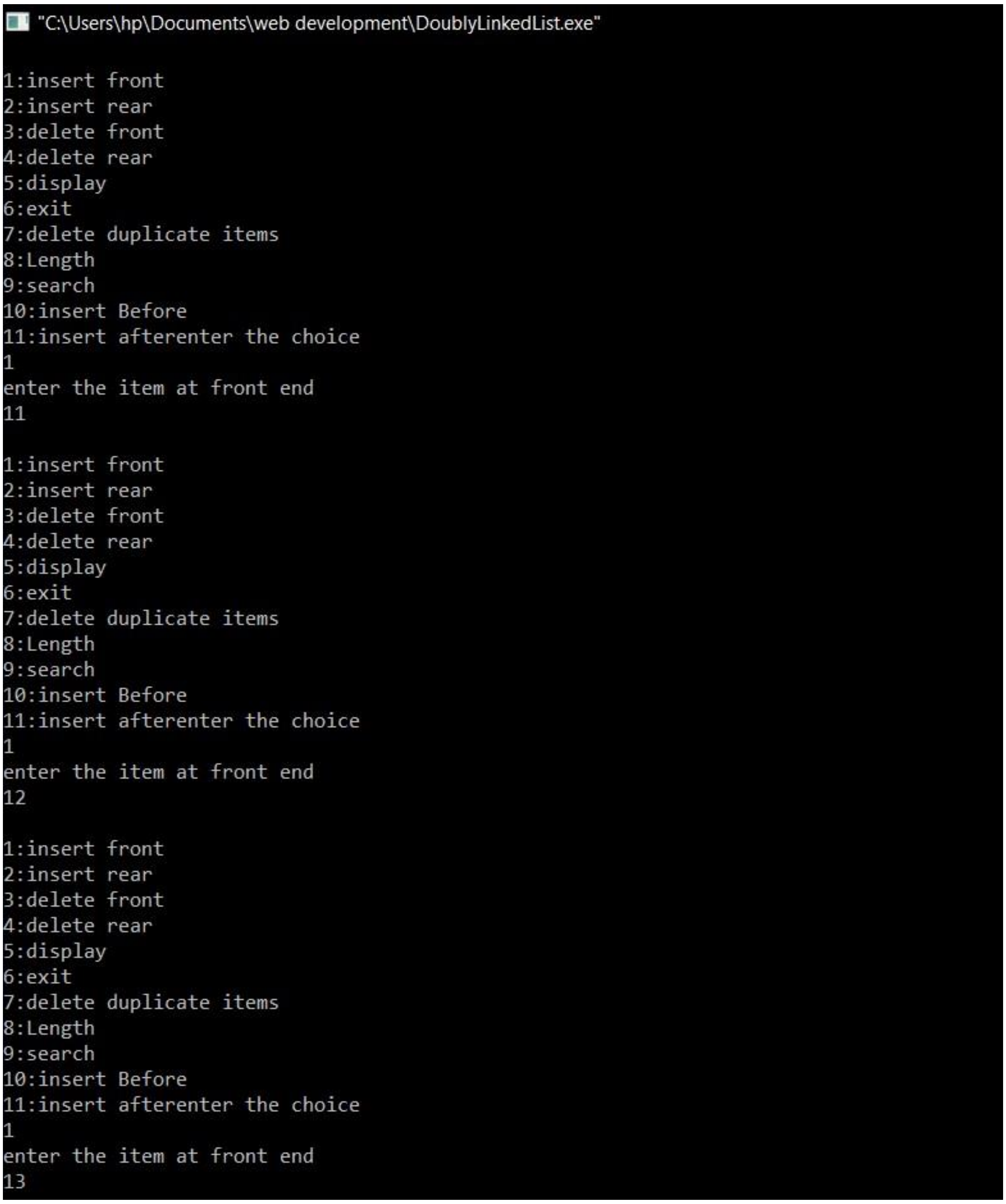
```
#include<stdio.h>
#include<stdlib.h>
struct node {
  int info;
    struct node *llink;
      struct node *rlink;
        };
typedef struct node *NODE;
NODE getnode()
{
  NODE x;
  x=(NODE)malloc(sizeof(struct node));
  if(x==NULL) {  printf("mem full\n");
  exit(0);  }
  return  x; }
  void freenode(NODE x)
  {
      free(x);
  }
NODE dinsert_front(int item,NODE head)
{
  NODE temp,cur; temp=getnode();
  temp->info=item;
  cur=head->rlink;
  head->rlink=temp;
  temp->llink=head;
  temp->rlink=cur;
  cur->llink=temp;
  return  head;
}
NODE dinsert_rear(int item,NODE head)
{
  NODE temp,cur;
  temp=getnode();
  temp->info=item;
  cur=head->llink;
  head->llink=temp;
  temp->rlink=head;
  temp->llink=cur;
  cur->rlink=temp;
  return head;
}
NODE ddelete_front(NODE head)
{
  NODE cur,next;
  if(head->rlink==head)
      {
printf("dq empty\n");
return head;
}
cur=head->rlink;
next=cur->rlink;
head->rlink=next;
next->llink=head;
printf("the node deleted is %d",cur->info);
freenode(cur);
return head;
}
NODE ddelete_rear(NODE head)
{
  NODE cur,prev;
  if(head->rlink==head)
      {
          printf("dq empty\n");
  return head; }
  cur=head->llink;
  prev=cur->llink;
  head->llink=prev;
