}

23p-0685 Raqeeb Task 4

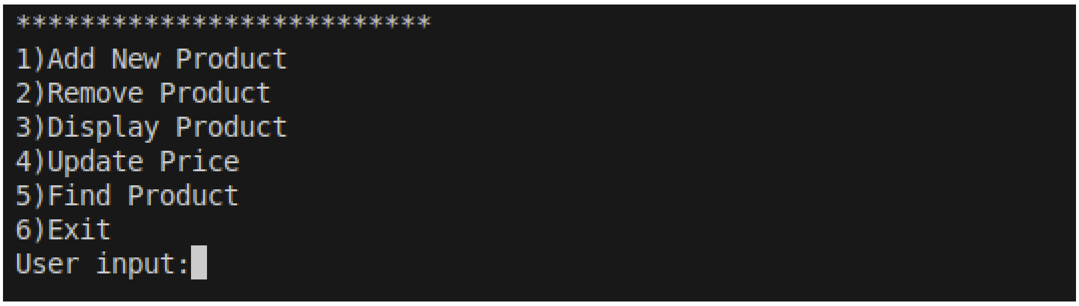
Question 1:

```
#include<iostream>

using namespace std;

class Nodes
{
public:
Nodes *Next;
Nodes *Prev;
int ID;
string Name;
int Quantity;
int Price;

Nodes(int value, string name, int quantity, int price)
{
ID= value;
Name=name;
Quantity=quantity;
Next=NULL;
Prev=NULL;
Price= price;
}


};


class Double_Linked_list
{
private:
int length;
Nodes *head;

public:
Double_Linked_list()
{
length=0;
head=NULL;

}

void insertion(int value, string name, int quantity,int price)
{
//Insertion with sorting the ID's
Nodes *ptr1=head;
Nodes *ptr2=head;
Nodes *new_node= new Nodes(value,name,quantity,price);
bool checker=false;
if(head!=NULL)
{
for(int i=1;i<=length;i++)
{
if(ptr2->ID > new_node->ID)
{
new_node->Next=ptr2;

if(ptr2==head)
{
head= new_node;
ptr2->Prev=new_node;
}
else
{
ptr1->Next=new_node;
new_node->Prev=ptr1;
ptr2->Prev=new_node;

}
checker=true;
break;
}
ptr1=ptr2;
if(ptr2->Next !=NULL)
ptr2=ptr2->Next;

}
if(!checker)
{
ptr2->Next=new_node;
new_node->Prev=ptr2;
}
}
else
{
head=new_node;
}


length+=1;
}

void remove(int ID)
{
if(length==0)
{
cout<<"\nTheir is no product exist !!!!First add the products\n"<<endl;
return;
}
Nodes *Curr_ptr=head;
Nodes *Slow_ptr=head;

for(int i=1;i<length;i++)
{
if(Curr_ptr->ID == ID)
{
if(Curr_ptr == head)
{
head=Curr_ptr->Next;
delete Curr_ptr;
length-=1;

return;
}
else
{
Slow_ptr->Next= Curr_ptr->Next;
Curr_ptr->Next->Prev=Slow_ptr;
delete Curr_ptr;
length-=1;

return;
}
}
else
{
Slow_ptr=Curr_ptr;
Curr_ptr=Curr_ptr->Next;

}
}
if(Curr_ptr->Next == NULL)
{
cout<<"\nNo such ID exist!!!"<<endl;
return;
}
}

void Doubly_display()
{
if(length==0)
{
cout<<"Their are not products to show first add the products details!!!!!"<<endl;
