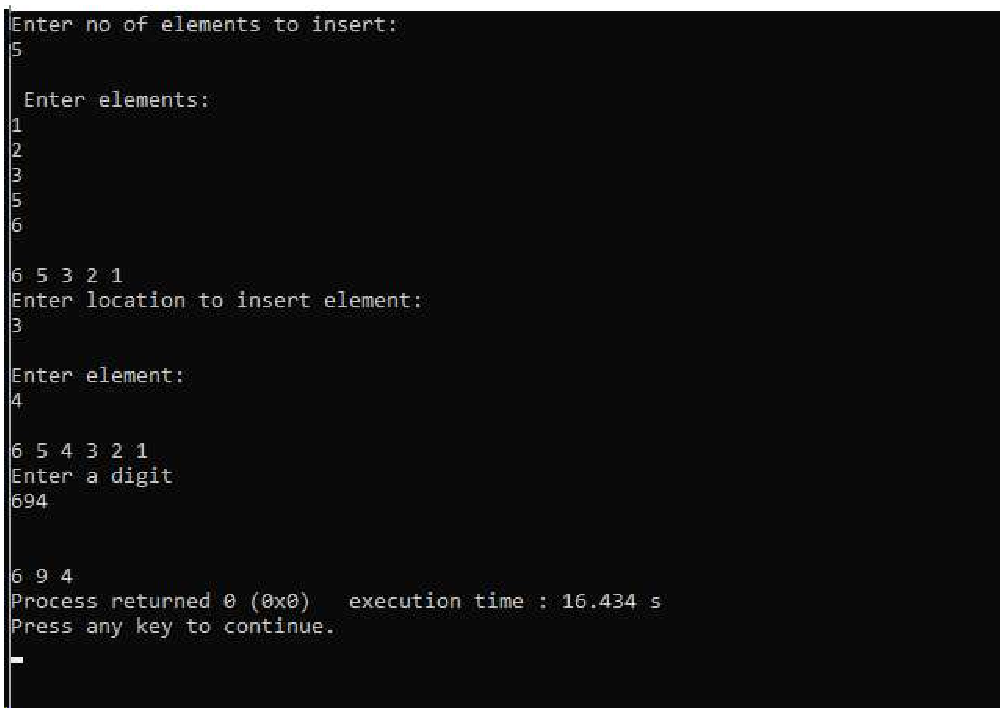
Week 1 Lab B

1)

```
#include<iostream>

using namespace std;

struct node{

int data;

struct node* next;

};

void print(struct node*head){

struct node*ptr=head;

cout<<endl;

while(ptr!=NULL){

cout<<ptr->data<<" ";

ptr=ptr->next;

}

}

struct node* insertatbeginning(struct node*head, int data)

{

   struct node*ptr=new struct node;

   if(head==NULL){


      ptr->data=data;

      ptr->next=NULL;

      head=ptr;
```

```
        return head;
    }
    ptr->data=data;
    ptr->next=head;
    head=ptr;
    return head;
}
struct node* insertatpos(struct node*head, int a, int b){
    struct node*ptr=head;
        struct node*p=new struct node;

if(a==1){
        p->data=b;
p->next=head;
head=p;
return head;
}
while(a!=2){
ptr=ptr->next;
a--;
}
p->data=b;
p->next=ptr->next;
ptr->next=p;
return head;

}
struct node* storeelements(struct node*head,int a){
    while(a!=0){
int k=a%10;
a=a/10;
head=insertatbeginning(head,k);
    }
return head;

}

int main(){
struct node* head= new struct node;
head=NULL;
cout<<"Enter no of elements to insert:\n";
int x,a,b;
cin>>x;
cout<<"\n Enter elements:\n";
for(int i=0;i<x;i++){
    int a;
    cin>>a;
    head=insertatbeginning(head,a);
}
print(head);
cout<<"\nEnter location to insert element:\n";
cin>>a;
```

```
cout<<"\nEnter element:\n";

cin>>b;

head=insertatpos(head,a,b);

print(head);



struct node*digit=new struct node;

digit=NULL;

cout<<"\nEnter a digit\n";

cin>>a;

digit=storeelements(digit,a);

cout<<endl;

print(digit);

return 0;

}
```



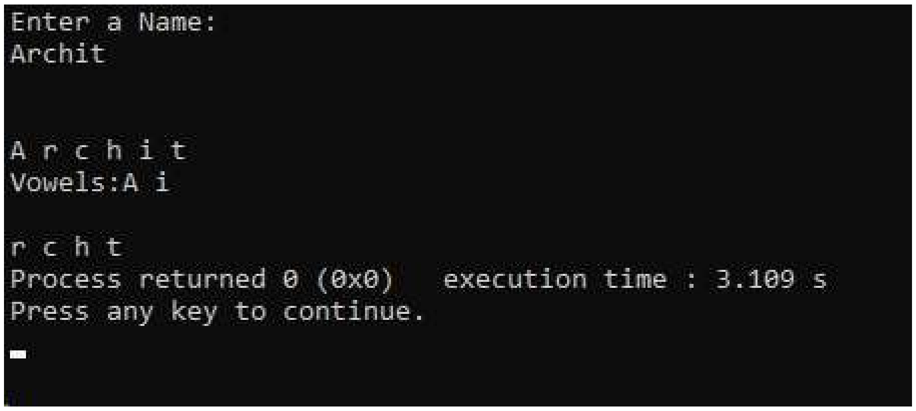
```
Enter no of elements to insert:
5

 Enter elements:
1
2
3
5
6

6 5 3 2 1
Enter location to insert element:
3

Enter element:
4

6 5 4 3 2 1
Enter a digit
694


6 9 4
Process returned 0 (0x0)   execution time : 16.434 s
Press any key to continue.