  prev->rlink=head;
  printf("the node deleted is %d",cur->info);
  freenode(cur);
  return head;
  }
void display(NODE head)
  {
      NODE temp;
  if(head->rlink==head)
      {
  printf("dq empty\n");
  return;
  }
  printf("contents of dq\n");
  temp=head->rlink;
  while(temp!=head) {
      printf("%d\t",temp->info);
      temp=temp->rlink; } printf("\n");
      }
int length(NODE first)
{
  NODE temp=first->rlink;
  int ct=0;
  while(temp!=first)
  {
      ct++;
      temp=temp->rlink;
  }
    printf("Length of list is %d",ct);return ct;
}
NODE search(NODE first)
{
   NODE temp=first->rlink;
   int count=0,key,flag=0;
   printf("Enter the KEY :");
   scanf("%d",&key);
   while(temp!=first)
   {         count++;
       if(temp->info==key)
       {
           flag=1;
           printf("key %d found in position %d",key,count);
       }temp=temp->rlink;
   }
   if(flag==0)
   {
       printf("Key is not found in list");
   }
   return  first;


}
NODE insert_after(NODE first)
{      int key,item,flag=0;
    printf("Enter the element :");
    scanf("%d",&key);NODE temp=first->rlink;
    NODE ptr=getnode();
    while(temp != first)
    {
        if(temp->info==key)
        {
           printf("Enter the item need to be inserted:");
