.Exit

Enter your choice?

3

Enter element value1

Enter the location after which you want to insert 1

Node inserted

\*\*\*\*\*\*\*\*\*Main Menu\*\*\*\*\*\*\*\*\*

Choose one option from the following list ...

===============================================

1.Insert in begining

2.Insert at last

3.Insert at any random location

4.Delete from Beginning

5.Delete from last

6.Delete node after specified location

7.Search for an element

8.Show

9.Exit

Enter your choice?

8

printing values . . . . .

1

2

1

\*\*\*\*\*\*\*\*\*Main Menu\*\*\*\*\*\*\*\*\*

Choose one option from the following list ...

===============================================

1.Insert in begining

2.Insert at last

3.Insert at any random location

4.Delete from Beginning

5.Delete from last

6.Delete node after specified location

7.Search for an element

8.Show

9.Exit

Enter your choice?

2

Enter value?

123

Node inserted

\*\*\*\*\*\*\*\*\*Main Menu\*\*\*\*\*\*\*\*\*

Choose one option from the following list ...

===============================================

1.Insert in begining

2.Insert at last

3.Insert at any random location

4.Delete from Beginning

5.Delete from last

6.Delete node after specified location

7.Search for an element

8.Show

9.Exit

Enter your choice?

1

Enter value

1234

Node inserted

\*\*\*\*\*\*\*\*\*Main Menu\*\*\*\*\*\*\*\*\*

Choose one option from the following list ...

===============================================

1.Insert in begining

2.Insert at last

3.Insert at any random location

4.Delete from Beginning

5.Delete from last

6.Delete node after specified location

7.Search for an element

8.Show

9.Exit

Enter your choice?

4

Node deleted from the begining ...

\*\*\*\*\*\*\*\*\*Main Menu\*\*\*\*\*\*\*\*\*

Choose one option from the following list ...

===============================================

1.Insert in begining

2.Insert at last

3.Insert at any random location

4.Delete from Beginning

5.Delete from last

6.Delete node after specified location

7.Search for an element

8.Show

9.Exit

Enter your choice?

5

Deleted Node from the last ...

\*\*\*\*\*\*\*\*\*Main Menu\*\*\*\*\*\*\*\*\*

Choose one option from the following list ...

===============================================

1.Insert in begining

2.Insert at last

3.Insert at any random location

4.Delete from Beginning

5.Delete from last

6.Delete node after specified location

7.Search for an element

8.Show

9.Exit

Enter your choice?

6

Enter the location of the node after which you want to perform deletion

1

Deleted node 2

\*\*\*\*\*\*\*\*\*Main Menu\*\*\*\*\*\*\*\*\*

Choose one option from the following list ...

===============================================

1.Insert in begining

2.Insert at last

3.Insert at any random location

4.Delete from Beginning

5.Delete from last

6.Delete node after specified location

7.Search for an element

8.Show

9.Exit

Enter your choice?

8

printing values . . . . .

1

1

\*\*\*\*\*\*\*\*\*Main Menu\*\*\*\*\*\*\*\*\*

Choose one option from the following list ...

===============================================

1.Insert in begining

2.Insert at last

3.Insert at any random location

4.Delete from Beginning

5.Delete from last

6.Delete node after specified location

7.Search for an element

8.Show

9.Exit

Enter your choice?

7

Enter item which you want to search?

1

item found at location 1

item found at location 2

\*\*\*\*\*\*\*\*\*Main Menu\*\*\*\*\*\*\*\*\*

Choose one option from the following list ...

===============================================

1.Insert in begining

2.Insert at last

3.Insert at any random location

4.Delete from Beginning

5.Delete from last

6.Delete node after specified location

7.Search for an element

8.Show

9.Exit

Enter your choice?

