Competitive Programming

(https://github.com/MalavikaJayakumar/Competitive-Programming-Problems)

1. **Linked List Set 2**
    a. **Doubly linked list**

Code:

```
#include <iostream>
using namespace std;

class node
{
    public:
    int data;
    node *next;
    node *prev;
};

void insertfront(node** head)
{
    node* n = new node();
       int g;
       cout<<"Enter value to insert at beginning: ";
       cin>>g;
    n->data = g;
    n->next = (*head);
    n->prev = NULL;
    (*head) = n;
}

void inserttail(node** head)
{
    node* n = new node();

    node *t = *head;
       int g;
       cout<<"Enter element to insert at end: ";
       cin>>g;
    n->data = g;
    n->next = NULL;
    if (*head == NULL)
    {
        *head = n;
        return;
    }
    while (t->next != NULL)
```

```
        t = t->next;
    t->next = n;
    n->prev = t;
    return;
}

void deleteback(node** head)
{
       node* t =*head;
       if(*head ==NULL)
              cout<<"List is empty";
       if(t->next==NULL)
       delete(head);
       while(t->next->next!=NULL)
       {
              t=t->next;
       }
       delete(t->next);
       t->next=NULL;
       return;
}

void deletefront(node** head)
{
       node* t=*head;
       node *x;
       x=t->next;
       x->prev=NULL;
       delete(t);

       *head=x;
       return;
}

void display(node *n)
{
       node* t;
       cout<<"List from start to end : ";
    while (n != NULL)
    {
        cout<<" "<<n->data;
        t = n;
        n = n->next;
    }
    cout<<"\n\n\nList from end to start : ";
    while(t!=NULL)
    {
              cout<<" "<<t->data;
              t=t->prev;

    }

}
```

```
int main()
{
    node* head = NULL;
    int ch;
    char c='y';
    cout<<"\t\tMENU\n 1.Insertion at beginning\n 2.Insertion at end\n
3.Deletion from back\n 4.Deletion from start\n";
              while(c=='y')
              {
               cout<<"\nEnter choice(1-4)";
               cin>>ch;
               switch(ch)
               {
                      case 1:insertfront(&head);
                      break;
                      case 2:inserttail(&head);
                      break;
                      case 3:deleteback(&head);
                      break;
                      case 4:deletefront(&head);
                      break;
                      default:cout<<"invalid choice";
               }
               cout<<"Do you want to continue:(y/n) ";
               cin>>c;
        }



    display(head);

    return 0;
}
```

Output:

```
Enter choice(1-4)1
Enter value to insert at beginning: 58
Do you want to continue:(y/n) y

Enter choice(1-4)2
Enter element to insert at end: 98
Do you want to continue:(y/n) y

Enter choice(1-4)2
Enter element to insert at end: 97
Do you want to continue:(y/n) y

Enter choice(1-4)1
Enter value to insert at beginning: 96
Do you want to continue:(y/n) y

Enter choice(1-4)3
Do you want to continue:(y/n) y

Enter choice(1-4)4
Do you want to continue:(y/n) n
List from start to end :  58 65 98


List from end to start :  98 65
```

b. Circular linked list

Code:

```
#include<iostream>

using namespace std;

class cnode
{
    public:
    int data;
    cnode *next;
};

void insert(cnode **head)
{
       cnode* n = new cnode();
       cnode *t =*head;
       cout<<"Enter value to insert : ";
       cin>>n->data;
       n->next = *head;
       if(*head != NULL)
       {
              while(t->next != *head)
              {
                     t=t->next;
              }
              t->next = n;
       }
       else
              n->next = n;
       *head = n;
}

void deletehead(cnode** head)
{
       cnode *s =*head;
       cnode *t = *head;
       if(*head == NULL)
       {
              cout<<"List is empty";
              return;
       }
       if(s->next == s)
       {
              *head = NULL;
              return;
       }
       while(s->next != *head)
              s = s->next;
       s->next = t->next;
       *head = s->next;
       delete(t);
```

```
}

void deleteback(cnode** head)
{
	cnode *n = *head;
	cnode *s;
	if(*head == NULL)
	{
		cout<<"List is empty";
		return;
	}
	if(n->next == n)
	{
		*head = NULL;
		return;
	}
	while(n->next != *head)
	{
		s = n;
		n = n->next;
	}
	s->next = n->next;
	*head = s->next;
	delete(n);
}

void display(cnode *head)
{
	cnode *t = head;
	if(head != NULL)
	{
		do
		{
			cout<<t->data<<" ";
			t = t->next;
		}
		while(t != head);
	}
}

int main()
{
	cnode* head = NULL;
    int ch;
    char c='y';
    cout<<"\t\tMENU\n 1.Insertion\n 2.Deletion from back\n 3.Deletion from
start\n";
		while(c=='y' || c=='Y')
		{
		 cout<<"\nEnter choice(1-3)";
		 cin>>ch;
		 switch(ch)
		 {
			case 1:insert(&head);
```

```
                    break;
                    case 2:deleteback(&head);
                    break;
                    case 3:deletehead(&head);
                    break;
                    default:cout<<"invalid choice";
                }
                cout<<"Do you want to continue:(y/n) ";
                cin>>c;
        }



    display(head);

    return 0;
}
```

Output:

```
                MENU
 1.Insertion
 2.Deletion from back
 3.Deletion from start

Enter choice(1-3)1
Enter value to insert : 58
Do you want to continue:(y/n) y

Enter choice(1-3)1
Enter value to insert : 32
Do you want to continue:(y/n) y

Enter choice(1-3)1
Enter value to insert : 65
Do you want to continue:(y/n) y

Enter choice(1-3)1
Enter value to insert : 325
Do you want to continue:(y/n) y

Enter choice(1-3)1
Enter value to insert : 78
Do you want to continue:(y/n) y

Enter choice(1-3)2
Do you want to continue:(y/n) y

Enter choice(1-3)3
Do you want to continue:(y/n) n
325 65 32
```