           scanf("%d",&item);
           ptr->info=item;
           ptr->rlink=temp->rlink;
           ptr->llink=temp;
           temp->rlink=ptr;
           flag=1;
           return first;
        }temp=temp->rlink;
    }
    if(flag==0)
        printf("There is no such element");
    return first;
}
NODE insert_before(NODE first)
{     int key,item,flag=0;
    printf("Enter the element :");
    scanf("%d",&key);
    NODE temp=first->rlink;
    NODE ptr=getnode();
    while(temp != first)
    {
        if(temp->info==key)
        {
           printf("Enter the item need to be inserted:");
           scanf("%d",&item);
           ptr->info=item;
           //temp->rlink=ptr;
           ptr->rlink=temp;
           ptr->llink=temp->llink;
           temp->llink=ptr;
           //ptr->rlink=temp;
```

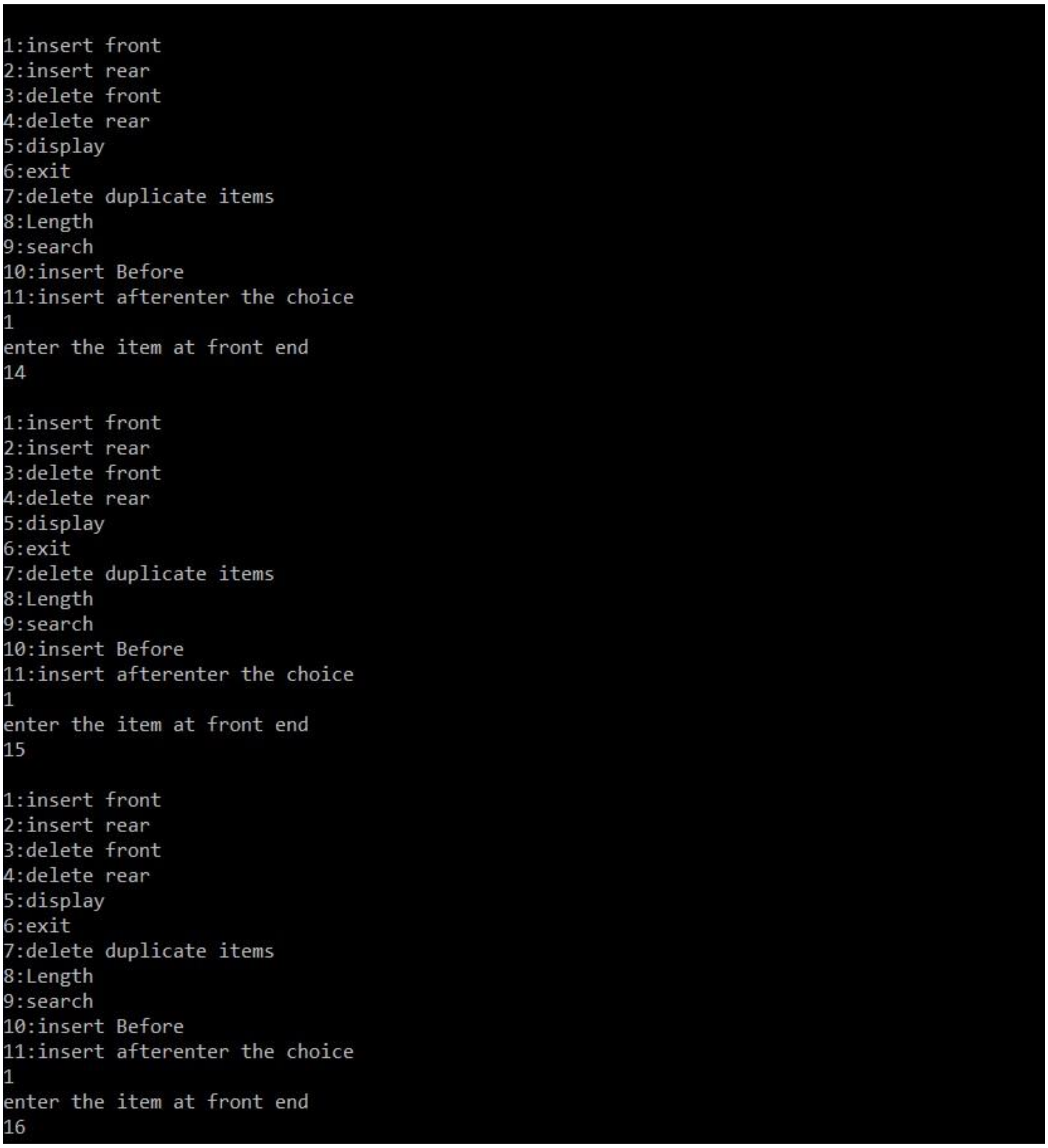
```
          flag=1;
          return first;
        }
        temp=temp->rlink;
    }
    if(flag==0)
        printf("There is no such element");
    return first;
}

void delete_dup(NODE head)
{
    NODE cur,temp,ptr,prev;
    if(head->rlink==head)
    {
        printf("List is empty\n");
        return ;
    }
    temp=head->rlink;
    cur=head->rlink;