9

DOUBLY LINKED LISTS

1. #include<stdio.h>
2. #include<stdlib.h>
3. struct node
4. {
5. struct node \*prev;
6. struct node \*next;
7. int data;
8. };
9. struct node \*head;
10. void insertion\_beginning();
11. void insertion\_last();
12. void insertion\_specified();
13. void deletion\_beginning();
14. void deletion\_last();
15. void deletion\_specified();
16. void display();
17. void search();
18. void main ()
19. {
20. int choice =0;
21. while(choice != 9)
22. {
23. printf("\n\*\*\*\*\*\*\*\*\*Main Menu\*\*\*\*\*\*\*\*\*\n");
24. printf("\nChoose one option from the following list ...\n");
25. printf("\n===============================================\n");
26. printf("\n1.Insert in begining\n2.Insert at last\n3.Insert at any random location\n4.Delete from Beginning\n
27. 5.Delete from last\n6.Delete the node after the given data\n7.Search\n8.Show\n9.Exit\n");
28. printf("\nEnter your choice?\n");
29. scanf("\n%d",&choice);
30. switch(choice)
31. {
32. case 1:
33. insertion\_beginning();
34. break;
35. case 2:
36. insertion\_last();
37. break;
38. case 3:
39. insertion\_specified();
40. break;
41. case 4:
42. deletion\_beginning();
43. break;
44. case 5:
45. deletion\_last();
46. break;
47. case 6:
48. deletion\_specified();
49. break;
50. case 7:
51. search();
52. break;
53. case 8:
54. display();
55. break;
56. case 9:
57. exit(0);
58. break;
59. default:
60. printf("Please enter valid choice..");
61. }
62. }
63. }
64. void insertion\_beginning()
65. {
66. struct node \*ptr;
67. int item;
68. ptr = (struct node \*)malloc(sizeof(struct node));
69. if(ptr == NULL)
70. {
71. printf("\nOVERFLOW");
72. }
73. else
74. {
75. printf("\nEnter Item value");
76. scanf("%d",&item);
78. if(head==NULL)
79. {
80. ptr->next = NULL;
81. ptr->prev=NULL;
82. ptr->data=item;
83. head=ptr;
84. }
85. else
86. {
87. ptr->data=item;
88. ptr->prev=NULL;
89. ptr->next = head;
90. head->prev=ptr;
91. head=ptr;
92. }
93. printf("\nNode inserted\n");
94. }
96. }
97. void insertion\_last()
98. {
99. struct node \*ptr,\*temp;
100. int item;
101. ptr = (struct node \*) malloc(sizeof(struct node));
102. if(ptr == NULL)
103. {
104. printf("\nOVERFLOW");
105. }
106. else
107. {
108. printf("\nEnter value");
109. scanf("%d",&item);
110. ptr->data=item;
111. if(head == NULL)
112. {
113. ptr->next = NULL;
114. ptr->prev = NULL;
115. head = ptr;
116. }
117. else
118. {
119. temp = head;
120. while(temp->next!=NULL)
121. {
122. temp = temp->next;
123. }
124. temp->next = ptr;
125. ptr ->prev=temp;
126. ptr->next = NULL;
127. }
129. }
130. printf("\nnode inserted\n");
131. }
132. void insertion\_specified()
133. {
134. struct node \*ptr,\*temp;
135. int item,loc,i;
136. ptr = (struct node \*)malloc(sizeof(struct node));
137. if(ptr == NULL)
138. {
139. printf("\n OVERFLOW");
140. }
141. else
142. {
143. temp=head;
144. printf("Enter the location");
145. scanf("%d",&loc);
146. for(i=0;i<loc;i++)
147. {
148. temp = temp->next;
149. if(temp == NULL)
150. {
151. printf("\n There are less than %d elements", loc);
152. return;
153. }
154. }
155. printf("Enter value");
156. scanf("%d",&item);
157. ptr->data = item;
158. ptr->next = temp->next;
159. ptr -> prev = temp;
160. temp->next = ptr;
161. temp->next->prev=ptr;
162. printf("\nnode inserted\n");
163. }
164. }
165. void deletion\_beginning()
166. {
167. struct node \*ptr;
168. if(head == NULL)
169. {
170. printf("\n UNDERFLOW");
171. }
172. else if(head->next == NULL)
173. {
174. head = NULL;
175. free(head);
176. printf("\nnode deleted\n");
177. }
178. else
179. {
180. ptr = head;
181. head = head -> next;
182. head -> prev = NULL;
183. free(ptr);
184. printf("\nnode deleted\n");
185. }
187. }
188. void deletion\_last()
189. {
190. struct node \*ptr;
191. if(head == NULL)
192. {
193. printf("\n UNDERFLOW");
194. }
195. else if(head->next == NULL)
196. {
197. head = NULL;
198. free(head);
199. printf("\nnode deleted\n");
200. }
201. else
202. {
203. ptr = head;
204. if(ptr->next != NULL)
205. {
206. ptr = ptr -> next;
207. }
208. ptr -> prev -> next = NULL;
209. free(ptr);
210. printf("\nnode deleted\n");
211. }
212. }
213. void deletion\_specified()
214. {
215. struct node \*ptr, \*temp;
216. int val;
217. printf("\n Enter the data after which the node is to be deleted : ");
218. scanf("%d", &val);
219. ptr = head;
220. while(ptr -> data != val)
221. ptr = ptr -> next;
222. if(ptr -> next == NULL)
223. {
224. printf("\nCan't delete\n");
225. }
226. else if(ptr -> next -> next == NULL)
227. {
228. ptr ->next = NULL;
229. }
230. else
231. {
232. temp = ptr -> next;
233. ptr -> next = temp -> next;
234. temp -> next -> prev = ptr;
235. free(temp);
236. printf("\nnode deleted\n");
237. }
238. }
239. void display()
240. {
241. struct node \*ptr;
242. printf("\n printing values...\n");
243. ptr = head;
244. while(ptr != NULL)
245. {
246. printf("%d\n",ptr->data);
247. ptr=ptr->next;
248. }
249. }
250. void search()
251. {
252. struct node \*ptr;
253. int item,i=0,flag;
254. ptr = head;
255. if(ptr == NULL)
256. {
257. printf("\nEmpty List\n");
258. }
259. else
260. {
261. printf("\nEnter item which you want to search?\n");
262. scanf("%d",&item);
263. while (ptr!=NULL)
264. {
265. if(ptr->data == item)
266. {
267. printf("\nitem found at location %d ",i+1);
268. flag=0;
269. break;
270. }
271. else
272. {
273. flag=1;
274. }
275. i++;
276. ptr = ptr -> next;
277. }
278. if(flag==1)
279. {
280. printf("\nItem not found\n");
281. }
282. }
284. }