```

2)

```
#include<iostream>
#include<cstring>
using namespace std;
struct node{
char data;
struct node* next;
};
void print(struct node*head){
struct node*ptr=head;
cout<<endl;
while(ptr!=NULL){
cout<<ptr->data<<" ";
ptr=ptr->next;
}
}
struct node* insertatbeginning(struct node*head, char data)
{
   struct node*ptr=new struct node;
   if(head==NULL){

       ptr->data=data;
       ptr->next=NULL;
       head=ptr;
```

```
        return head;

    }

    ptr->data=data;

    ptr->next=head;

    head=ptr;

    return head;

}

struct node* deletenode(struct node*head, struct node* node){


    if(head==node){

              struct node*k=head;

              head=head->next;

              delete k;

        return head;

    }

     struct node*p=head;

    struct node*q=head->next;

    while(q!=node)

    {

        p=p->next;

        q=q->next;

    }

    p->next=q->next;

    delete q;

    return head;

}
```

```
struct node* deletevowels(struct node*head){

struct node*p=head;

cout<<endl<<"Vowels:";

while(p!=NULL)

{


if(p->data=='a'||p->data=='e'||p->data=='i'||p->data=='o'||p->data=='u'||p->data=='A'||p->data=='E'||p->data=='I'||p->data=='O'||p->data=='U')

   {

       cout<<p->data<<" ";

       struct node*k=p->next;

      head= deletenode(head,p);

      p=k;


   }

   else{

   p=p->next;

   }

}

cout<<endl;

return head;

}

int main()

{

   string n;

   struct node* head=NULL;

   cout<<"Enter a Name:\n";
   cin>>n;

   int a=n.size();

   a--;

   while(a>=0){

       head=insertatbeginning(head,n[a]);

       a--;

   }

   cout<<endl;

   print(head);

   head=deletevowels(head);

   print(head);

   return 0;

}
```



```
Enter a Name:
Archit


A r c h i t
Vowels:A i

r c h t
Process returned 0 (0x0)   execution time : 3.109 s
Press any key to continue.