    while(temp!=head)
    {
        prev=cur;
        cur=temp->rlink;
        while(cur!=head){

        if(temp->info==cur->info)
        {
          ptr=cur->rlink;ptr->llink=cur->llink;
          ptr=cur->llink;ptr->rlink=cur->rlink;
          freenode(cur);
        }
       cur=cur->rlink;
       }
       temp=temp->rlink;
    }
    return ;}

void main() {
      NODE head,last;
      int item, choice,len;
      head=getnode();
       head->rlink=head;
       head->llink=head;
          for(;;)
             {
printf("\n1:insert front\n2:insert rear\n3:delete front\n4:delete rear\n5:display\n6:exit\n7:delete duplicate
items\n8:Length\n9:search\n10:insert Before\n11:insert after\n");
printf("enter the choice\n");
 scanf("%d",&choice);
 switch(choice)
   {  case  1: printf("enter the item at front end\n");
   scanf("%d",&item);
      last=dinsert_front(item,head);
      break;
 case 2: printf("enter the item at rear end\n");
 scanf("%d",&item);
 last=dinsert_rear(item,head);
    break;
       case 3:last=ddelete_front(head);
          break;
 case 4: last=ddelete_rear(head);
    break;
       case 5: display(head);
```

```
break;
case 7:delete_dup(head);break;
case 8:len=length(head);break;
case 9:head=search(head);break;