OUTPUT:

\*\*\*\*\*\*\*\*\*Main Menu\*\*\*\*\*\*\*\*\*

Choose one option from the following list ...

===============================================

1.Insert in begining

2.Insert at last

3.Insert at any random location

4.Delete from Beginning

5.Delete from last

6.Delete the node after the given data

7.Search

8.Show

9.Exit

Enter your choice?

8

printing values...

\*\*\*\*\*\*\*\*\*Main Menu\*\*\*\*\*\*\*\*\*

Choose one option from the following list ...

===============================================

1.Insert in begining

2.Insert at last

3.Insert at any random location

4.Delete from Beginning

5.Delete from last

6.Delete the node after the given data

7.Search

8.Show

9.Exit

Enter your choice?

1

Enter Item value12

Node inserted

\*\*\*\*\*\*\*\*\*Main Menu\*\*\*\*\*\*\*\*\*

Choose one option from the following list ...

===============================================

1.Insert in begining

2.Insert at last

3.Insert at any random location

4.Delete from Beginning

5.Delete from last

6.Delete the node after the given data

7.Search

8.Show

9.Exit

Enter your choice?

1

Enter Item value123

Node inserted

\*\*\*\*\*\*\*\*\*Main Menu\*\*\*\*\*\*\*\*\*

Choose one option from the following list ...

===============================================

1.Insert in begining

2.Insert at last

3.Insert at any random location

4.Delete from Beginning

5.Delete from last

6.Delete the node after the given data

7.Search

8.Show

9.Exit

Enter your choice?

1

Enter Item value1234

Node inserted

\*\*\*\*\*\*\*\*\*Main Menu\*\*\*\*\*\*\*\*\*

Choose one option from the following list ...

===============================================

1.Insert in begining

2.Insert at last

3.Insert at any random location

4.Delete from Beginning

5.Delete from last

6.Delete the node after the given data

7.Search

8.Show

9.Exit

Enter your choice?

8

printing values...