return;
}
cout<<"\nAll Products Details:\n";
Nodes *Curr_ptr=head;
for (int i=1; i<=length;i++)
```

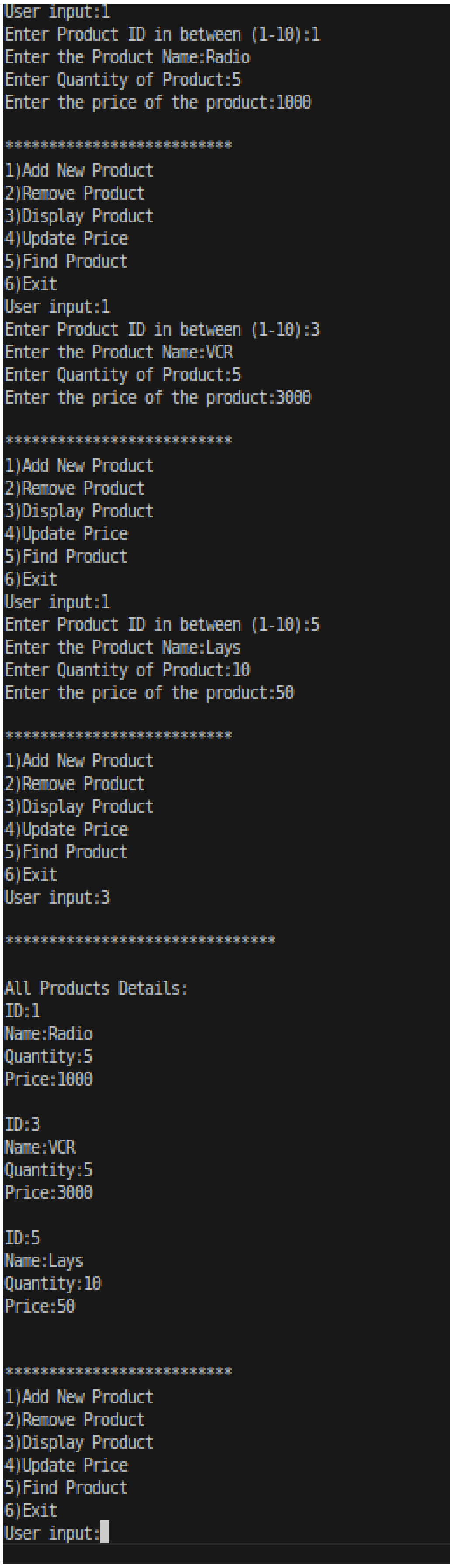
```
{

cout<<"ID:"<<Curr_ptr->ID<<endl;
cout<<"Name:"<<Curr_ptr->Name<<endl;
cout<<"Quantity:"<<Curr_ptr->Quantity<<endl;
cout<<"Price:"<<Curr_ptr->Price<<endl;
if(Curr_ptr->Next!=NULL)
Curr_ptr= Curr_ptr->Next;
cout<<endl;
}

}

void Update_price(int ID, int price)
{
if(length==0)
{
cout<<"\nTheir are no products yet !!!!\nAdd the products first!!!\n";
return;
}
Nodes *curr=head;
for(int i=1;i<=length;i++)
{
if(curr->ID==ID)
{
curr->Price=price;
cout<<"\n*********\nPrice is updated successfully !!!\n*********"<<endl;
return;
}
curr=curr->Next;
}
cout<<"Invalid ID"<<endl;
}

void find_product(int id)
{
if(length==0)
{
cout<<"\nTheir are no products to find !!!!\nFirst add the products!!!\n";
return;
}
Nodes *curr=head;
for(int i=1;i<=length;i++)
{ if(curr->ID==id)
{
cout<<"\nProduct Details:"<<endl;
cout<<"ID:"<<curr->ID<<endl;
cout<<"Product Name:"<<curr->Name<<endl;
cout<<"Quantity:"<<curr->Quantity<<endl;
cout<<"Price:"<<curr->Price<<endl;
return;
}
curr=curr->Next;
}
{
cout<<"Invalid Id such ID does not exist!!!"<<endl;
}
}
};


int main(void)
{

Double_Linked_list DL1;
int user_input;
int ID;
string name;
int quantity;
int price;
//Infinite loop to give end of program control to user
while(1)
{

cout<<"\n**************************"<<endl;
cout<<"1)Add New Product\n2)Remove Product\n3)Display Product\n4)Update Price\n5)Find Product\n6)Exit";
cout<<"\nUser input:";
cin>>user_input;
if(user_input==1)
{
cout<<"Enter Product ID in between (1-10):";
cin>>ID;
cout<<"Enter the Product Name:";
cin.ignore();
getline(cin,name);
cout<<"Enter Quantity of Product:";
cin>>quantity;
cout<<"Enter the price of the product:";