case 10:head=insert_after(head);break;
case 11:head=insert_before(head);break;
default:break;
}
}
```

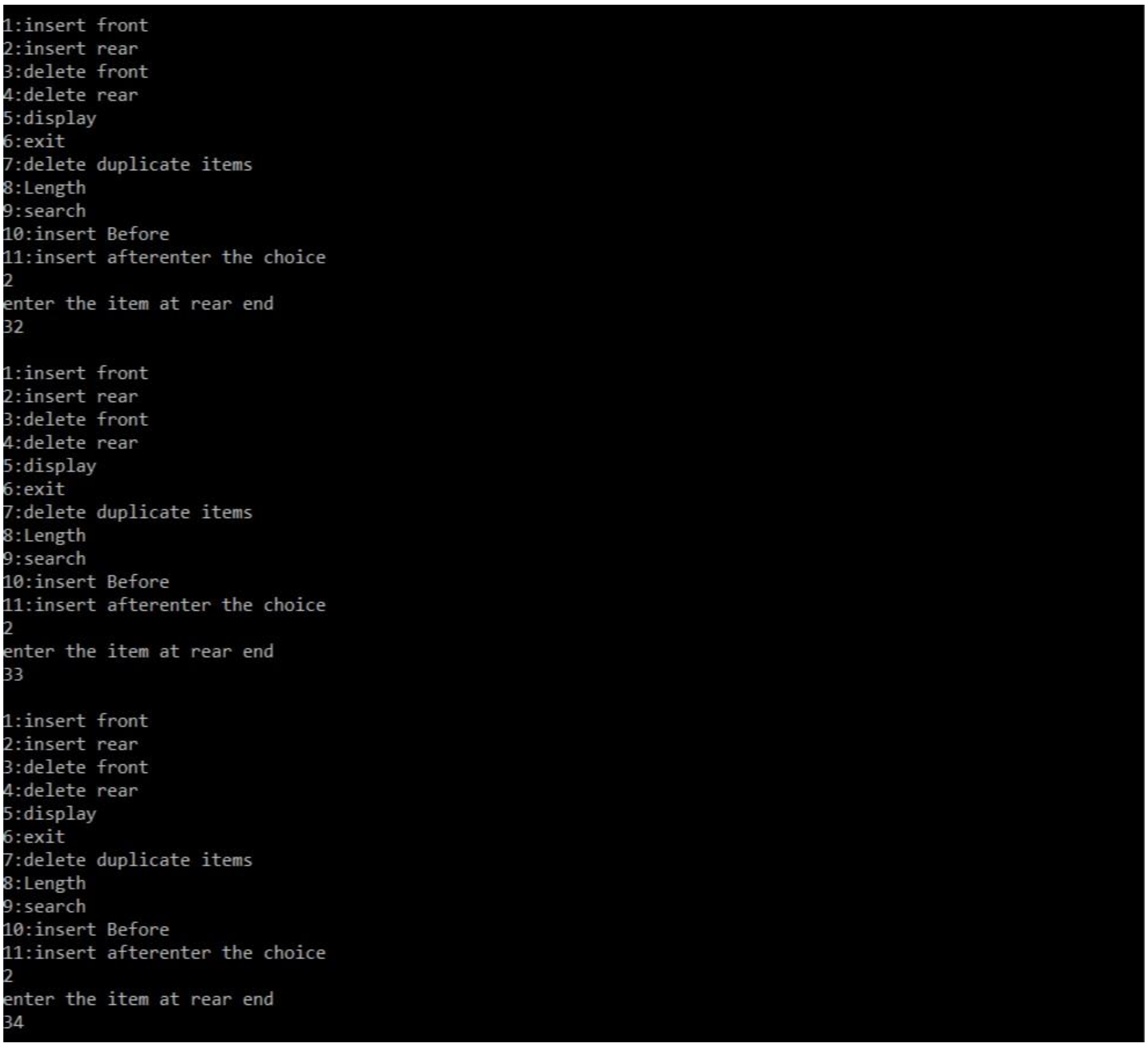
OUTPUT:



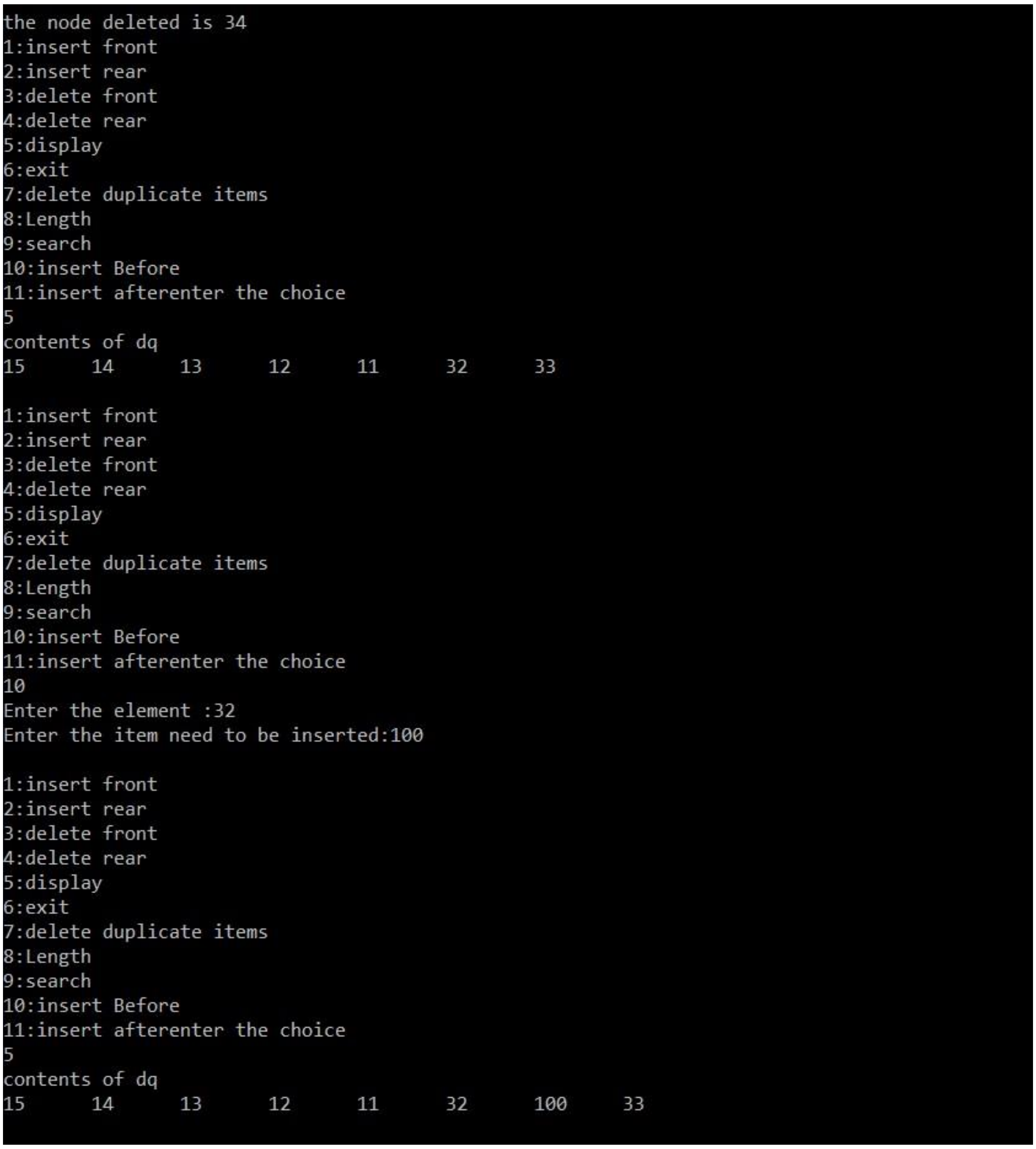
```
"C:\Users\hp\Documents\web development\DoublyLinkedList.exe"

1:insert front
2:insert rear
3:delete front
4:delete rear
5:display
6:exit
7:delete duplicate items
8:Length
9:search
10:insert Before
11:insert afterenter the choice
1
enter the item at front end
11

1:insert front
2:insert rear
3:delete front
4:delete rear
5:display
6:exit
7:delete duplicate items
8:Length
9:search
10:insert Before
11:insert afterenter the choice
1
enter the item at front end
12

1:insert front
2:insert rear
3:delete front
4:delete rear
5:display
6:exit
7:delete duplicate items
8:Length
9:search
10:insert Before
11:insert afterenter the choice
1
enter the item at front end
13
```



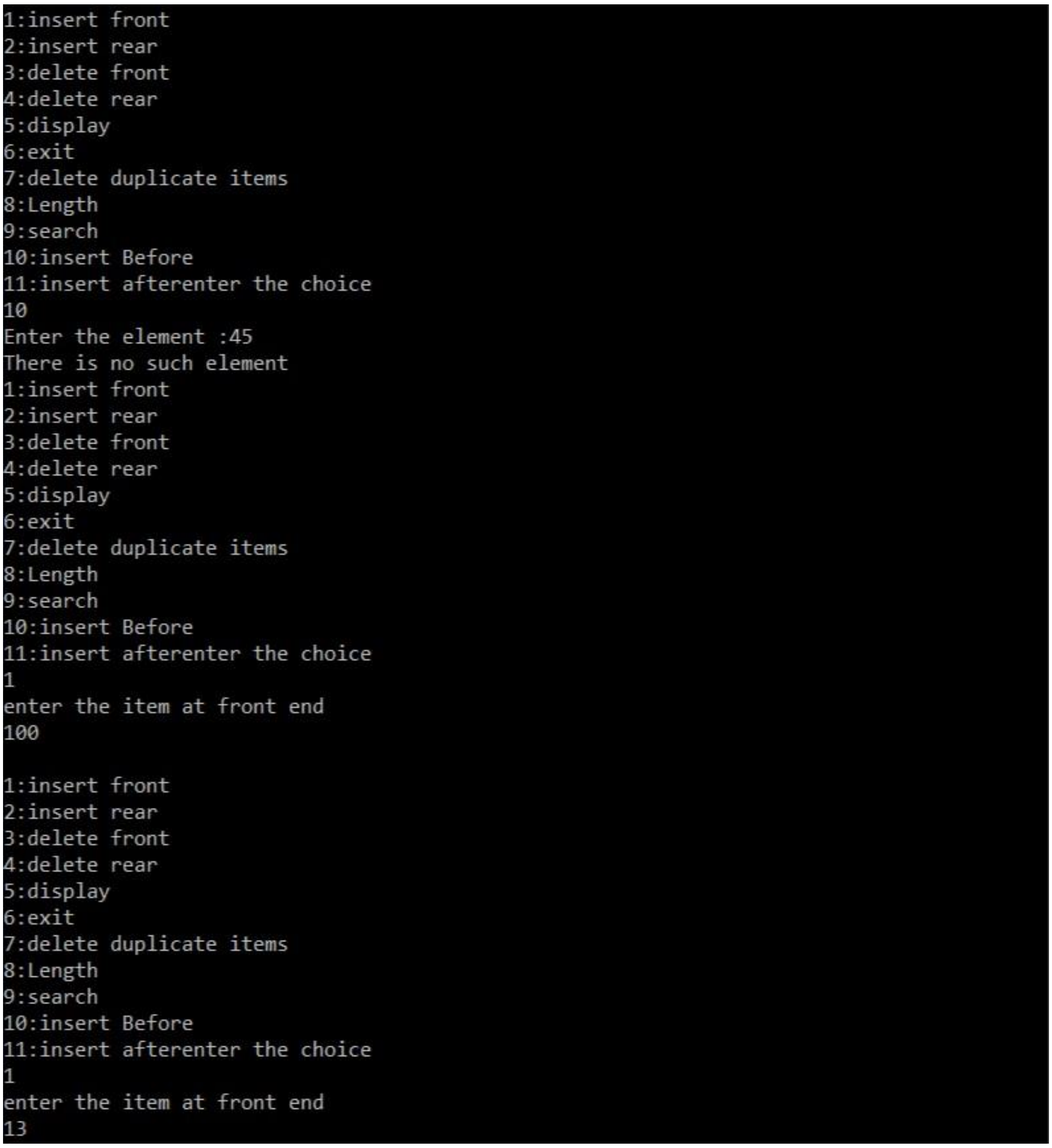
```
1:insert front
2:insert rear
3:delete front
4:delete rear
5:display
6:exit
7:delete duplicate items
8:Length
9:search
10:insert Before
11:insert afterenter the choice
1
enter the item at front end
14

1:insert front
2:insert rear
3:delete front
4:delete rear
5:display
6:exit
7:delete duplicate items
8:Length
9:search