1234

123

12

\*\*\*\*\*\*\*\*\*Main Menu\*\*\*\*\*\*\*\*\*

Choose one option from the following list ...

===============================================

1.Insert in begining

2.Insert at last

3.Insert at any random location

4.Delete from Beginning

5.Delete from last

6.Delete the node after the given data

7.Search

8.Show

9.Exit

Enter your choice?

2

Enter value89

node inserted

\*\*\*\*\*\*\*\*\*Main Menu\*\*\*\*\*\*\*\*\*

Choose one option from the following list ...

===============================================

1.Insert in begining

2.Insert at last

3.Insert at any random location

4.Delete from Beginning

5.Delete from last

6.Delete the node after the given data

7.Search

8.Show

9.Exit

Enter your choice?

3

Enter the location1

Enter value12345

node inserted

\*\*\*\*\*\*\*\*\*Main Menu\*\*\*\*\*\*\*\*\*

Choose one option from the following list ...

===============================================

1.Insert in begining

2.Insert at last

3.Insert at any random location

4.Delete from Beginning

5.Delete from last

6.Delete the node after the given data

7.Search

8.Show

9.Exit

Enter your choice?

8

printing values...

1234

123

12345

12

89

\*\*\*\*\*\*\*\*\*Main Menu\*\*\*\*\*\*\*\*\*

Choose one option from the following list ...

===============================================

1.Insert in begining

2.Insert at last

3.Insert at any random location

4.Delete from Beginning

5.Delete from last

6.Delete the node after the given data

7.Search

8.Show

9.Exit

Enter your choice?

4

node deleted

\*\*\*\*\*\*\*\*\*Main Menu\*\*\*\*\*\*\*\*\*

Choose one option from the following list ...

===============================================

1.Insert in begining

2.Insert at last

3.Insert at any random location

4.Delete from Beginning

5.Delete from last

6.Delete the node after the given data

7.Search

8.Show

9.Exit

Enter your choice?

5

node deleted

\*\*\*\*\*\*\*\*\*Main Menu\*\*\*\*\*\*\*\*\*

Choose one option from the following list ...

===============================================

1.Insert in begining

2.Insert at last

3.Insert at any random location

4.Delete from Beginning

5.Delete from last

6.Delete the node after the given data

7.Search

8.Show

9.Exit

Enter your choice?

8

printing values...

123

12345

\*\*\*\*\*\*\*\*\*Main Menu\*\*\*\*\*\*\*\*\*

Choose one option from the following list ...

===============================================

1.Insert in begining

2.Insert at last

3.Insert at any random location

4.Delete from Beginning

5.Delete from last

6.Delete the node after the given data

7.Search

8.Show

9.Exit

Enter your choice?

6

Enter the data after which the node is to be deleted : 123

\*\*\*\*\*\*\*\*\*Main Menu\*\*\*\*\*\*\*\*\*

Choose one option from the following list ...

===============================================

1.Insert in begining

2.Insert at last

3.Insert at any random location

4.Delete from Beginning

5.Delete from last

6.Delete the node after the given data

7.Search

8.Show

9.Exit

Enter your choice?

8

printing values...

123

\*\*\*\*\*\*\*\*\*Main Menu\*\*\*\*\*\*\*\*\*

Choose one option from the following list ...

===============================================

1.Insert in begining

2.Insert at last

3.Insert at any random location

4.Delete from Beginning

5.Delete from last

6.Delete the node after the given data

7.Search

8.Show

9.Exit

Enter your choice?

7

Enter item which you want to search?

123

item found at location 1

\*\*\*\*\*\*\*\*\*Main Menu\*\*\*\*\*\*\*\*\*

Choose one option from the following list ...

===============================================

1.Insert in begining

2.Insert at last

3.Insert at any random location

4.Delete from Beginning

5.Delete from last

6.Delete the node after the given data

7.Search

8.Show

9.Exit

Enter your choice?

6

Enter the data after which the node is to be deleted : 123

Can't delete

\*\*\*\*\*\*\*\*\*Main Menu\*\*\*\*\*\*\*\*\*

Choose one option from the following list ...