```

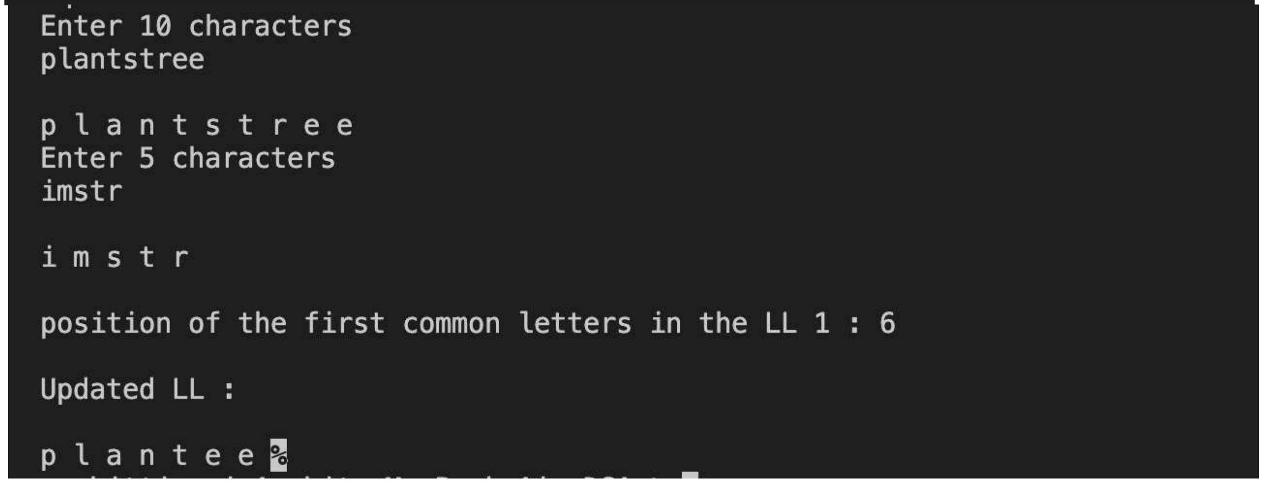
3)

```
/*Create a link list of users supplied ten characters to store a name. Create a
```

```
second link list of same type of user supplied five characters. Now using a
function remove(), traverse first link list and if any three consecutive characters
of second link list appears as consecutive characters of first link list, remove
those from first link list.*/
#include<iostream>
#include<cstring>
using namespace std;
struct node{
char data;
struct node* next;
};
void print(struct node*head){
struct node*ptr=head;
cout<<endl;
while(ptr!=NULL){
cout<<ptr->data<<" ";
ptr=ptr->next;
}
}
struct node* insertatend(struct node*head, char data)
{
   struct node*p=new struct node;
   struct node*ptr=head;
    p->data=data;
   if(ptr==NULL)
   {
       p->next=NULL;

       head=p;

       return head;

   }

  while(ptr->next!=NULL){

   ptr=ptr->next;

  }

  ptr->next=p;

  p->next=NULL;

   return head;

}


bool checksublist(struct node*h1, struct node*h2,int *index)

{




   struct node*p=h1;

   struct node*q=h2;

   for(int b=0;b<3;b++){

       int count=0;

       struct node*r=q;

   for(int c=0;c<8;c++){

       struct node*s=p;

       pin:

       if(s->data==q->data){

