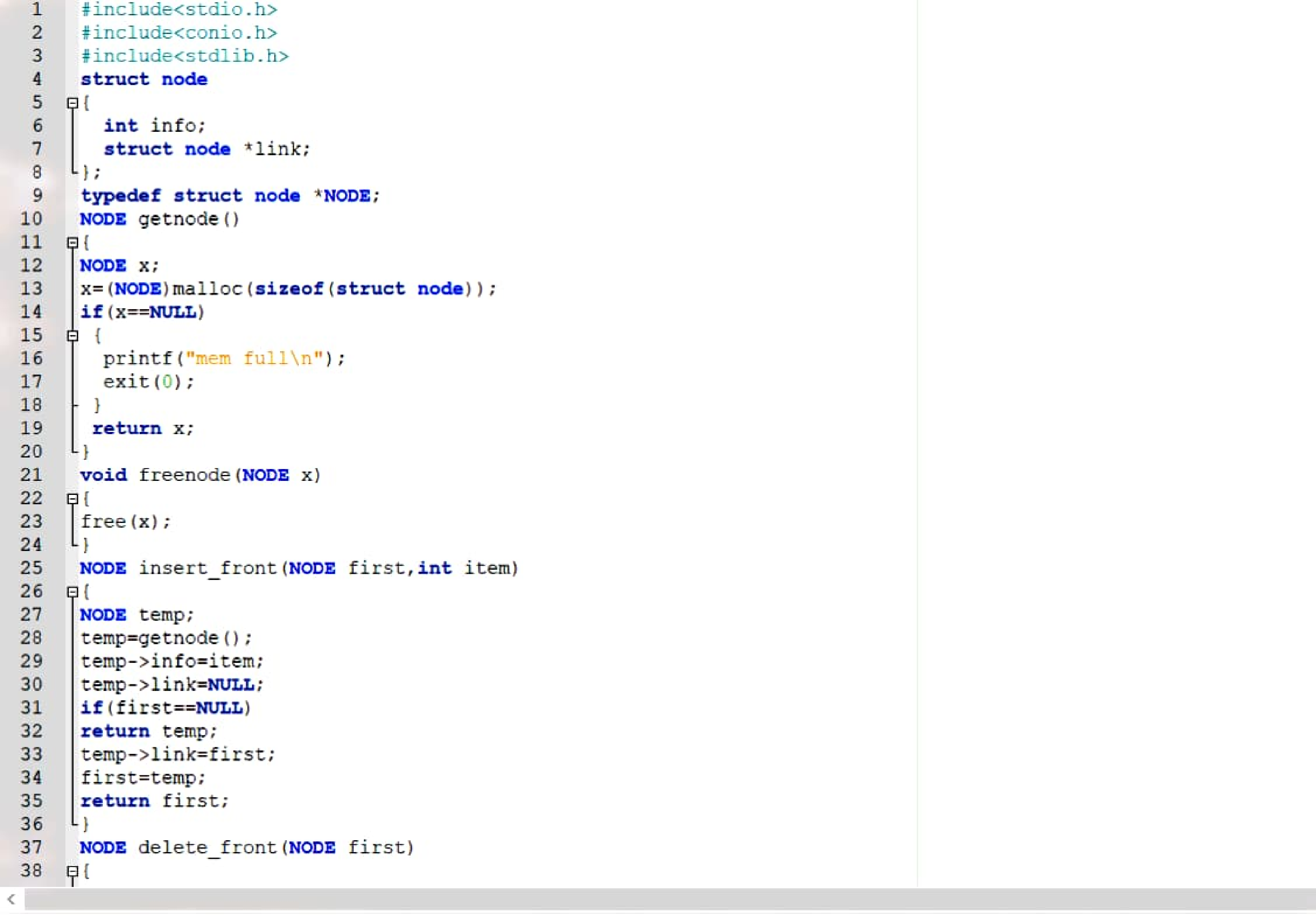
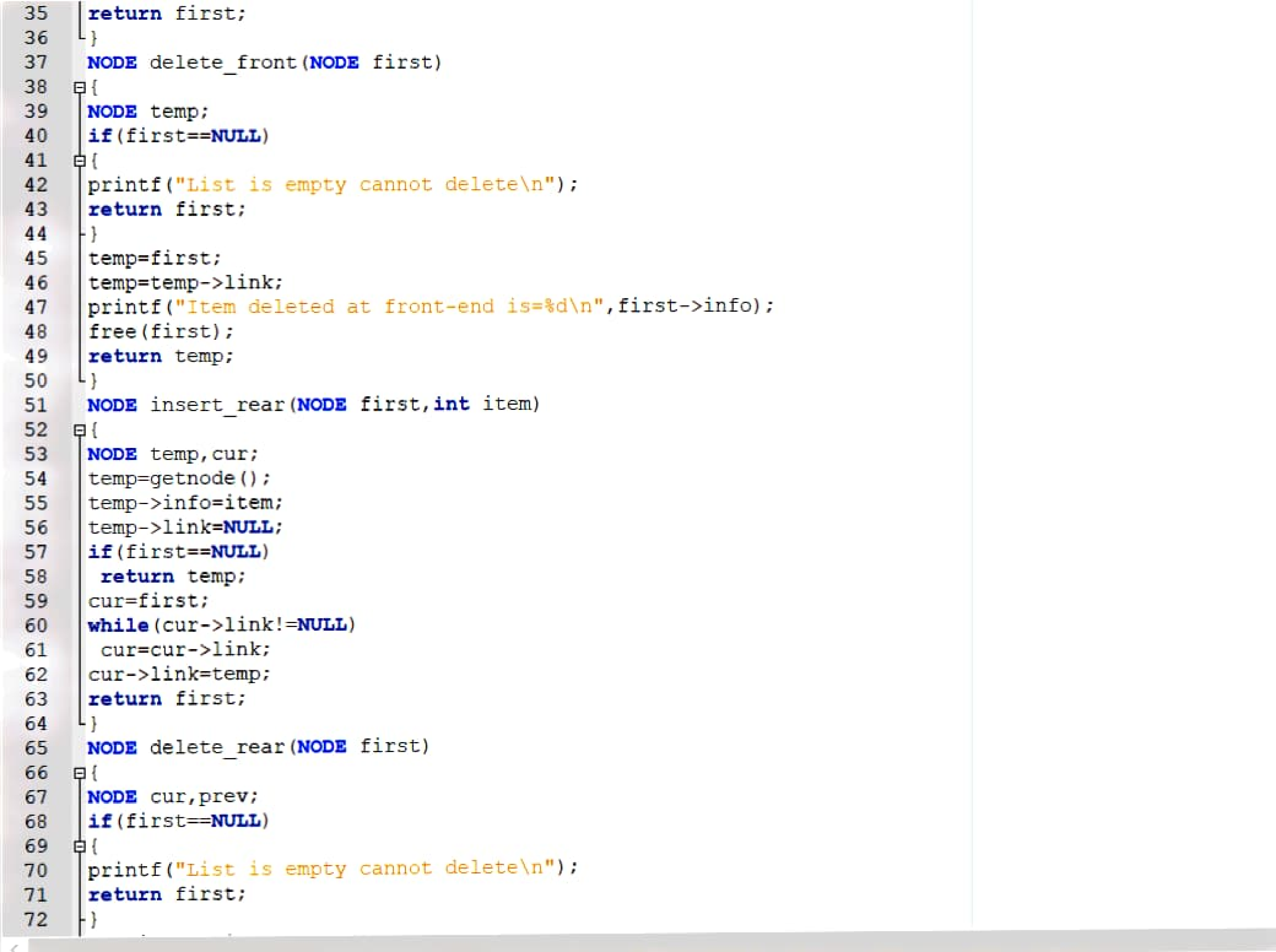
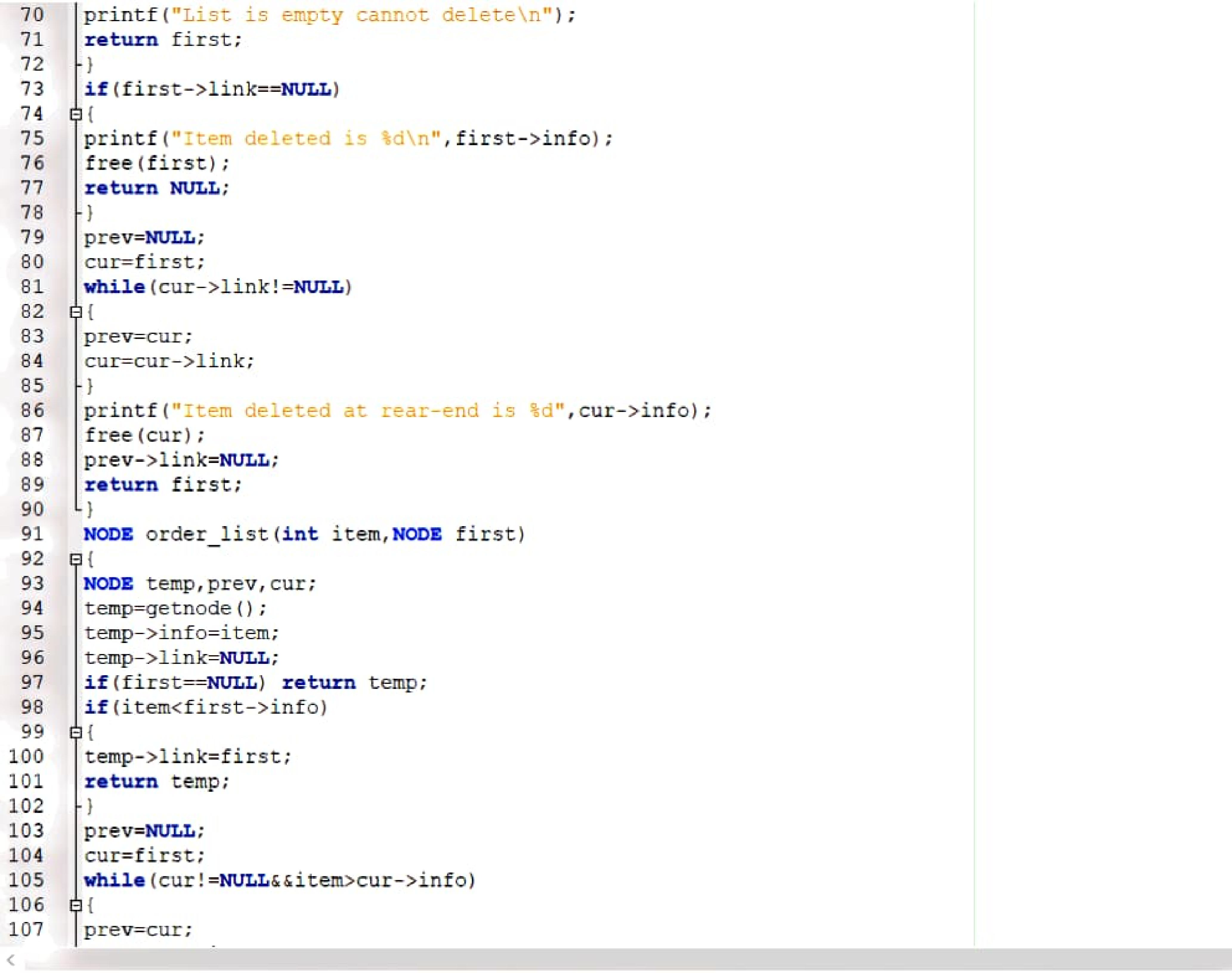


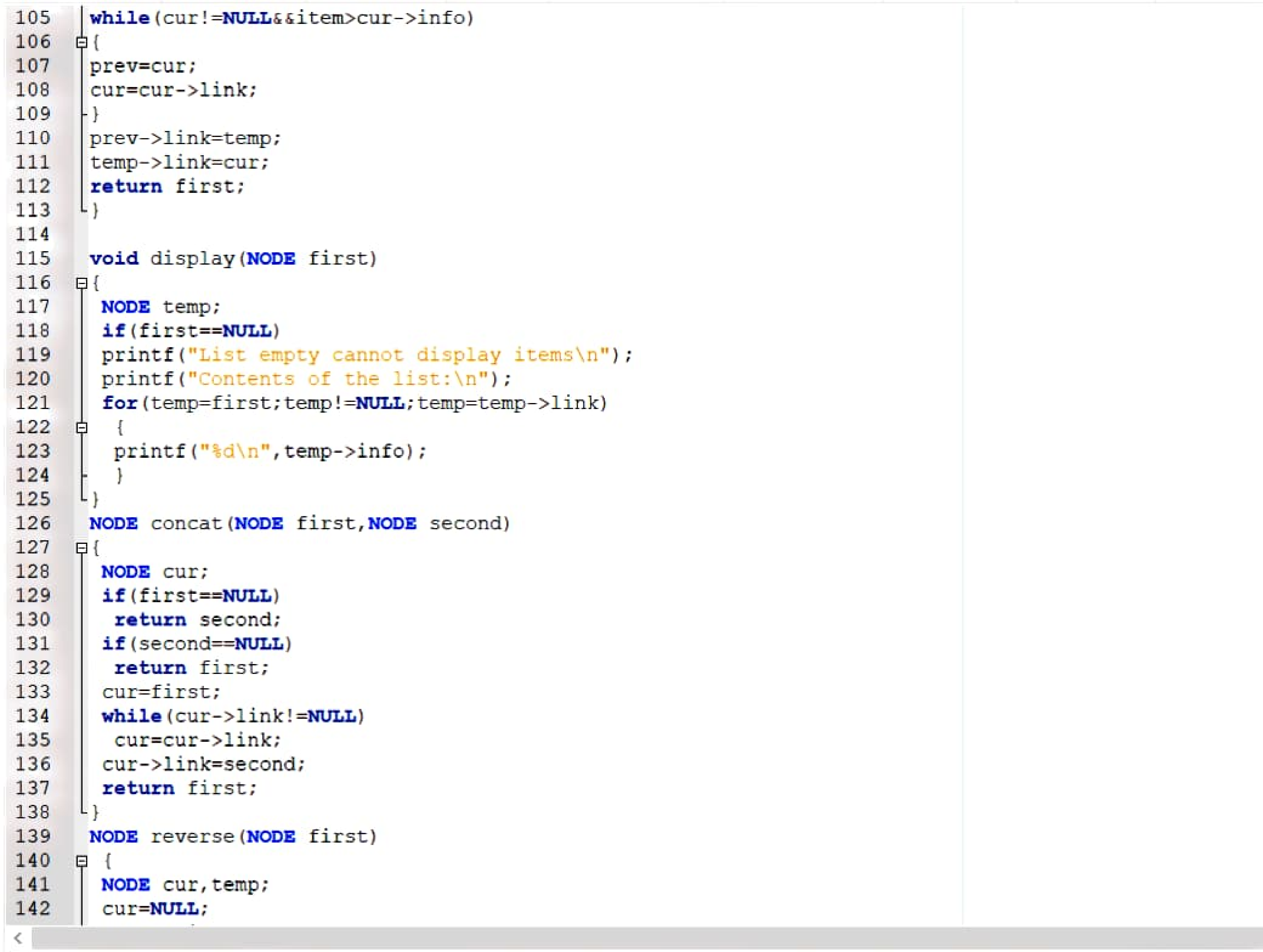
```c
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
struct node
{
  int info;
  struct node *link;
};
typedef struct node *NODE;
NODE getnode()
{
NODE x;
x=(NODE)malloc(sizeof(struct node));
if(x==NULL)
 {
  printf("mem full\n");
  exit(0);
 }
 return x;
}
void freenode(NODE x)
{
free(x);
}
NODE insert_front(NODE first,int item)
{
NODE temp;
temp=getnode();
temp->info=item;
temp->link=NULL;
if(first==NULL)
return temp;
temp->link=first;
first=temp;
return first;
}
NODE delete_front(NODE first)
{
```



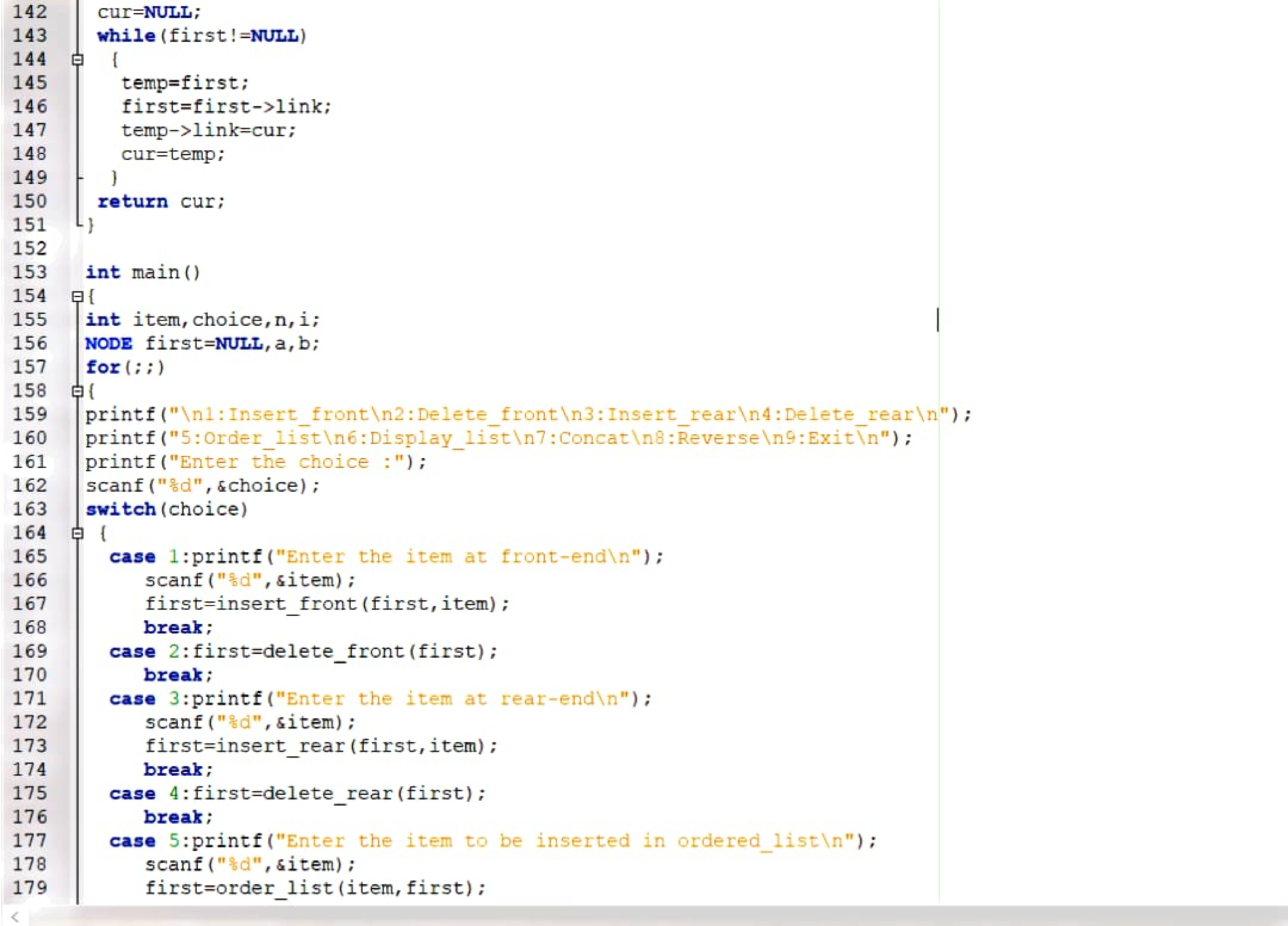
```c
return first;
}
NODE delete_front(NODE first)
{
NODE temp;
if(first==NULL)
{
printf("List is empty cannot delete\n");
return first;
}
temp=first;
temp=temp->link;
printf("Item deleted at front-end is=%d\n",first->info);