===============================================

1.Insert in begining

2.Insert at last

3.Insert at any random location

4.Delete from Beginning

5.Delete from last

6.Delete the node after the given data

7.Search

8.Show

9.Exit

Enter your choice?

9

Exited..

**CIRCULAR LINKED LIST**

1. #include<stdio.h>
2. #include<stdlib.h>
3. struct node
4. {
5. int data;
6. struct node \*next;
7. };
8. struct node \*head;
10. void beginsert ();
11. void lastinsert ();
12. void randominsert();
13. void begin\_delete();
14. void last\_delete();
15. void random\_delete();
16. void display();
17. void search();
18. void main ()
19. {
20. int choice =0;
21. while(choice != 7)
22. {
23. printf("\n\*\*\*\*\*\*\*\*\*Main Menu\*\*\*\*\*\*\*\*\*\n");
24. printf("\nChoose one option from the following list ...\n");
25. printf("\n===============================================\n");
26. printf("\n1.Insert in begining\n2.Insert at last\n3.Delete from Beginning\n4.Delete from last\n5.Search for an element\n6.Show\n7.Exit\n");
27. printf("\nEnter your choice?\n");
28. scanf("\n%d",&choice);
29. switch(choice)
30. {
31. case 1:
32. beginsert();
33. break;
34. case 2:
35. lastinsert();
36. break;
37. case 3:
38. begin\_delete();
39. break;
40. case 4:
41. last\_delete();
42. break;
43. case 5:
44. search();
45. break;
46. case 6:
47. display();
48. break;
49. case 7:
50. exit(0);
51. break;
52. default:
53. printf("Please enter valid choice..");
54. }
55. }
56. }
57. void beginsert()
58. {
59. struct node \*ptr,\*temp;
60. int item;
61. ptr = (struct node \*)malloc(sizeof(struct node));
62. if(ptr == NULL)
63. {
64. printf("\nOVERFLOW");
65. }
66. else
67. {
68. printf("\nEnter the node data?");
69. scanf("%d",&item);
70. ptr -> data = item;
71. if(head == NULL)
72. {
73. head = ptr;
74. ptr -> next = head;
75. }
76. else
77. {
78. temp = head;
79. while(temp->next != head)
80. temp = temp->next;
81. ptr->next = head;
82. temp -> next = ptr;
83. head = ptr;
84. }
85. printf("\nnode inserted\n");
86. }
88. }
89. void lastinsert()
90. {
91. struct node \*ptr,\*temp;
92. int item;
93. ptr = (struct node \*)malloc(sizeof(struct node));
94. if(ptr == NULL)
95. {
96. printf("\nOVERFLOW\n");
97. }
98. else
99. {
100. printf("\nEnter Data?");
101. scanf("%d",&item);
102. ptr->data = item;
103. if(head == NULL)
104. {
105. head = ptr;
106. ptr -> next = head;
107. }
108. else
109. {
110. temp = head;
111. while(temp -> next != head)
112. {
113. temp = temp -> next;
114. }
115. temp -> next = ptr;
116. ptr -> next = head;
117. }
119. printf("\nnode inserted\n");
120. }
122. }
124. void begin\_delete()
125. {
126. struct node \*ptr;
127. if(head == NULL)
128. {
129. printf("\nUNDERFLOW");
130. }
131. else if(head->next == head)
132. {
133. head = NULL;
134. free(head);
135. printf("\nnode deleted\n");
136. }
138. else
139. {   ptr = head;
140. while(ptr -> next != head)
141. ptr = ptr -> next;
142. ptr->next = head->next;
143. free(head);
144. head = ptr->next;
145. printf("\nnode deleted\n");
147. }
148. }
149. void last\_delete()
150. {
151. struct node \*ptr, \*preptr;
152. if(head==NULL)
153. {
154. printf("\nUNDERFLOW");
155. }
156. else if (head ->next == head)
157. {
158. head = NULL;
159. free(head);
160. printf("\nnode deleted\n");
162. }
163. else
164. {
165. ptr = head;
166. while(ptr ->next != head)
167. {
168. preptr=ptr;
169. ptr = ptr->next;
170. }
171. preptr->next = ptr -> next;
172. free(ptr);
173. printf("\nnode deleted\n");
175. }
176. }
178. void search()
179. {
180. struct node \*ptr;
181. int item,i=0,flag=1;
182. ptr = head;
183. if(ptr == NULL)
184. {
185. printf("\nEmpty List\n");
186. }
187. else
188. {
189. printf("\nEnter item which you want to search?\n");
190. scanf("%d",&item);
191. if(head ->data == item)
192. {
193. printf("item found at location %d",i+1);
194. flag=0;
195. }
196. else
197. {
198. while (ptr->next != head)
199. {
200. if(ptr->data == item)
201. {
202. printf("item found at location %d ",i+1);
203. flag=0;
204. break;
205. }
206. else
207. {
208. flag=1;
209. }
210. i++;
211. ptr = ptr -> next;
212. }
213. }
214. if(flag != 0)
215. {
216. printf("Item not found\n");
217. }
218. }
220. }
222. void display()
223. {
224. struct node \*ptr;
225. ptr=head;
226. if(head == NULL)
227. {
228. printf("\nnothing to print");
229. }
230. else
231. {
232. printf("\n printing values ... \n");
234. while(ptr -> next != head)
235. {
237. printf("%d\n", ptr -> data);
238. ptr = ptr -> next;
239. }
240. printf("%d\n", ptr -> data);
241. }
243. }