       count++;
```

```
        if(count==3)

   {

       *index=c+1;


       return true;

   }

       q=q->next;

       s=s->next;


       goto pin;

       }


       else{


           count=0;

        p=p->next;

        q=r;

        continue;

       }



   }

  p=h1;

q=r->next;

   }

return false;

}
```

```
   struct node* deleteatindex(struct node*head, int index){


   struct node*p=head;

   struct node*q=head->next;

   while((index-2)!=0){

       p=p->next;

       q=q->next;

       index--;

   }

   p->next=q->next;

   delete q;

   return head;

}

struct node* deletesublist(struct node*h1, int a){

   struct node*p=h1;

   if (a==1)

   {

       struct node*p=h1;

       struct node*q=h1->next;

       struct node*r=q->next;;

       struct node*s=r->next;

       h1=s;

       return h1;

       delete p,q,r;

   }

      h1=deleteatindex(h1,a);
      h1=deleteatindex(h1,a+1);

      h1=deleteatindex(h1,a+2);



   return h1;

}

int main(){

struct node*h1=NULL;

struct node*h2= NULL;

char a;

cout<<"Enter 10 characters\n";

for(int i=0;i<10;i++){

   cin>>a;

   h1=insertatend(h1,a);

}

print(h1);

cout<<"\nEnter 5 characters\n";

for(int i=0;i<5;i++){

   cin>>a;

  h2= insertatend(h2,a);