cin>>price;

DL1.insertion(ID,name,quantity,price);
}
else if(user_input==2)
{
cout<<"Enter the ID of the Product you want to remove:";
cin>>ID;
cout<<"\n*******************************\n";

DL1.remove(ID);
}
else if(user_input==3)
{
```

```
cout<<"\n*******************************\n";
DL1.Doubly_display();
}
else if(user_input==4)
{
cout<<"Enter the ID:";
cin>>ID;
cout<<"Enter the price you want to Update:";
cin>>price;
cout<<"\n*******************************\n";

DL1.Update_price(ID,price);
}
else if(user_input==5)
{
cout<<"Enter the ID:";
cin>>ID;
cout<<"\n*******************************\n";

DL1.find_product(ID);
}
else if(user_input==6)
{
cout<<"**********************************************"<<endl;
cout<<"Thanks for using our services. Have a nice day!"<<endl;
cout<<"**********************************************"<<endl;

break;
}
}




}
```

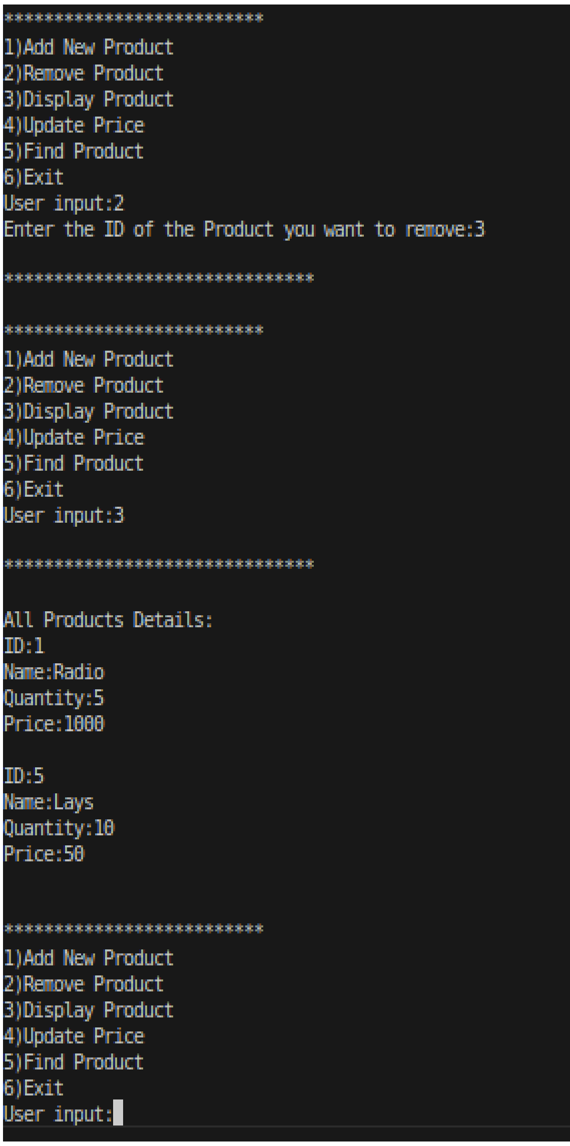
Output Screen Shots:



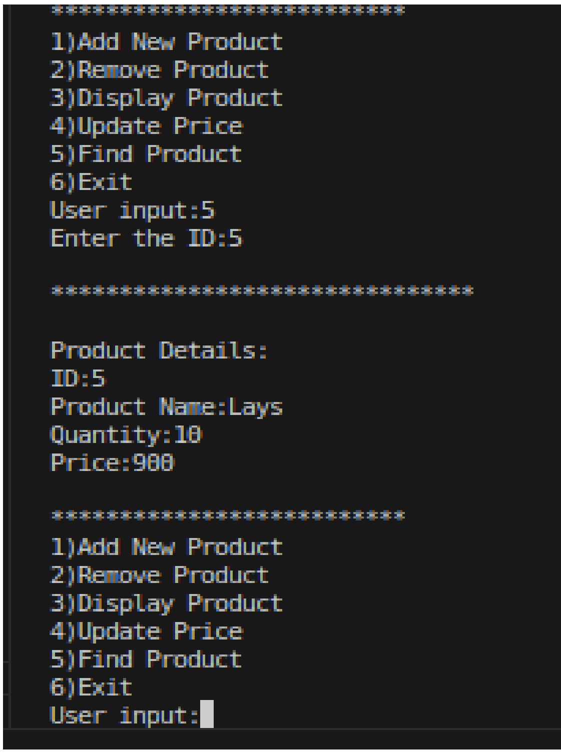
```
**************************
1)Add New Product
2)Remove Product
3)Display Product
4)Update Price
5)Find Product
6)Exit
User input:
```



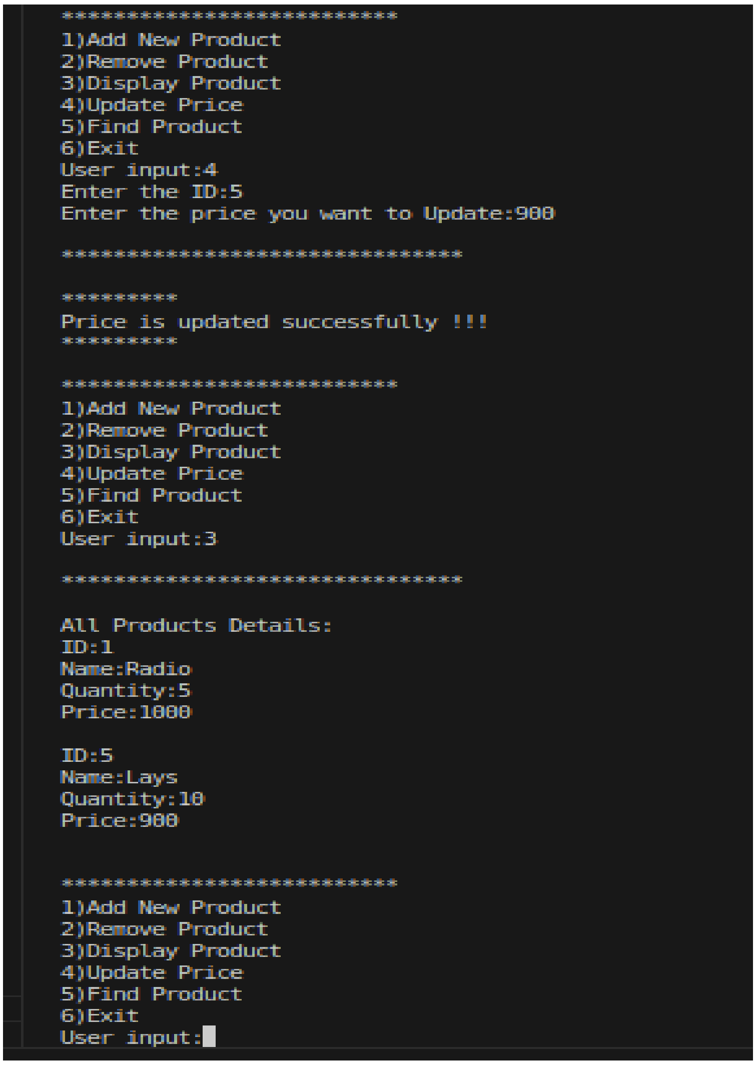
```
User input:1
Enter Product ID in between (1-10):1
Enter the Product Name:Radio
Enter Quantity of Product:5
Enter the price of the product:1000

**************************
1)Add New Product
2)Remove Product
3)Display Product
4)Update Price
5)Find Product
6)Exit
User input:1
Enter Product ID in between (1-10):3
Enter the Product Name:VCR
Enter Quantity of Product:5
Enter the price of the product:3000

**************************
1)Add New Product
2)Remove Product
3)Display Product
4)Update Price
5)Find Product
6)Exit
User input:1
Enter Product ID in between (1-10):5
Enter the Product Name:Lays
