ADS lab-1

Program-1(xor linked list)

Code:

```
#include <bits/stdc++.h>
#include <cinttypes>

using namespace std;

class Node {
   public :
   int data;
   Node* xn;
};

Node* Xor(Node* node1, Node* node2)
{
   return reinterpret_cast<Node*>(
      reinterpret_cast<uintptr_t>(node1)
      ^ reinterpret_cast<uintptr_t>(node2));
}

Node* insertfront(Node* head, int data){
   Node* newNode = new Node();
   newNode->data = data;
```

```
    newNode->data = data;
    newNode->xn = head;
    if (head!=nullptr) {
        head->xn=Xor(newNode,head->xn);
    }
    head=newNode;
    return head;
}

Node* deletefront(Node* head){
    Node* next=head->xn;
    next->xn=Xor(head,next->xn);
    free(head);
    return next;
}

Node* insertrear(Node* head,int data){
    Node* newNode = new Node();
    newNode->data = data;
    if(head==nullptr){
        newNode->xn = head;
        return newNode;
    }
    Node* temp = head;
    Node* prev = NULL;
    Node* next;
    while (temp->xn!=prev) {
        next = Xor(prev, temp -> xn);
        prev = temp;
        temp = next;
    }
    temp->xn=Xor(prev,newNode);
    newNode->xn=temp;
    return head;
}

Node* deleterear(Node* head){
    Node* temp = head;
    Node* prev = NULL;
    Node* next;
    while (temp->xn!=prev) {
        next = Xor(prev, temp -> xn);
        prev = temp;
        temp = next;
    }
   prev->xn=Xor(temp,prev->xn);
   free(temp);
   return head;
}

void print(Node* head)
```