}



print(h2);

cout<<endl;

int index ;



if(checksublist(h1,h2, &index )){
    cout<<endl<<"position of the first common letters in the LL 1 : "<<index<<endl;

}


else{

   cout<<"No 3 consecutive characters of 2nd LL appears in the 1st LL\n ";

   return 0;

}

h1=deletesublist(h1, index);

cout<<endl<<"Updated LL :\n";

print(h1);

}
```



```
Enter 10 characters
plantstree

p l a n t s t r e e
Enter 5 characters
imstr

i m s t r

position of the first common letters in the LL 1 : 6

Updated LL :

p l a n t e e %
```

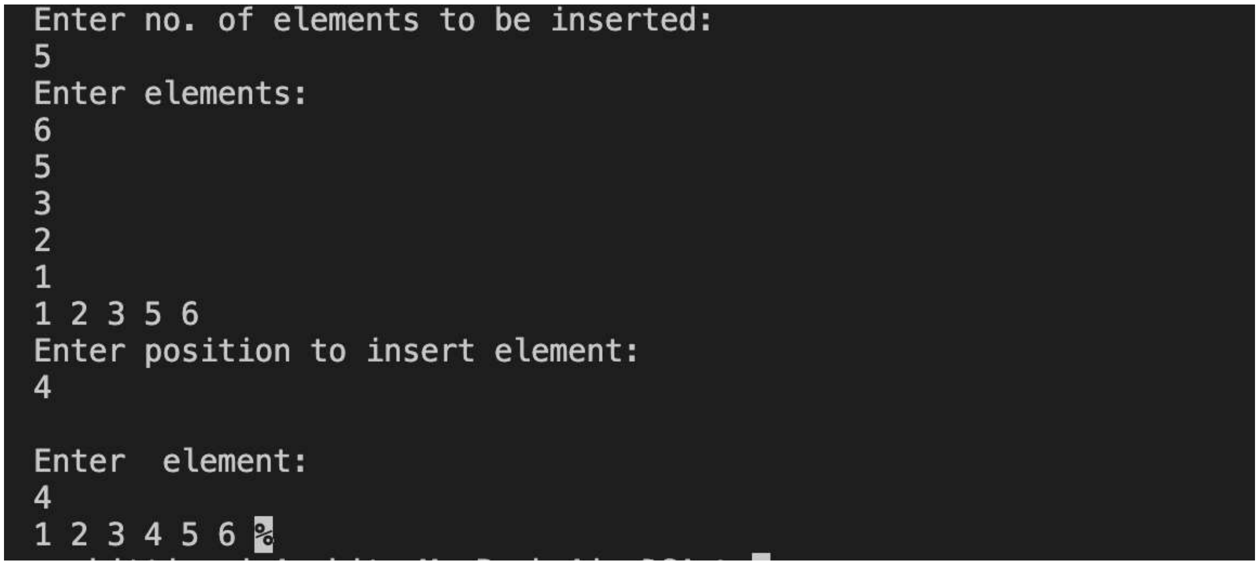
4)

```
// 4. Write a program to insert an element at specific location in doubly linked list.

#include<iostream>

using namespace std;

   struct node{
```

```
int data;

struct node*next;

struct node*prev;

};

struct node *insertathead(struct node*head,int data)

{

    if(head==NULL){

        struct node*p= new struct node;

        p->data=data;

        p->next=NULL;

        p->prev=NULL;

        return p;

    }

    struct node*p=new struct node;

    p->next=head;

    p->data=data;

    p->prev=NULL;

    head=p;

    return head;

}

struct node* insertatposition(struct node*head, int data, int pos){

 if( pos==1){

  head=insertathead(head,data);