free(first);
return temp;
}
NODE insert_rear(NODE first,int item)
{
NODE temp,cur;
temp=getnode();
temp->info=item;
temp->link=NULL;
if(first==NULL)
 return temp;
cur=first;
while(cur->link!=NULL)
 cur=cur->link;
cur->link=temp;
return first;
}
NODE delete_rear(NODE first)
{
NODE cur,prev;
if(first==NULL)
{
printf("List is empty cannot delete\n");
return first;
}
```



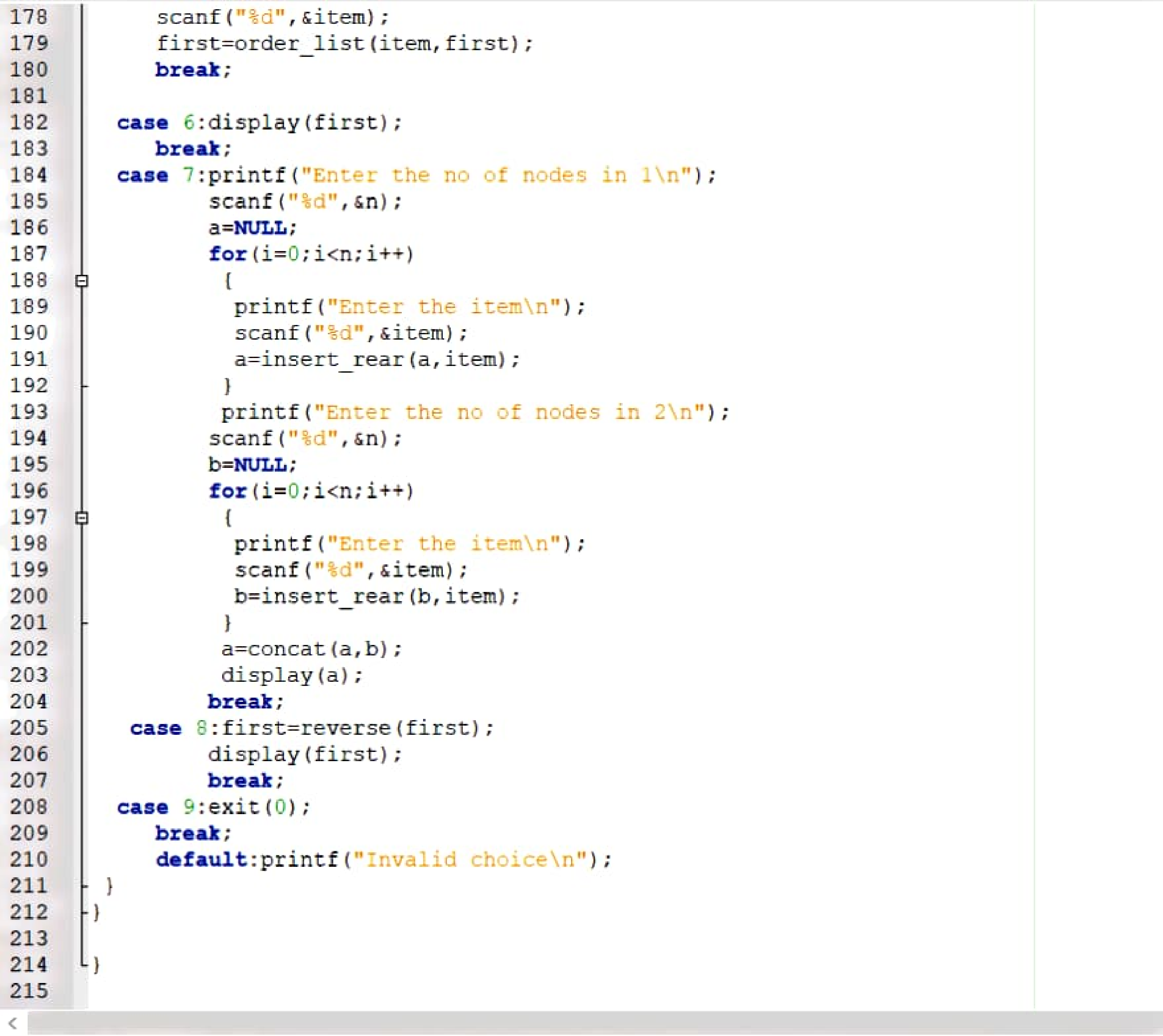
```c
printf("List is empty cannot delete\n");
return first;
}
if(first->link==NULL)
{
printf("Item deleted is %d\n",first->info);
free(first);
return NULL;
}
prev=NULL;
cur=first;
while(cur->link!=NULL)
{
prev=cur;
cur=cur->link;
}
printf("Item deleted at rear-end is %d",cur->info);
free(cur);
prev->link=NULL;
return first;
}
NODE order_list(int item,NODE first)
{
NODE temp,prev,cur;
temp=getnode();
temp->info=item;
temp->link=NULL;
if(first==NULL) return temp;
if(item<first->info)
{
temp->link=first;
return temp;
}