**OUTPUT:**

\*\*\*\*\*\*\*\*\*Main Menu\*\*\*\*\*\*\*\*\*

Choose one option from the following list ...

===============================================

1.Insert in begining

2.Insert at last

3.Delete from Beginning

4.Delete from last

5.Search for an element

6.Show

7.Exit

Enter your choice?

1

Enter the node data?10

node inserted

\*\*\*\*\*\*\*\*\*Main Menu\*\*\*\*\*\*\*\*\*

Choose one option from the following list ...

===============================================

1.Insert in begining

2.Insert at last

3.Delete from Beginning

4.Delete from last

5.Search for an element

6.Show

7.Exit

Enter your choice?

2

Enter Data?20

node inserted

\*\*\*\*\*\*\*\*\*Main Menu\*\*\*\*\*\*\*\*\*

Choose one option from the following list ...

===============================================

1.Insert in begining

2.Insert at last

3.Delete from Beginning

4.Delete from last

5.Search for an element

6.Show

7.Exit

Enter your choice?

2

Enter Data?30

node inserted

\*\*\*\*\*\*\*\*\*Main Menu\*\*\*\*\*\*\*\*\*

Choose one option from the following list ...

===============================================

1.Insert in begining

2.Insert at last

3.Delete from Beginning

4.Delete from last

5.Search for an element

6.Show

7.Exit

Enter your choice?

3

node deleted

\*\*\*\*\*\*\*\*\*Main Menu\*\*\*\*\*\*\*\*\*

Choose one option from the following list ...

===============================================

1.Insert in begining

2.Insert at last

3.Delete from Beginning

4.Delete from last

5.Search for an element

6.Show

7.Exit

Enter your choice?

4

node deleted

\*\*\*\*\*\*\*\*\*Main Menu\*\*\*\*\*\*\*\*\*

Choose one option from the following list ...

===============================================

1.Insert in begining

2.Insert at last

3.Delete from Beginning

4.Delete from last

5.Search for an element

6.Show

7.Exit

Enter your choice?

5

Enter item which you want to search?

20

item found at location 1

\*\*\*\*\*\*\*\*\*Main Menu\*\*\*\*\*\*\*\*\*

Choose one option from the following list ...

===============================================

1.Insert in begining

2.Insert at last

3.Delete from Beginning

4.Delete from last

5.Search for an element

6.Show

7.Exit

Enter your choice?

6

printing values ...

20

\*\*\*\*\*\*\*\*\*Main Menu\*\*\*\*\*\*\*\*\*

Choose one option from the following list ...

===============================================

1.Insert in begining

2.Insert at last

3.Delete from Beginning

4.Delete from last

5.Search for an element

6.Show

7.Exit

Enter your choice?