  return head;

 }

 struct node*p=head;
```

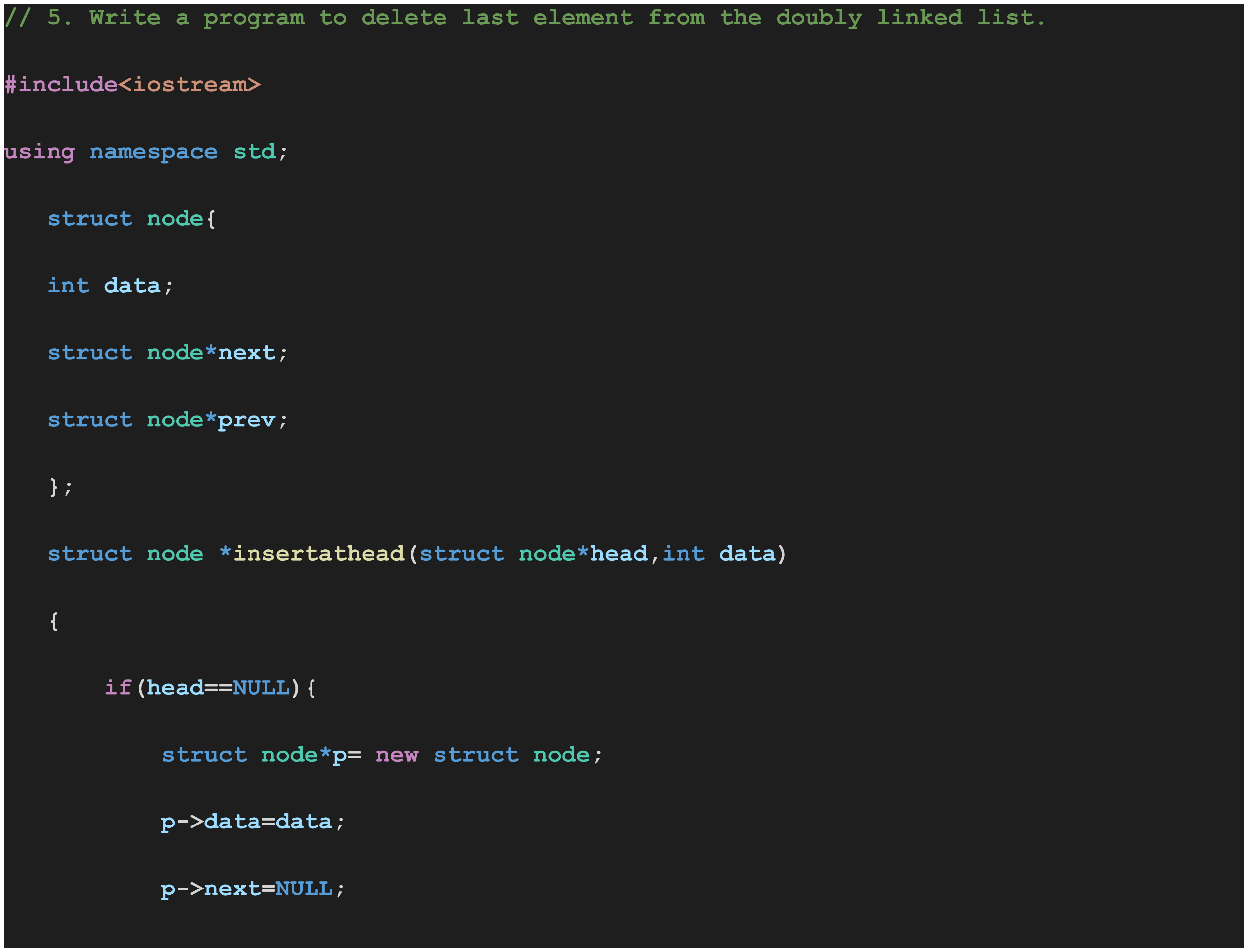
```
        struct node*q=head->next;

        struct node*ptr=new struct node;

        while((pos-2)!=0){

          p=p->next;

          q=q->next;

          pos--;

        }

        ptr->data=data;

        p->next=ptr;

        ptr->next=q;

        ptr->prev=p;

        q->prev=ptr;

        return head;

}

void print(struct node* head){

while(head!=NULL){

    cout<<head->data<<" ";

    head=head->next;

}

}

int main()

{

 struct node*head;

head=NULL;

cout<<"Enter no. of elements to be inserted:\n";

int a,k;
```

```
cin>>a;

cout<<"Enter elements:\n";

while(a!=0){



    cin>>k;

    head=insertathead(head,k);

    a--;

}

print(head);

cout<<"\nEnter position to insert element:\n";

cin>>a;

 cout<<"\nEnter  element:\n";

cin>>k;

head=insertatposition(head,k,a);

print(head);

 return 0;

}
```



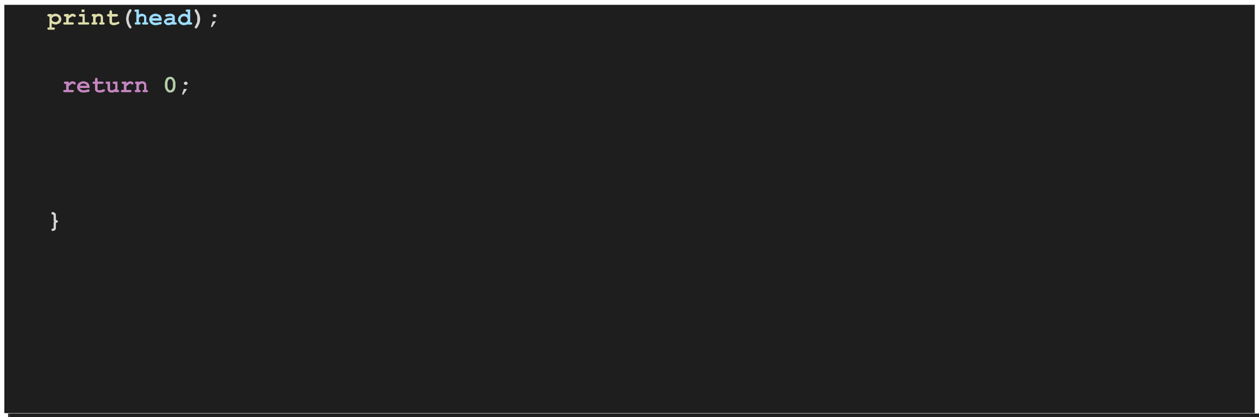
```
Enter no. of elements to be inserted:
5
Enter elements:
6
5
3
2
1
1 2 3 5 6
Enter position to insert element:
4

Enter  element:
4
1 2 3 4 5 6 %
```

5)

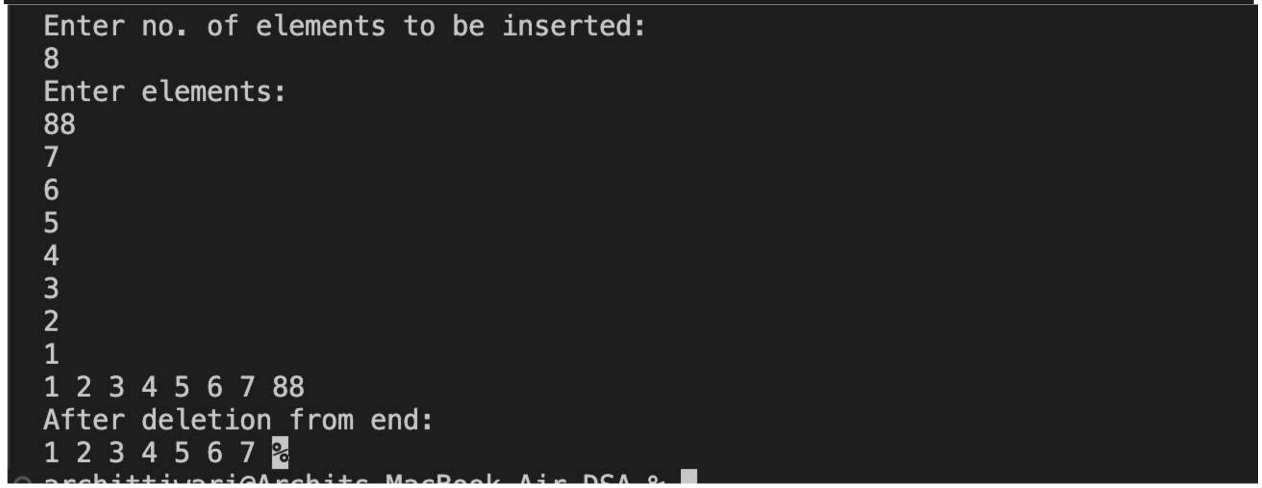


```
// 5. Write a program to delete last element from the doubly linked list.