prev=NULL;
cur=first;
while(cur!=NULL&&item>cur->info)
{
prev=cur;
```



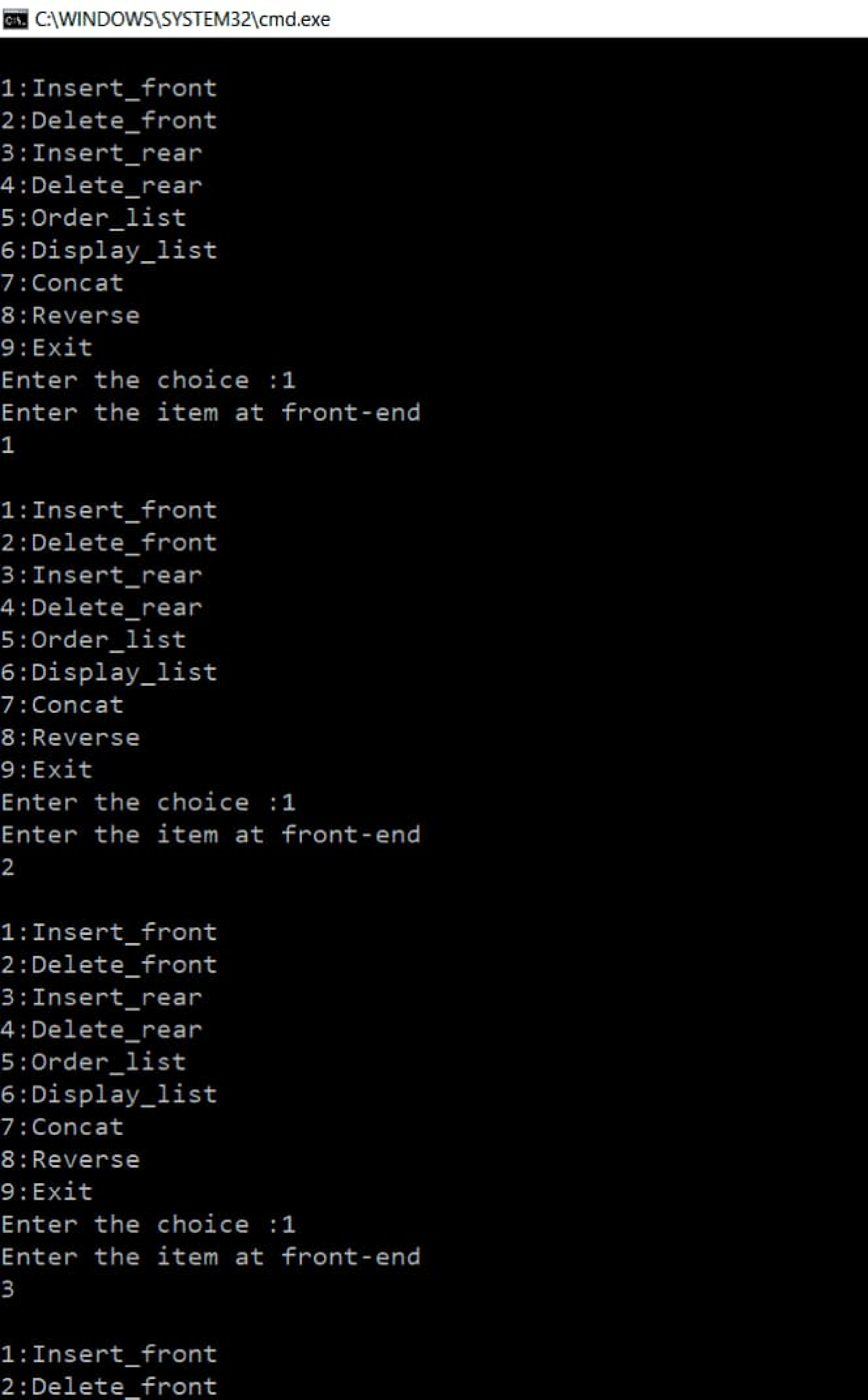
```
while(cur!=NULL&&item>cur->info)
{
prev=cur;
cur=cur->link;
}
prev->link=temp;
temp->link=cur;
return first;
}

void display(NODE first)
{
 NODE temp;
 if(first==NULL)
 printf("List empty cannot display items\n");
 printf("Contents of the list:\n");
 for(temp=first;temp!=NULL;temp=temp->link)
  {
  printf("%d\n",temp->info);
  }
}
NODE concat(NODE first,NODE second)
{
 NODE cur;
 if(first==NULL)
  return second;
 if(second==NULL)
  return first;
 cur=first;
 while(cur->link!=NULL)
  cur=cur->link;
 cur->link=second;
 return first;
}
NODE reverse(NODE first)
 {
 NODE cur,temp;
 cur=NULL;
```



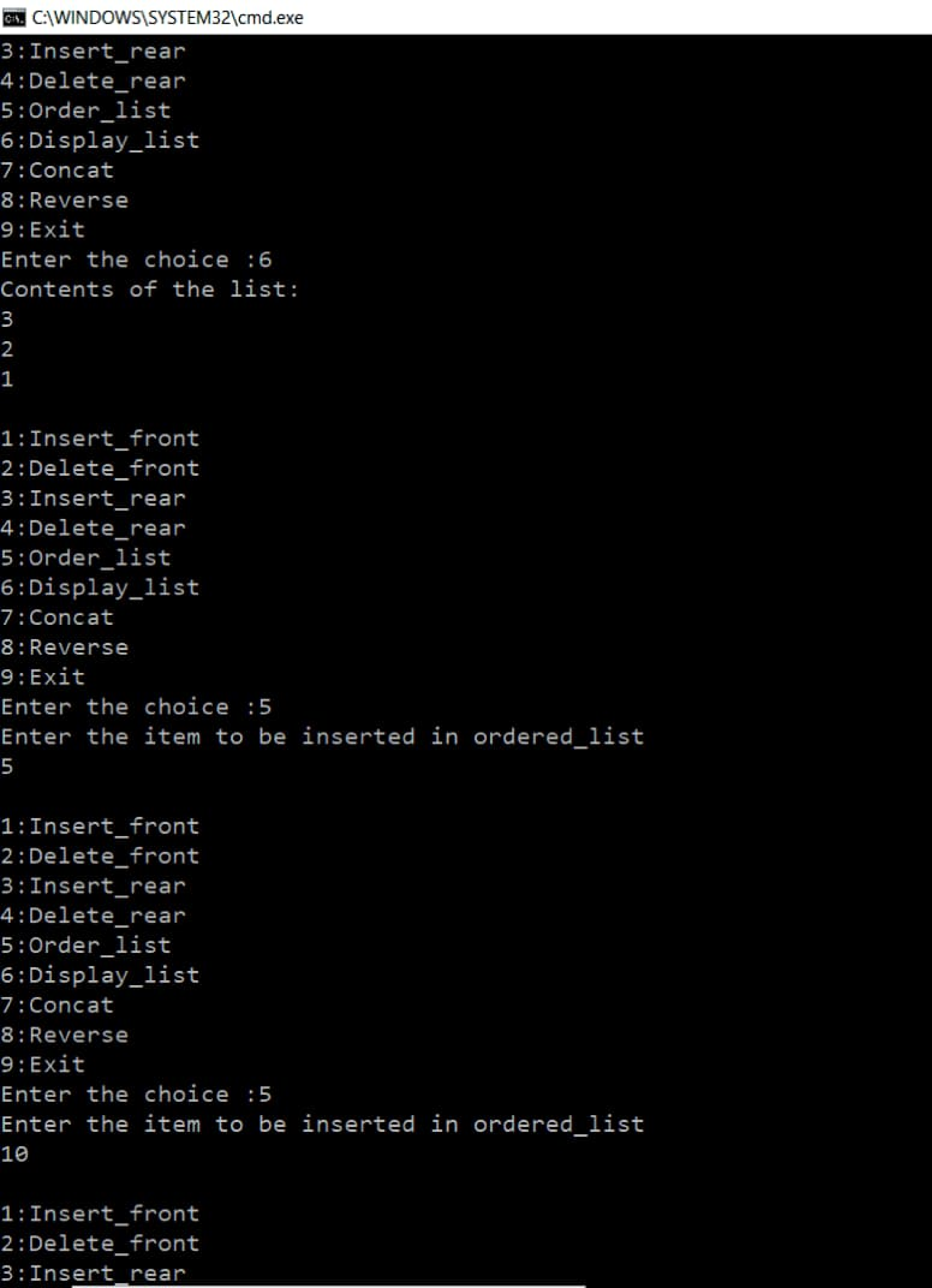
```
 cur=NULL;
 while(first!=NULL)
  {
   temp=first;
   first=first->link;
   temp->link=cur;
   cur=temp;
  }
 return cur;
}

int main()
{
int item,choice,n,i;
NODE first=NULL,a,b;
for(;;)