Enter Quantity of Product:10
Enter the price of the product:50

**************************
1)Add New Product
2)Remove Product
3)Display Product
4)Update Price
5)Find Product
6)Exit
User input:3

*******************************

All Products Details:
ID:1
Name:Radio
Quantity:5
Price:1000

ID:3
Name:VCR
Quantity:5
Price:3000

ID:5
Name:Lays
Quantity:10
Price:50


**************************
1)Add New Product
2)Remove Product
3)Display Product
4)Update Price
5)Find Product
6)Exit
User input:
```



```
**************************
1)Add New Product
2)Remove Product
3)Display Product
4)Update Price
5)Find Product
6)Exit
User input:2
Enter the ID of the Product you want to remove:3

*******************************

**************************
1)Add New Product
2)Remove Product
3)Display Product
4)Update Price
5)Find Product
6)Exit
User input:3

*******************************

All Products Details:
ID:1
Name:Radio
Quantity:5
Price:1000

ID:5
Name:Lays
Quantity:10
Price:50


**************************
1)Add New Product
2)Remove Product
3)Display Product
4)Update Price
5)Find Product
6)Exit
User input:
```



```
**************************
1)Add New Product
2)Remove Product
3)Display Product
4)Update Price
5)Find Product
6)Exit
User input:5
Enter the ID:5

*******************************

Product Details:
ID:5
Product Name:Lays
Quantity:10
Price:900

**************************
1)Add New Product
2)Remove Product
3)Display Product
4)Update Price
5)Find Product
6)Exit
User input:
```



```
**************************
1)Add New Product
2)Remove Product
3)Display Product
4)Update Price
5)Find Product
6)Exit
User input:4
Enter the ID:5
Enter the price you want to Update:900

*******************************

*********
Price is updated successfully !!!