#include<iostream>

using namespace std;

   struct node{

   int data;

   struct node*next;

   struct node*prev;

   };

   struct node *insertathead(struct node*head,int data)

   {

       if(head==NULL){

           struct node*p= new struct node;

           p->data=data;

           p->next=NULL;
```

```
        p->prev=NULL;

        return p;

    }

    struct node*p=new struct node;

    p->next=head;

    p->data=data;

    p->prev=NULL;

    head=p;

    return head;

}


void print(struct node* head){

while(head!=NULL){

    cout<<head->data<<" ";

    head=head->next;

}

}

struct node* deleteatend(struct node*head)

{

    struct node*p=head;

    struct node*q=head->next;

    if(q==NULL)

    {

        cout<<"After deletion linked list is now empty\n";

        delete p;

        return head;
    }
    while(q->next!=NULL){
        p=p->next;
        q=q->next;
    }
    p->next=NULL;
    delete q;
    return head;
}
int main()
{
 struct node*head;
head=NULL;
cout<<"Enter no. of elements to be inserted:\n";
int a,k;
cin>>a;
cout<<"Enter elements:\n";
while(a!=0){

    cin>>k;
    head=insertathead(head,k);
    a--;
}
print(head);
head=deleteatend(head);
cout<<"\nAfter deletion from end:\n";
```

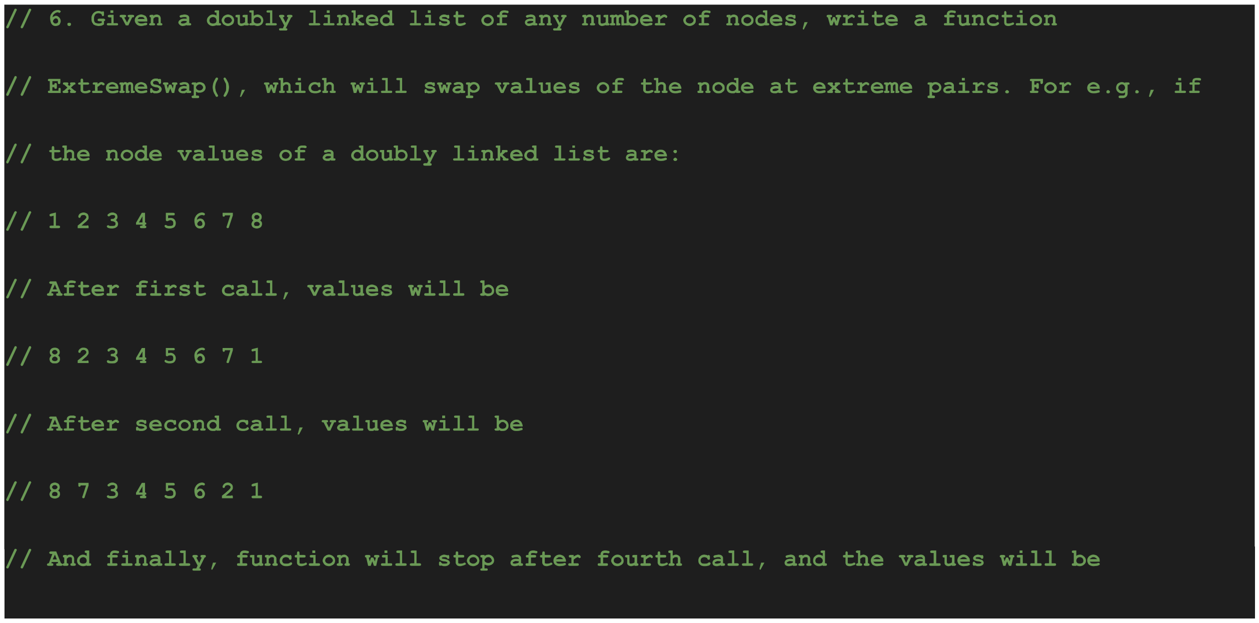


```
  print(head);

   return 0;



  }
```



```
Enter no. of elements to be inserted:
8
Enter elements:
88
7
6
5
4
3
2
1
1 2 3 4 5 6 7 88
After deletion from end:
1 2 3 4 5 6 7 %
architttiwari@Archits-MacBook-Air DSA %
```

6)

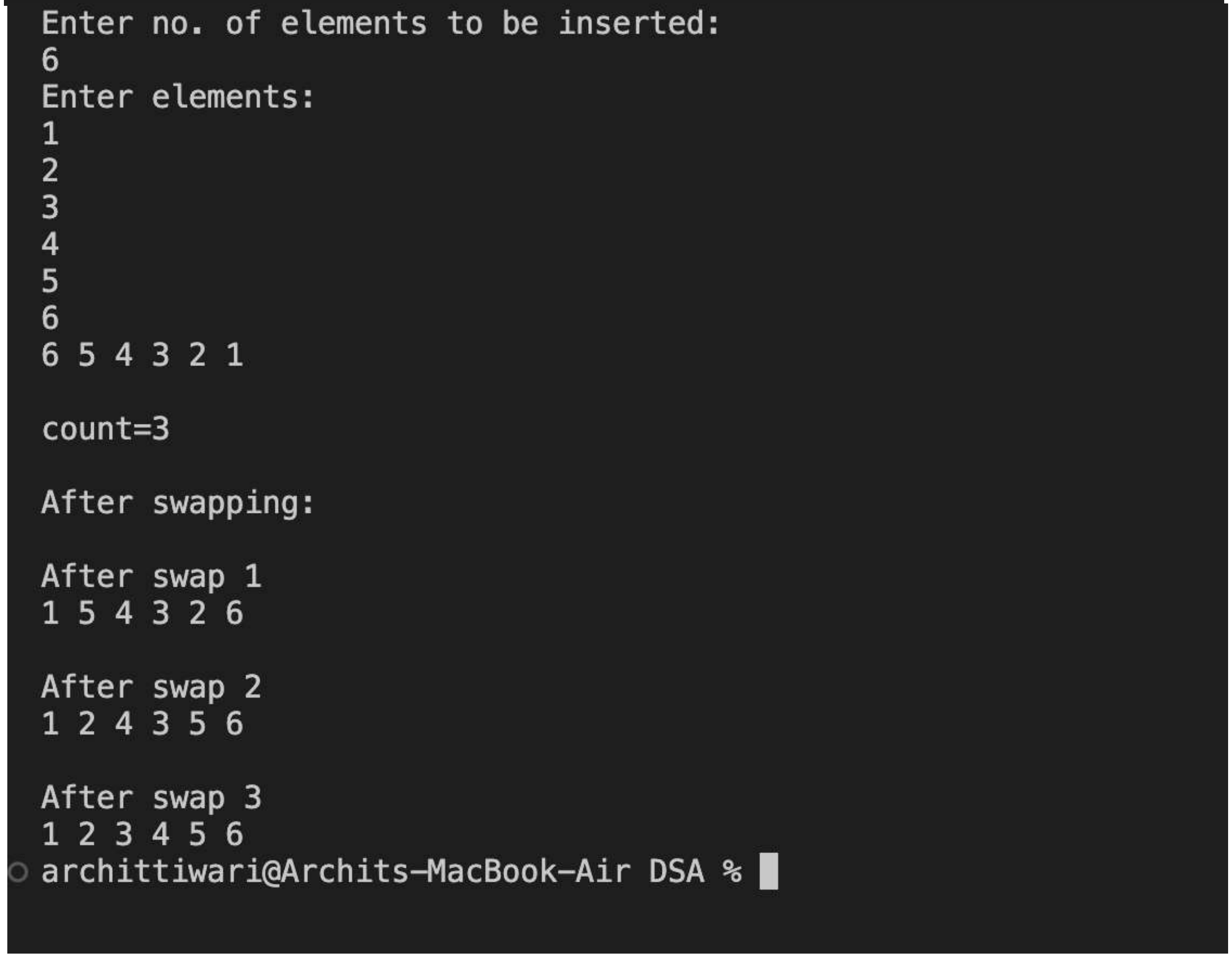


```
// 6. Given a doubly linked list of any number of nodes, write a function

// ExtremeSwap(), which will swap values of the node at extreme pairs. For e.g., if