10:insert Before
11:insert afterenter the choice
1
enter the item at front end
15

1:insert front
2:insert rear
3:delete front
4:delete rear
5:display
6:exit
7:delete duplicate items
8:Length
9:search
10:insert Before
11:insert afterenter the choice
1
enter the item at front end
16
```



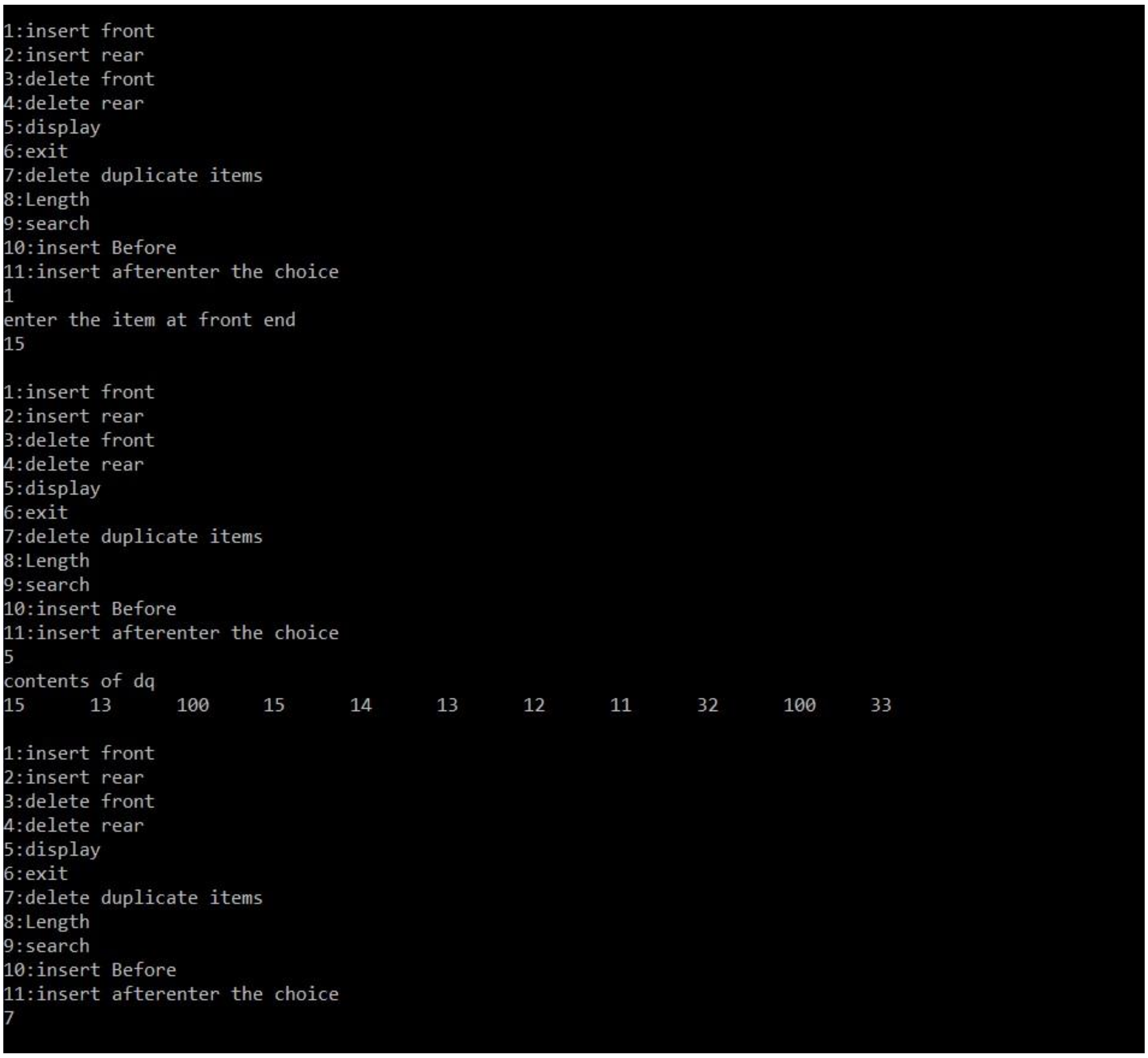
```
1:insert front
2:insert rear
3:delete front
4:delete rear
5:display
6:exit
7:delete duplicate items
8:Length
9:search
10:insert Before
11:insert afterenter the choice
2
enter the item at rear end
32

1:insert front
2:insert rear
3:delete front
4:delete rear
5:display
6:exit
7:delete duplicate items
8:Length
9:search
10:insert Before
11:insert afterenter the choice
2
enter the item at rear end
33

1:insert front
2:insert rear
3:delete front
4:delete rear
5:display
6:exit
7:delete duplicate items
8:Length
9:search
10:insert Before
11:insert afterenter the choice
2
enter the item at rear end
34
```



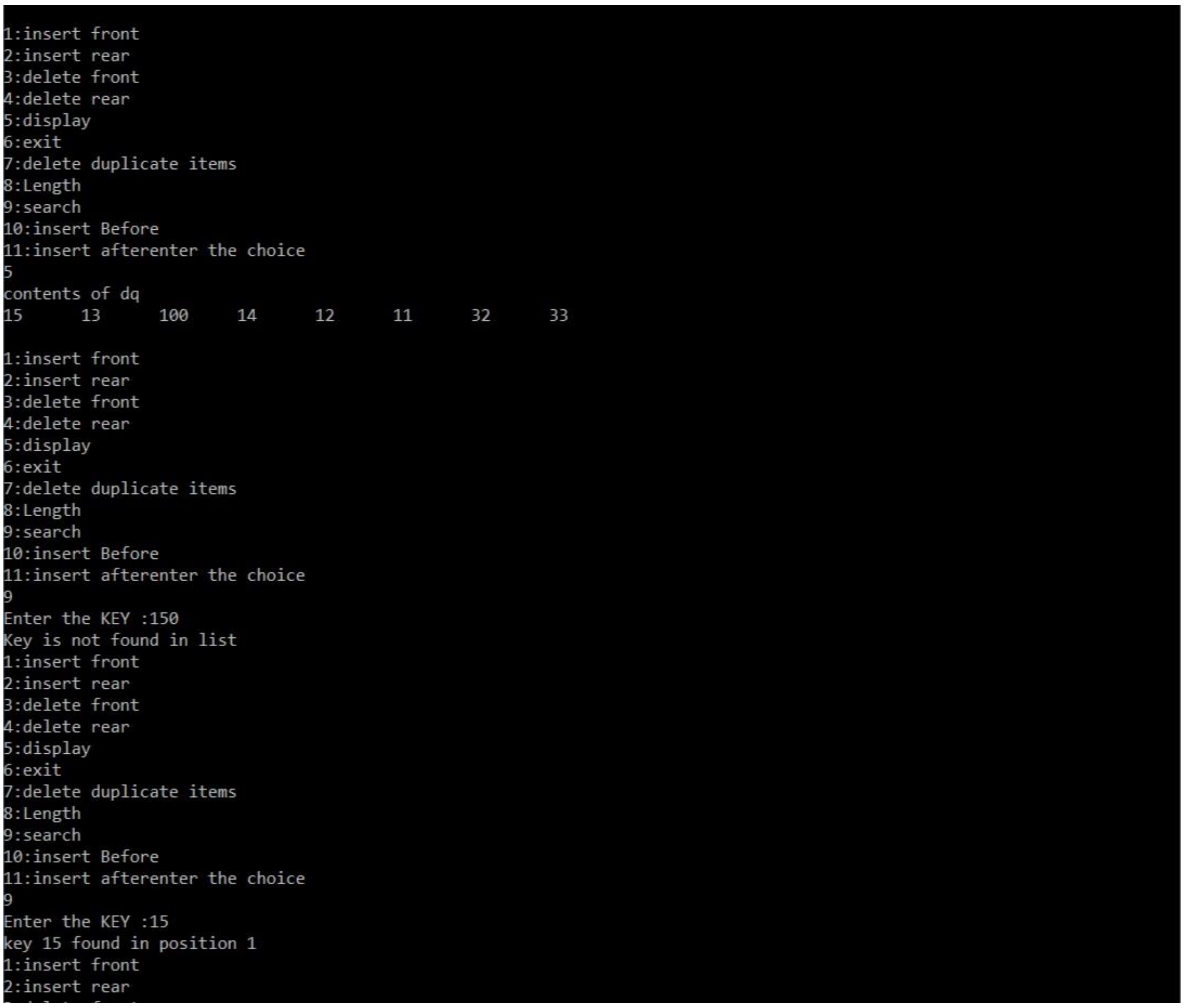
```
the node deleted is 34
1:insert front
2:insert rear
3:delete front
4:delete rear
5:display
6:exit
7:delete duplicate items
8:Length
9:search
10:insert Before
11:insert afterenter the choice
5
contents of dq
15      14      13      12      11      32      33