*********

**************************
1)Add New Product
2)Remove Product
3)Display Product
4)Update Price
5)Find Product
6)Exit
User input:3

*******************************

All Products Details:
ID:1
Name:Radio
Quantity:5
Price:1000

ID:5
Name:Lays
Quantity:10
Price:900


**************************
1)Add New Product
2)Remove Product
3)Display Product
4)Update Price
5)Find Product
6)Exit
User input:
```

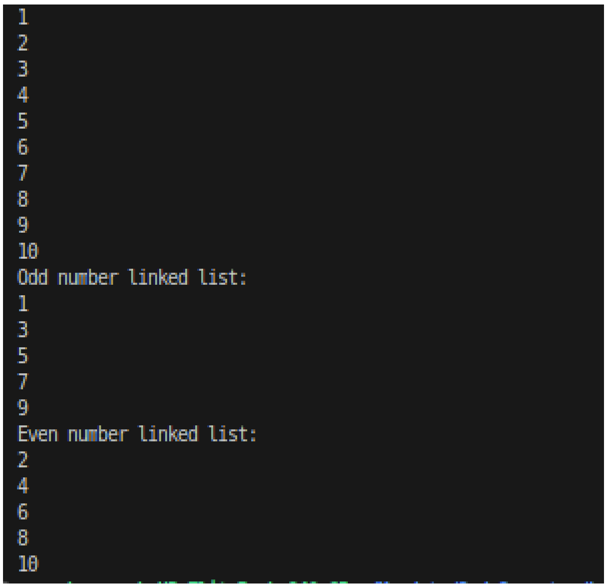
Question 2:

```
#include<iostream>


using namespace std;


class Nodes
{
public:
Nodes *next;
int info;
Nodes *prev;

Nodes(int value)
{
info=value;
next=NULL;
prev=NULL;
}

};

class LinkedList
{
private:
```

```
Nodes *head;
int length;

public:
LinkedList()
{
head=NULL;
length=0;
}

void insertion(int value, int position)
{
//Position check
if(position<0 || position> length+1)
{
cout<<"\nInvalid Position entered!!!!!"<<endl;
return;
}

Nodes *new_node= new Nodes(value);
if(position==1)
{
if(head!=NULL)
{
Nodes *Curr_ptr=head;
new_node->next=Curr_ptr;
Curr_ptr->prev=new_node;
head=new_node;
for(int i=1;i<length;i++)
{
Curr_ptr=Curr_ptr->next;
}
}
else
{
head=new_node;
}
}
else if(position!=length+1)
{
Nodes *Curr_ptr=head;
Nodes *slow2=head;
for (int i=1;i<position;i++)
{
slow2=Curr_ptr;
Curr_ptr=Curr_ptr->next;
}
new_node->next=Curr_ptr;
slow2->next=new_node;
Curr_ptr->prev=new_node;
new_node->prev=slow2;
}
else//For insertion at the end of the circular doubly linkedlist
{ Nodes*Curr_ptr=head;
Nodes *slow2=head;
for (int i=1;i<position-1;i++)
{
Curr_ptr=Curr_ptr->next;
}
Curr_ptr->next=new_node;
new_node->prev=Curr_ptr;
}


length+=1;
}

//To extract the odd values from the original list to the new list
void Odd_creator(LinkedList OL)
{
Nodes *Curr1=head;
Nodes *Curr2=OL.head;

for( int i=1;i<=OL.length;i++)
{
if(Curr2->info % 2 != 0)
{
Nodes *new_node= new Nodes(Curr2->info);
if(head==NULL)
{
head=new_node;
Curr1=new_node;
}
else
{
Curr1->next=new_node;
new_node->prev=Curr1;
Curr1= Curr1->next;
}
length+=1;
}
Curr2=Curr2->next;

}
Curr1->next=head;

}

//To extract the even values from the original list to the new list