{
printf("\n1:Insert_front\n2:Delete_front\n3:Insert_rear\n4:Delete_rear\n");
printf("5:Order_list\n6:Display_list\n7:Concat\n8:Reverse\n9:Exit\n");
printf("Enter the choice :");
scanf("%d",&choice);
switch(choice)
 {
  case 1:printf("Enter the item at front-end\n");
     scanf("%d",&item);
     first=insert_front(first,item);
     break;
  case 2:first=delete_front(first);
     break;
  case 3:printf("Enter the item at rear-end\n");
     scanf("%d",&item);
     first=insert_rear(first,item);
     break;
  case 4:first=delete_rear(first);
     break;
  case 5:printf("Enter the item to be inserted in ordered_list\n");
     scanf("%d",&item);
     first=order_list(item,first);
```



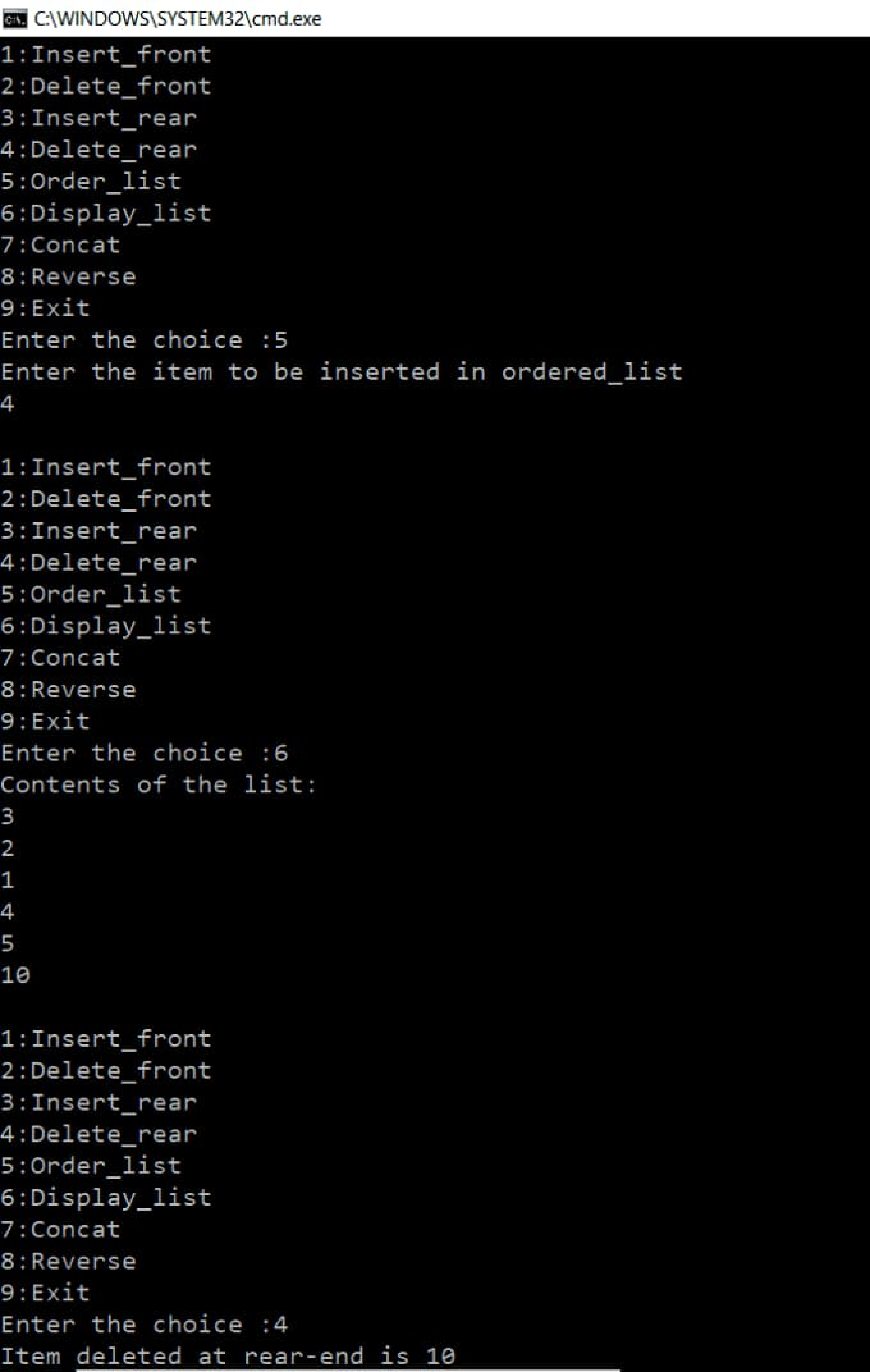
```
     scanf("%d",&item);
     first=order_list(item,first);
     break;

  case 6:display(first);
     break;
  case 7:printf("Enter the no of nodes in 1\n");
         scanf("%d",&n);
         a=NULL;