7

**CIRCULAR DOUBLY LINKED LIST**

1. #include<stdio.h>
2. #include<stdlib.h>
3. struct node
4. {
5. struct node \*prev;
6. struct node \*next;
7. int data;
8. };
9. struct node \*head;
10. void insertion\_beginning();
11. void insertion\_last();
12. void deletion\_beginning();
13. void deletion\_last();
14. void display();
15. void search();
16. void main ()
17. {
18. int choice =0;
19. while(choice != 9)
20. {
21. printf("\n\*\*\*\*\*\*\*\*\*Main Menu\*\*\*\*\*\*\*\*\*\n");
22. printf("\nChoose one option from the following list ...\n");
23. printf("\n===============================================\n");
24. printf("\n1.Insert in Beginning\n2.Insert at last\n3.Delete from Beginning\n4.Delete from last\n5.Search\n6.Show\n7.Exit\n");
25. printf("\nEnter your choice?\n");
26. scanf("\n%d",&choice);
27. switch(choice)
28. {
29. case 1:
30. insertion\_beginning();
31. break;
32. case 2:
33. insertion\_last();
34. break;
35. case 3:
36. deletion\_beginning();
37. break;
38. case 4:
39. deletion\_last();
40. break;
41. case 5:
42. search();
43. break;
44. case 6:
45. display();
46. break;
47. case 7:
48. exit(0);
49. break;
50. default:
51. printf("Please enter valid choice..");
52. }
53. }
54. }
55. void insertion\_beginning()
56. {
57. struct node \*ptr,\*temp;
58. int item;
59. ptr = (struct node \*)malloc(sizeof(struct node));
60. if(ptr == NULL)
61. {
62. printf("\nOVERFLOW");
63. }
64. else
65. {
66. printf("\nEnter Item value");
67. scanf("%d",&item);
68. ptr->data=item;
69. if(head==NULL)
70. {
71. head = ptr;
72. ptr -> next = head;
73. ptr -> prev = head;
74. }
75. else
76. {
77. temp = head;
78. while(temp -> next != head)
79. {
80. temp = temp -> next;
81. }
82. temp -> next = ptr;
83. ptr -> prev = temp;
84. head -> prev = ptr;
85. ptr -> next = head;
86. head = ptr;
87. }
88. printf("\nNode inserted\n");
89. }
91. }
92. void insertion\_last()
93. {
94. struct node \*ptr,\*temp;
95. int item;
96. ptr = (struct node \*) malloc(sizeof(struct node));
97. if(ptr == NULL)
98. {
99. printf("\nOVERFLOW");
100. }
101. else
102. {
103. printf("\nEnter value");
104. scanf("%d",&item);
105. ptr->data=item;
106. if(head == NULL)
107. {
108. head = ptr;
109. ptr -> next = head;
110. ptr -> prev = head;
111. }
112. else
113. {
114. temp = head;
115. while(temp->next !=head)
116. {
117. temp = temp->next;
118. }
119. temp->next = ptr;
120. ptr ->prev=temp;
121. head -> prev = ptr;
122. ptr -> next = head;
123. }
124. }
125. printf("\nnode inserted\n");
126. }
128. void deletion\_beginning()
129. {
130. struct node \*temp;
131. if(head == NULL)
132. {
133. printf("\n UNDERFLOW");
134. }
135. else if(head->next == head)
136. {
137. head = NULL;
138. free(head);
139. printf("\nnode deleted\n");
140. }
141. else
142. {
143. temp = head;
144. while(temp -> next != head)
145. {
146. temp = temp -> next;
147. }
148. temp -> next = head -> next;
149. head -> next -> prev = temp;
150. free(head);
151. head = temp -> next;
152. }
154. }
155. void deletion\_last()
156. {
157. struct node \*ptr;
158. if(head == NULL)
159. {
160. printf("\n UNDERFLOW");
161. }
162. else if(head->next == head)
163. {
164. head = NULL;
165. free(head);
166. printf("\nnode deleted\n");
167. }
168. else
169. {
170. ptr = head;
171. if(ptr->next != head)
172. {
173. ptr = ptr -> next;
174. }
175. ptr -> prev -> next = head;
176. head -> prev = ptr -> prev;
177. free(ptr);
178. printf("\nnode deleted\n");
179. }
180. }
182. void display()
183. {
184. struct node \*ptr;
185. ptr=head;
186. if(head == NULL)
187. {
188. printf("\nnothing to print");
189. }
190. else
191. {
192. printf("\n printing values ... \n");
194. while(ptr -> next != head)
195. {
197. printf("%d\n", ptr -> data);
198. ptr = ptr -> next;
199. }
200. printf("%d\n", ptr -> data);
201. }
203. }
205. void search()
206. {
207. struct node \*ptr;
208. int item,i=0,flag=1;
209. ptr = head;
210. if(ptr == NULL)
211. {
212. printf("\nEmpty List\n");
213. }
214. else
215. {
216. printf("\nEnter item which you want to search?\n");
217. scanf("%d",&item);
218. if(head ->data == item)
219. {
220. printf("item found at location %d",i+1);
221. flag=0;
222. }
223. else
224. {
225. while (ptr->next != head)
226. {
227. if(ptr->data == item)
228. {
229. printf("item found at location %d ",i+1);
230. flag=0;
231. break;
232. }
233. else
234. {
235. flag=1;
236. }
237. i++;
238. ptr = ptr -> next;
239. }
240. }
241. if(flag != 0)
242. {
243. printf("Item not found\n");
244. }
245. }
247. }