void Even_creator(LinkedList OL)
{
Nodes *Curr1=head;
Nodes *Curr2=OL.head;

for( int i=1;i<=OL.length;i++)
{
if(Curr2->info % 2 == 0)
{
Nodes *new_node= new Nodes(Curr2->info);
if(head==NULL)
{
head=new_node;
Curr1=new_node;
}
else
{
Curr1->next=new_node;
new_node->prev=Curr1;
Curr1= Curr1->next;
}
length+=1;
}
Curr2=Curr2->next;

}
Curr1->next=head;

}
void Doubly_display()
{
Nodes *Curr_ptr=head;

for (int i=1; i<=length;i++)
{

cout<<Curr_ptr->info<<endl;

Curr_ptr= Curr_ptr->next;
}

}

};


int main(void)
{
LinkedList OL,L1,L2;

for(int i=1;i<=10;i++)
{
OL.insertion(i,i);
}
```

```
OL.Doubly_display();

L1.Odd_creator(OL);
L2.Even_creator(OL);

cout<<"Odd number linked list:"<<endl;
L1.Doubly_display();

cout<<"Even number linked list:"<<endl;
L2.Doubly_display();


return 0;
}
```

Output Screen Shots:



```
1
2
3
4
5
6
7
8
9
10
Odd number linked list:
1
3
5
7
9
Even number linked list:
2
4
6
8
10
```

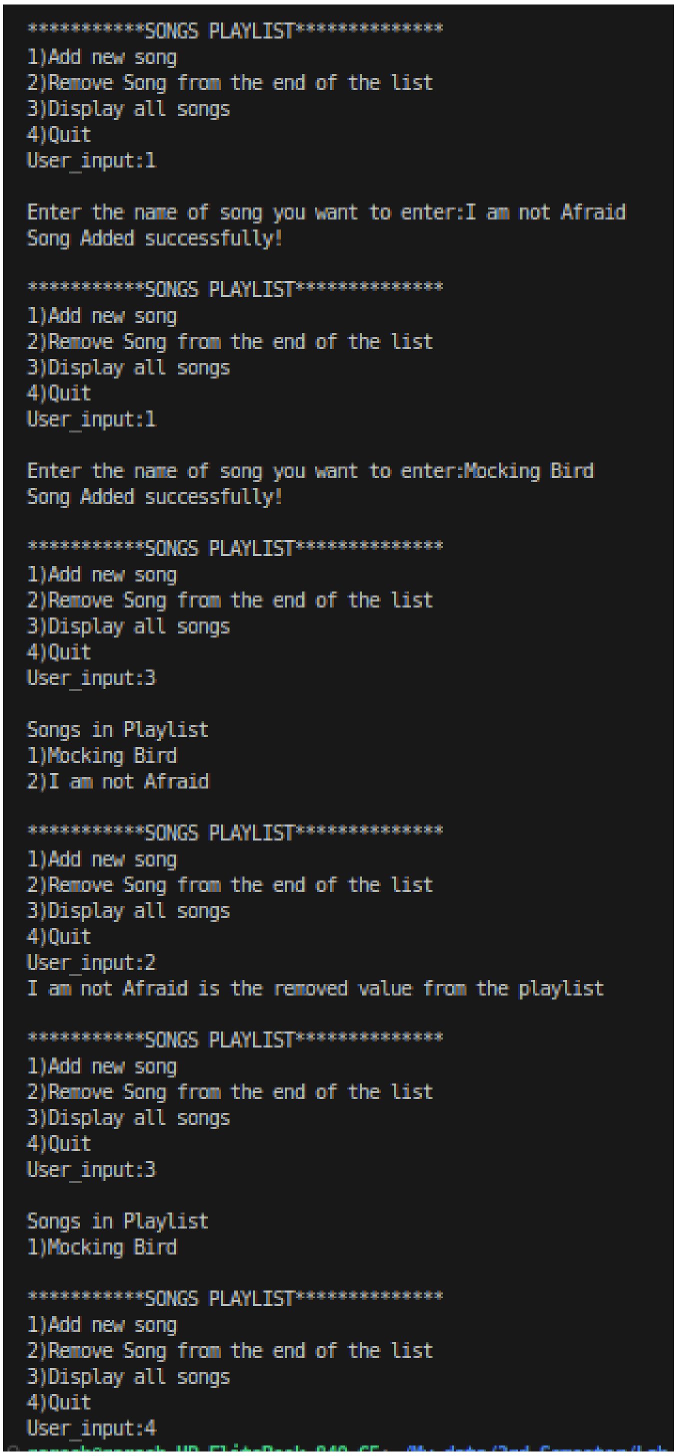
Question 3:

```
#include<iostream>


using namespace std;

class Nodes
{
public:
Nodes *Next;
```

```
string song_name;

Nodes(string name)
{
song_name= name;
Next=NULL;
}

};


class Circular_singly_Linked_list
{
private:
int length;
Nodes *head;

public:
Circular_singly_Linked_list()
{
length=0;
head=NULL;

}

void insertion(string value, int position)
{
//Position check
if(position<0 || position>length+1)
{
cout<<"INvalid position entered !!!"<<endl;
return;
}
Nodes *New_node= new Nodes(value);
if(position==1)
{
if(head!=NULL)
{
New_node->Next=head;
head=New_node;
Nodes *curr_ptr= head;
for (int i=1;i<=length;i++)
{
curr_ptr=curr_ptr->Next;
}
curr_ptr->Next=New_node;
}
else
{
head=New_node;
New_node->Next=head;
}
}
length+=1;


}
void display_linked_list()
{
Nodes *Curr_ptr=head;
if(length==0)
{
cout<<"\nPlaylist is empty!!!Enter the songs name first\n";
return;
}
for(int i=1;i<=length;i++)
{
cout<<i<<")"<<Curr_ptr->song_name<<endl;

Curr_ptr=Curr_ptr->Next;
}
}

//removing song from the end of playlist
string remove()
{
Nodes *curr1=head;
Nodes *slow=head;

if(length==0)
{