         for(i=0;i<n;i++)
          {
           printf("Enter the item\n");
           scanf("%d",&item);
           a=insert_rear(a,item);
          }
          printf("Enter the no of nodes in 2\n");
         scanf("%d",&n);
         b=NULL;
         for(i=0;i<n;i++)
          {
           printf("Enter the item\n");
           scanf("%d",&item);
           b=insert_rear(b,item);
          }
          a=concat(a,b);
          display(a);
         break;
   case 8:first=reverse(first);
         display(first);
         break;
  case 9:exit(0);
     break;
     default:printf("Invalid choice\n");
 }
}

}
```



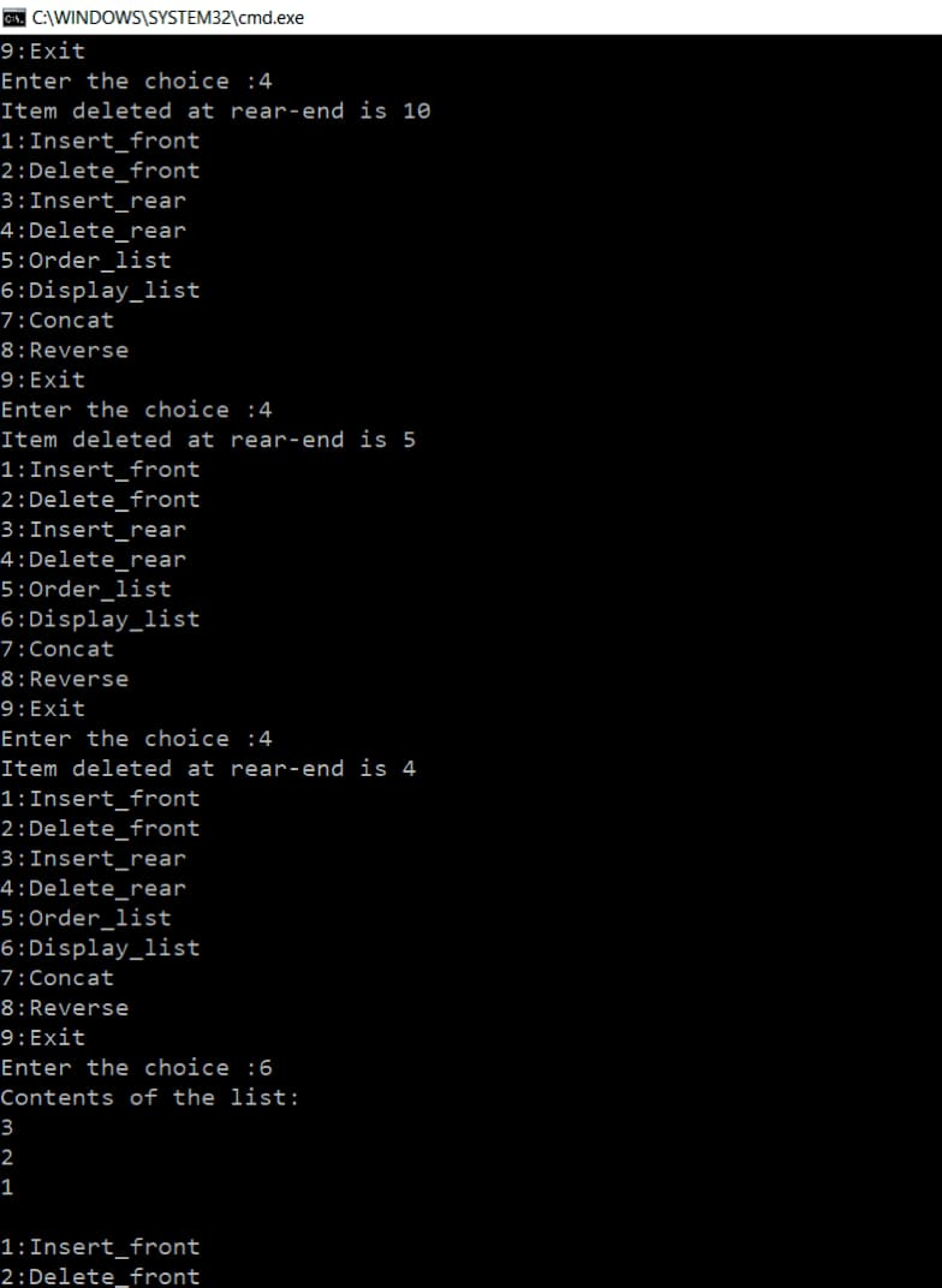
```

1:Insert_front
2:Delete_front
3:Insert_rear
4:Delete_rear
5:Order_list
6:Display_list
7:Concat
8:Reverse
9:Exit
Enter the choice :1
Enter the item at front-end
1

1:Insert_front
2:Delete_front
3:Insert_rear
4:Delete_rear
5:Order_list
6:Display_list
7:Concat
8:Reverse
9:Exit