1:insert front
2:insert rear
3:delete front
4:delete rear
5:display
6:exit
7:delete duplicate items
8:Length
9:search
10:insert Before
11:insert afterenter the choice
10
Enter the element :32
Enter the item need to be inserted:100

1:insert front
2:insert rear
3:delete front
4:delete rear
5:display
6:exit
7:delete duplicate items
8:Length
9:search
10:insert Before
11:insert afterenter the choice
5
contents of dq
15      14      13      12      11      32      100     33
```



```
1:insert front
2:insert rear
3:delete front
4:delete rear
5:display
6:exit
7:delete duplicate items
8:Length
9:search
10:insert Before
11:insert afterenter the choice
10
Enter the element :45
There is no such element
1:insert front
2:insert rear
3:delete front
4:delete rear
5:display
6:exit
7:delete duplicate items
8:Length
9:search
10:insert Before
11:insert afterenter the choice
1
enter the item at front end
100

1:insert front
2:insert rear
3:delete front
4:delete rear
5:display
6:exit
7:delete duplicate items
8:Length
9:search
10:insert Before
11:insert afterenter the choice
1
enter the item at front end
13
```



```
1:insert front
2:insert rear
3:delete front
4:delete rear
5:display
6:exit
7:delete duplicate items
8:Length
9:search