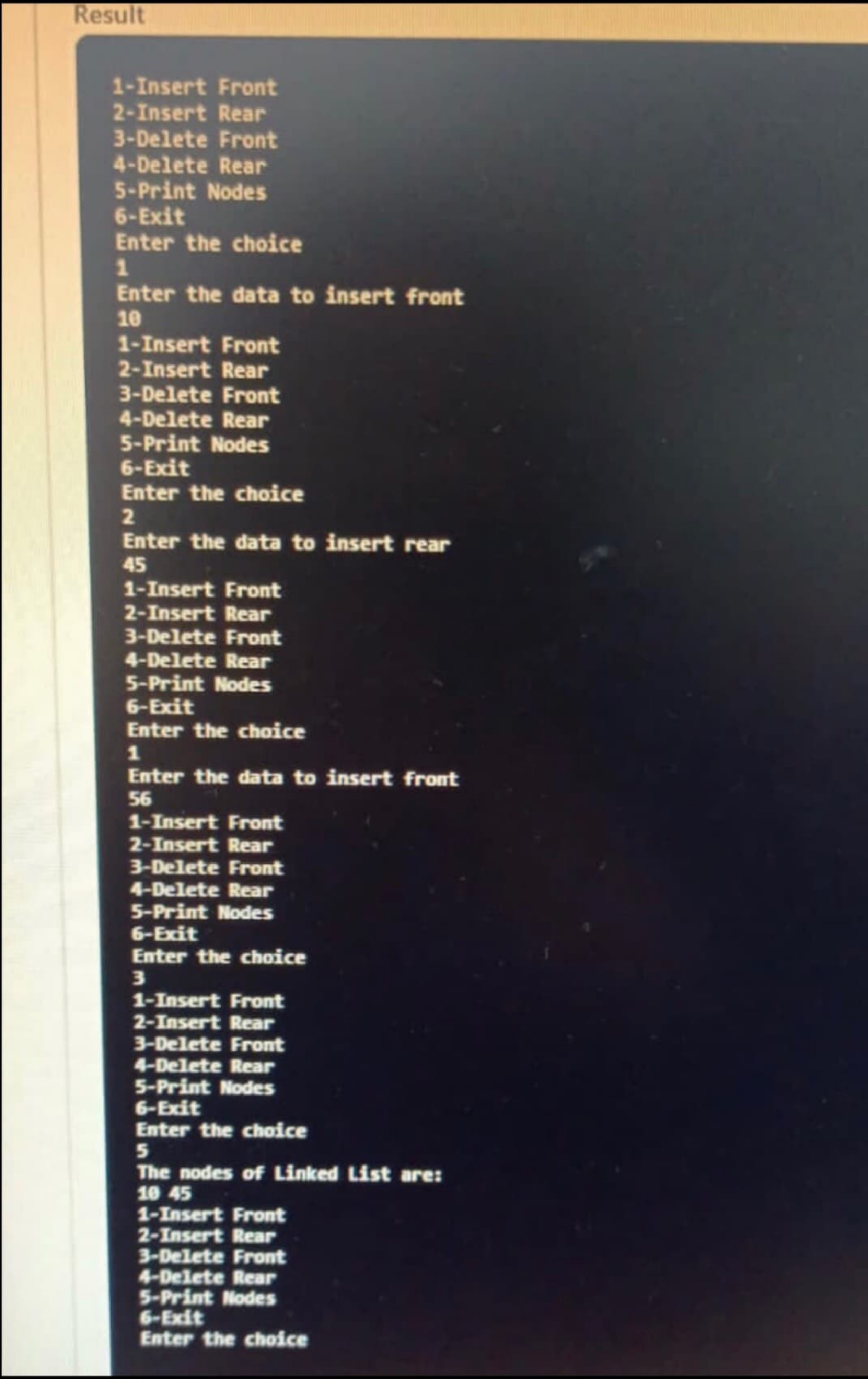
```
{
   Node* temp = head;
   Node* prev = NULL;
   Node* next;

   cout << "The nodes of Linked List are: \n";
   while (temp != nullptr) {
      cout << temp -> data << " ";
      next = Xor(prev, temp -> xn);
      prev = temp;
      temp = next;
   }
   cout<<endl;
}

int main()
{
   Node* head = nullptr;
   int n;
   while(true){
   cout<<"1-Insert Front\n"<<"2-Insert
Rear\n"<<"3-Delete Front\n"<<"4-Delete
Rear\n"<<"5-Print Nodes\n"<<"6-Exit"<<endl;
   int ch;
   cout<<"Enter the choice"<<endl;
   cin>>ch:
    switch(ch){
      case 1:
        int data;
        cout<<"Enter the data to insert
front"<<endl;
        cin>>data;
        head=insertfront(head,data);
        break;
      case 2:
        int data1;
        cout<<"Enter the data to insert
rear"<<endl;
        cin>>data1;
        head=insertrear(head,data1);
        break;
      case 3:
        head=deletefront(head);
        break;
      case 4:
        head=deleterear(head);
        break;
      case 5:
        print(head);
        break;
      case 6:
            head=insertfront(head,data);
            break;
         case 2:
            int data1;
            cout<<"Enter the data to insert
rear"<<endl;
            cin>>data1;
            head=insertrear(head,data1);
            break;
         case 3:
            head=deletefront(head);
            break;
         case 4:
            head=deleterear(head);
            break;
         case 5:
            print(head);
            break;
         case 6:
            exit(0);
      }
   }
   return 0;

}
```

Output:



Result

```
1-Insert Front
2-Insert Rear
3-Delete Front
4-Delete Rear
5-Print Nodes
6-Exit
Enter the choice
1
Enter the data to insert front
10
1-Insert Front
2-Insert Rear
3-Delete Front
4-Delete Rear
5-Print Nodes
6-Exit
Enter the choice
2
Enter the data to insert rear
45
1-Insert Front
2-Insert Rear
3-Delete Front
4-Delete Rear
5-Print Nodes
6-Exit
Enter the choice
1
Enter the data to insert front
56
1-Insert Front
2-Insert Rear
3-Delete Front
4-Delete Rear
5-Print Nodes
6-Exit
Enter the choice
3
1-Insert Front
2-Insert Rear
3-Delete Front
4-Delete Rear
5-Print Nodes
6-Exit
Enter the choice
5
The nodes of Linked List are:
10 45
1-Insert Front
2-Insert Rear
3-Delete Front
4-Delete Rear
5-Print Nodes
6-Exit
Enter the choice
```