PROGRAM TO IMPLEMENT LINKED LIST OPERATIONS

```
1   #include <stdio.h>
2   #include <stdlib.h>
3   struct node
4   {
5           int data;
6           struct node *link;
7   }*head;
8   void insertionatbegining();
9   void insertionatend();
10  void insertionatanyposition();
11  void deletionatbegining();
12  void deletionatend();
13  void deletionatanyposition();
14  void display();
15  void main()
16  {
17          int ch;
18          do
19          {
20                  printf("1)insertion at begining\n2)insertion at
21          end\n3)insertion at any position\n4)deletion at
22          begining\n5)deletion at end\n6)deletion at any
23          position\n7)Display\n8)Exit\nEnter Your Choice\n");
24          scanf("%d",&ch);
25          switch(ch)
26          {
27                  case 1:
```

```
28              {
29                      insertionatbegining(); break;
30              }
31              case 2:
32              {
33                      insertionatend(); break;
34              }
35              case 3:
36              {
37                      insertionatanyposition(); break;
38              }
39              case 4:
40              {
41                      deletionatbegining(); break;
42              }
43              case 5:
44              {
45                      deletionatend(); break;
46              }
47              case 6:
48              {
49                      deletionatanyposition(); break;
50              }
51              case 7:
52              {
53                      display(); break;
54              }
55              default:
```

```
56                      printf("Wrong Choice");
57              }
58      }
59      while(ch!=8);
60  }
61      void insertionatbegining()
62      {
63              int item;
64              struct node *temp;
65              temp=(struct node*)malloc(sizeof(struct node));
66              printf("Enter the data to be inserted");
67              scanf("%d",&item);
68              if(temp==NULL)
69              {
70                      printf("No memory space available");
71              }
72              else
73              {
74                      temp->data=item;
75                      temp->link=head;
76                      head=temp;
77              }
78      }
79      void insertionatend()
80      {
81              int item;
82              struct node *temp,*ptr;
83              if(head==NULL)
```

```
84              {
85                      insertionatbegining();
86              }
87              else
88              {
89                      printf("Enter the data to be inserted");
90                      scanf("%d",&item);
91                      temp=(struct node*)malloc(sizeof(struct
92                       node));
93                      ptr=head;
94                      while(ptr->link!=NULL)
95                      {
96                              ptr=ptr->link;
97                      }
98                      ptr->link=temp;
99                      temp->data=item;
100                     temp->link=NULL;
101             }
102     }
103     void insertionatanyposition()
104     {
105             int item,key;
106             struct node *temp,*ptr;
107             if(head==NULL)
108             {
109
110                     insertionatbegining();
111             }
```

```
112             else
113             {
114                     printf("Enter the data to be inserted");
115                     scanf("%d",&item);
116                     temp=(struct node*)malloc(sizeof(struct
117     node));
118                     printf("Enter the number after which value to
119     be added");
120                     scanf("%d",&key);
121                     ptr=head;
122                     while(ptr->data!=key&&ptr->link!=NULL)
123                     {
124                             ptr=ptr->link;
125                     }
126                     if(ptr->data!=key&&ptr->link==NULL)
127                     printf("search fails\n");
128                     else
129                     {
130                             temp->data=item;
131                             temp->link=ptr->link;
132                             ptr->link=temp;
133                     }
134             }
135     }
136     void deletionatbegining()
137     {
138             struct node *ptr;
139             if(head==NULL)
140             printf("Empty\n");
```

```
141             else
142             {
143                     ptr=head;
144                     head=ptr->link;
145                     printf("the data %d is deleted\n",ptr->data);
146                     free(ptr);
147             }
148     }
149     void deletionatend()
150     {
151             struct node *temp,*ptr;
152             if(head==NULL)
153             {
154                     printf("Empty\n");
155             }
156             else if(head->link==NULL)
157             {
158                     ptr=head;
159                     head=NULL;
160                     printf("the data %d is deleted\n",ptr->data);
161                     free(ptr);
162             }
163             else
164             {
165                     ptr=head;
166                     temp=head->link;
167                     while(temp->link!=NULL)
168                     {
```

```
169                             ptr=temp;
170                             temp=temp->link;
171                     }
172                     ptr->link=NULL;
173                     printf("the data %d is deleted\n",temp->data);
174                     free(temp);
175             }
176     }
177      void deletionatanyposition()
178      {
179             struct node *ptr,*temp;
180             int key;
181             printf("enter the data to be deleted\n");
182             scanf("%d",&key);
183             if(head==NULL)
184             {
185                     printf("Empty\n");
186             }
187             else if(head->link==NULL)
188             {
189                     deletionatbegining();
190                     head==NULL;
191             }
192             else if(head->data==key)
193             {
194                     deletionatbegining();
195             }
196             else
```

```
197             {
198                     temp=head;
199                     ptr=temp->link;
200                     while(ptr->data!=key&&ptr->link!=NULL)
201                     {
202                             temp=ptr;
203                             ptr=ptr->link;
204                     }
205                     if(ptr->data!=key&&ptr->link==NULL)
206                     {
207                             printf("search failed\n");
208                     }
209                     else
210                     {
211                             temp->link=ptr->link;
212                             printf("the data %d is deleted\n",ptr
213     >data);
214                             free(ptr);
215                     }
216             }
217       }
218        void display()
219         {
220             struct node *ptr;
221             ptr=head;
222             if(head==NULL)
223             {
224                     printf("Empty\n");
```

```
225             }
226             else
227             {
228                     printf("Linked list:");
229                     while(ptr!=NULL)
230                     {
231                             printf("%d ",ptr->data);
232                             ptr=ptr->link;
233                     }
234                     printf("\n");
235             }
236     }
```