10:insert Before
11:insert afterenter the choice
1
enter the item at front end
15

1:insert front
2:insert rear
3:delete front
4:delete rear
5:display
6:exit
7:delete duplicate items
8:Length
9:search
10:insert Before
11:insert afterenter the choice
5
contents of dq
15      13      100     15      14      13      12      11      32      100     33

1:insert front
2:insert rear
3:delete front
4:delete rear
5:display
6:exit
7:delete duplicate items
8:Length
9:search
10:insert Before
11:insert afterenter the choice
7
```



```
1:insert front
2:insert rear
3:delete front
4:delete rear
5:display
6:exit
7:delete duplicate items
8:Length
9:search
10:insert Before
11:insert afterenter the choice
5
contents of dq
15      13      100     14      12      11      32      33

1:insert front
2:insert rear
3:delete front
4:delete rear
5:display
6:exit
7:delete duplicate items
8:Length
9:search
10:insert Before
11:insert afterenter the choice
9
Enter the KEY :150
Key is not found in list
1:insert front
2:insert rear
3:delete front
4:delete rear
5:display
6:exit
7:delete duplicate items
8:Length
9:search
10:insert Before
11:insert afterenter the choice
9
Enter the KEY :15
key 15 found in position 1
1:insert front
2:insert rear
```

}