**OUTPUT:**

\*\*\*\*\*\*\*\*\*Main Menu\*\*\*\*\*\*\*\*\*

Choose one option from the following list ...

===============================================

1.Insert in Beginning

2.Insert at last

3.Delete from Beginning

4.Delete from last

5.Search

6.Show

7.Exit

Enter your choice?

1

Enter Item value123

Node inserted

\*\*\*\*\*\*\*\*\*Main Menu\*\*\*\*\*\*\*\*\*

Choose one option from the following list ...

===============================================

1.Insert in Beginning

2.Insert at last

3.Delete from Beginning

4.Delete from last

5.Search

6.Show

7.Exit

Enter your choice?

2

Enter value234

node inserted

\*\*\*\*\*\*\*\*\*Main Menu\*\*\*\*\*\*\*\*\*

Choose one option from the following list ...

===============================================

1.Insert in Beginning

2.Insert at last

3.Delete from Beginning

4.Delete from last

5.Search

6.Show

7.Exit

Enter your choice?

1

Enter Item value90

Node inserted

\*\*\*\*\*\*\*\*\*Main Menu\*\*\*\*\*\*\*\*\*

Choose one option from the following list ...

===============================================

1.Insert in Beginning

2.Insert at last

3.Delete from Beginning

4.Delete from last

5.Search

6.Show

7.Exit

Enter your choice?

2

Enter value80

node inserted

\*\*\*\*\*\*\*\*\*Main Menu\*\*\*\*\*\*\*\*\*

Choose one option from the following list ...

===============================================

1.Insert in Beginning

2.Insert at last

3.Delete from Beginning

4.Delete from last

5.Search

6.Show

7.Exit

Enter your choice?

3

\*\*\*\*\*\*\*\*\*Main Menu\*\*\*\*\*\*\*\*\*

Choose one option from the following list ...

===============================================

1.Insert in Beginning

2.Insert at last

3.Delete from Beginning

4.Delete from last

5.Search

6.Show

7.Exit

Enter your choice?

4

node deleted

\*\*\*\*\*\*\*\*\*Main Menu\*\*\*\*\*\*\*\*\*

Choose one option from the following list ...

===============================================

1.Insert in Beginning

2.Insert at last

3.Delete from Beginning

4.Delete from last

5.Search

6.Show

7.Exit

Enter your choice?

6

printing values ...

123

\*\*\*\*\*\*\*\*\*Main Menu\*\*\*\*\*\*\*\*\*

Choose one option from the following list ...

===============================================

1.Insert in Beginning

2.Insert at last

3.Delete from Beginning

4.Delete from last

5.Search

6.Show

7.Exit

Enter your choice?

5

Enter item which you want to search?

123

item found at location 1

\*\*\*\*\*\*\*\*\*Main Menu\*\*\*\*\*\*\*\*\*

Choose one option from the following list ...

============================================

1.Insert in Beginning

2.Insert at last

3.Delete from Beginning

4.Delete from last

5.Search

6.Show

7.Exit

Enter your choice?

7