Enter the choice :1
Enter the item at front-end
2

1:Insert_front
2:Delete_front
3:Insert_rear
4:Delete_rear
5:Order_list
6:Display_list
7:Concat
8:Reverse
9:Exit
Enter the choice :1
Enter the item at front-end
3

1:Insert_front
2:Delete_front
```



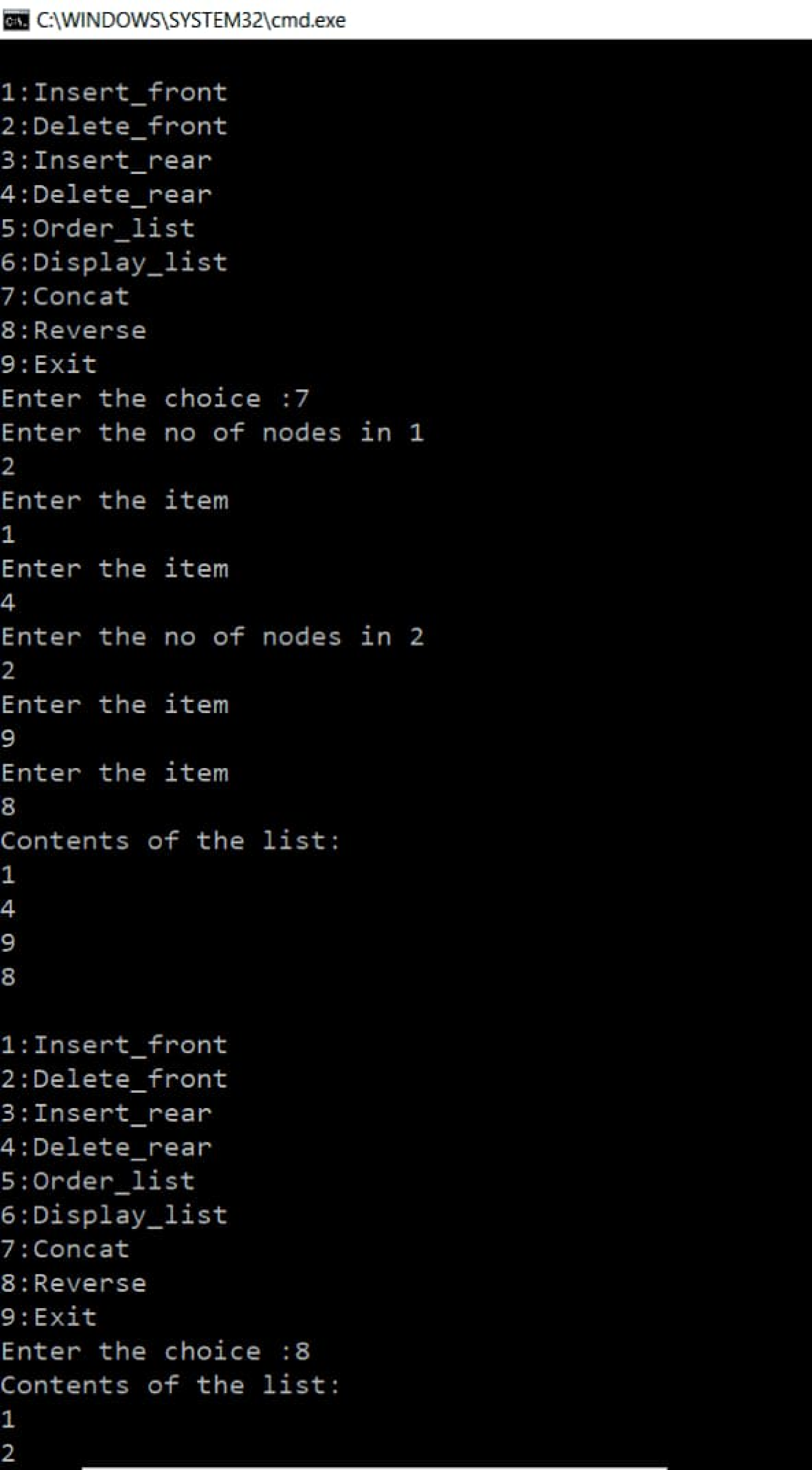
```
3:Insert_rear
4:Delete_rear
5:Order_list
6:Display_list
7:Concat
8:Reverse
9:Exit
Enter the choice :6
Contents of the list:
3
2
1

1:Insert_front
2:Delete_front
3:Insert_rear
4:Delete_rear
5:Order_list
6:Display_list
7:Concat
8:Reverse
9:Exit
Enter the choice :5
Enter the item to be inserted in ordered_list
5

1:Insert_front
2:Delete_front
3:Insert_rear
4:Delete_rear
5:Order_list
6:Display_list
7:Concat
8:Reverse
9:Exit
Enter the choice :5
Enter the item to be inserted in ordered_list
10

1:Insert_front
2:Delete_front
3:Insert_rear
```



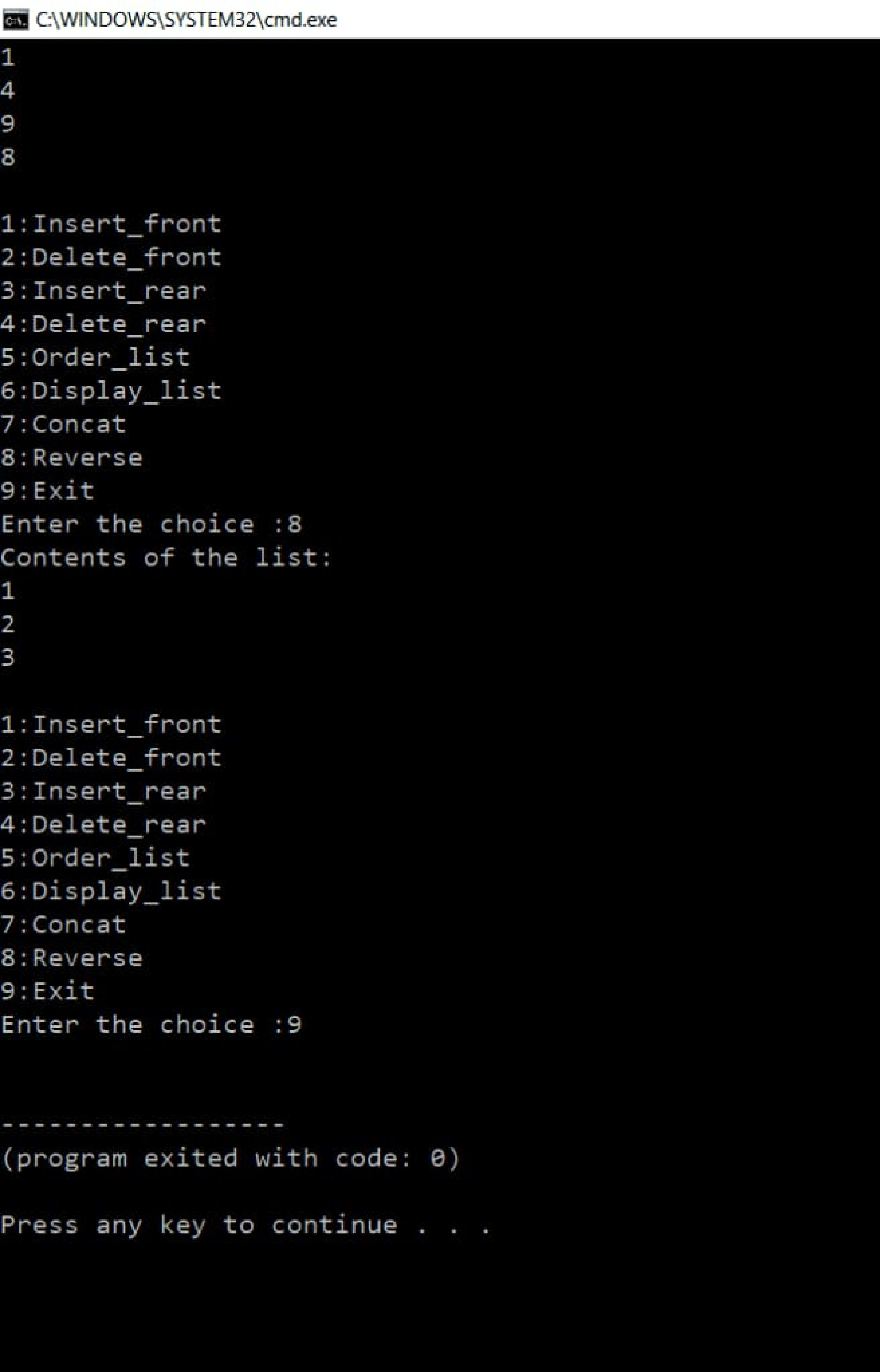
```
1:Insert_front
2:Delete_front
3:Insert_rear
4:Delete_rear
5:Order_list