cout<<"\nThe playlist is empty! First add some songs"<<endl;
return " ";
}
for(int i=0;i<length-1;i++)
{
slow=curr1;
curr1=curr1->Next;
}
slow->Next=head;
string value=curr1->song_name;
length-=1;
delete curr1;
return value;
}
};


int main(void)
{
Circular_singly_Linked_list CSL1;
int user_input;
```

```
while(1)
{
cout<<"\n***********SONGS PLAYLIST**************\n";
cout<<"1)Add new song\n2)Remove Song from the end of the list\n3)Display all songs\n4)Quit";
cout<<"\nUser_input:";
cin>>user_input;
if(user_input==1)
{
cout<<"\nEnter the name of song you want to enter:";
string song_name;
cin.ignore();
getline(cin,song_name);
CSL1.insertion(song_name,1);
cout<<"Song Added successfully!"<<endl;
}
else if(user_input==2)
{

string check=CSL1.remove();
if(check!=" ")
{
cout<<check<<" is the removed value from the
playlist"<<endl;
}
}
else if(user_input==3)
{
cout<<"\nSongs in Playlist\n";
CSL1.display_linked_list();
}
else if(user_input==4)
{
break;
}
}


}
```

Output Screen Shots:



```
***********SONGS PLAYLIST**************
1)Add new song
2)Remove Song from the end of the list
3)Display all songs
4)Quit
User_input:1

Enter the name of song you want to enter:I am not Afraid
Song Added successfully!

***********SONGS PLAYLIST**************
1)Add new song
2)Remove Song from the end of the list
3)Display all songs
4)Quit
User_input:1

Enter the name of song you want to enter:Mocking Bird
Song Added successfully!

***********SONGS PLAYLIST**************
1)Add new song
2)Remove Song from the end of the list
3)Display all songs
4)Quit
User_input:3

Songs in Playlist
1)Mocking Bird
2)I am not Afraid

***********SONGS PLAYLIST**************
1)Add new song
2)Remove Song from the end of the list
3)Display all songs
4)Quit
User_input:2
I am not Afraid is the removed value from the playlist

***********SONGS PLAYLIST**************
1)Add new song
2)Remove Song from the end of the list
3)Display all songs
4)Quit
User_input:3

Songs in Playlist
1)Mocking Bird

***********SONGS PLAYLIST**************
1)Add new song
2)Remove Song from the end of the list
3)Display all songs
4)Quit
User_input:4
```