6:Display_list
7:Concat
8:Reverse
9:Exit
Enter the choice :5
Enter the item to be inserted in ordered_list
4

1:Insert_front
2:Delete_front
3:Insert_rear
4:Delete_rear
5:Order_list
6:Display_list
7:Concat
8:Reverse
9:Exit
Enter the choice :6
Contents of the list:
3
2
1
4
5
10

1:Insert_front
2:Delete_front
3:Insert_rear
4:Delete_rear
5:Order_list
6:Display_list
7:Concat
8:Reverse
9:Exit
Enter the choice :4
Item deleted at rear-end is 10
```



```
9:Exit
Enter the choice :4
Item deleted at rear-end is 10
1:Insert_front
2:Delete_front
3:Insert_rear
4:Delete_rear
5:Order_list
6:Display_list
7:Concat
8:Reverse
9:Exit
Enter the choice :4
Item deleted at rear-end is 5
1:Insert_front
2:Delete_front
3:Insert_rear
4:Delete_rear
5:Order_list
6:Display_list
7:Concat
8:Reverse
9:Exit
Enter the choice :4
Item deleted at rear-end is 4
1:Insert_front
2:Delete_front
3:Insert_rear
4:Delete_rear
5:Order_list
6:Display_list
7:Concat
8:Reverse
9:Exit
Enter the choice :6
Contents of the list:
3
2
1

1:Insert_front
2:Delete_front
```



```
1:Insert_front
2:Delete_front
3:Insert_rear
4:Delete_rear
5:Order_list
6:Display_list
7:Concat
8:Reverse
9:Exit
Enter the choice :7
Enter the no of nodes in 1
2
Enter the item
1
Enter the item
4
Enter the no of nodes in 2
2
Enter the item
9
Enter the item
8
Contents of the list:
1
4
9
8

1:Insert_front
2:Delete_front
3:Insert_rear
4:Delete_rear
5:Order_list
6:Display_list
7:Concat
8:Reverse
9:Exit
Enter the choice :8
Contents of the list:
1
2
```



```
1
4
9
8

1:Insert_front
2:Delete_front
3:Insert_rear
4:Delete_rear
5:Order_list
6:Display_list
7:Concat
8:Reverse
9:Exit
Enter the choice :8
Contents of the list:
1
2
3

1:Insert_front
2:Delete_front
3:Insert_rear
4:Delete_rear
5:Order_list
6:Display_list
7:Concat
8:Reverse
9:Exit
Enter the choice :9


------------------
(program exited with code: 0)

Press any key to continue . . .
```