// the node values of a doubly linked list are:

// 1 2 3 4 5 6 7 8

// After first call, values will be

// 8 2 3 4 5 6 7 1

// After second call, values will be

// 8 7 3 4 5 6 2 1

// And finally, function will stop after fourth call, and the values will be
// 8 7 6 5 4 3 2 1


#include<iostream>

using namespace std;

   struct node{

   int data;

   struct node*next;

   struct node*prev;

   };

   struct node *insertathead(struct node*head,int data)

   {

       if(head==NULL){

           struct node*p= new struct node;

           p->data=data;

           p->next=NULL;

           p->prev=NULL;

           return p;

       }

       struct node*p=new struct node;

       p->next=head;

       p->data=data;

       p->prev=NULL;

       head->prev=p;

       head=p;

       return head;

   }
```

```
void print(struct node* head){
while(head!=NULL){
    cout<<head->data<<" ";
    head=head->next;
}
cout<<endl;
}
struct node*ExtremeSwap(struct node*head, int count){
    struct node*p=head;
    struct node*q=head;
    while(q->next!=NULL){
        q=q->next;
}
for(int i=0;i< (count);i++){
 int temp;
temp=p->data;
p->data=q->data;
q->data=temp;
 p=p->next;
    q=q->prev;
cout<<endl<<"After swap "<<i+1<<endl;

print(head);


}
```

```
return head;

}

int main()

{

 struct node*head=NULL;


cout<<"Enter no. of elements to be inserted:\n";

int a,k;

cin>>a;

int count=a/2;

cout<<"Enter elements:\n";

while(a!=0){


    cin>>k;

    head=insertathead(head,k);

    a--;

}


print(head);




cout<<endl<<"count="<<count<<endl;

cout<<"\nAfter swapping:\n";

head=ExtremeSwap(head,count);

 return 0;
```

```
  }
```



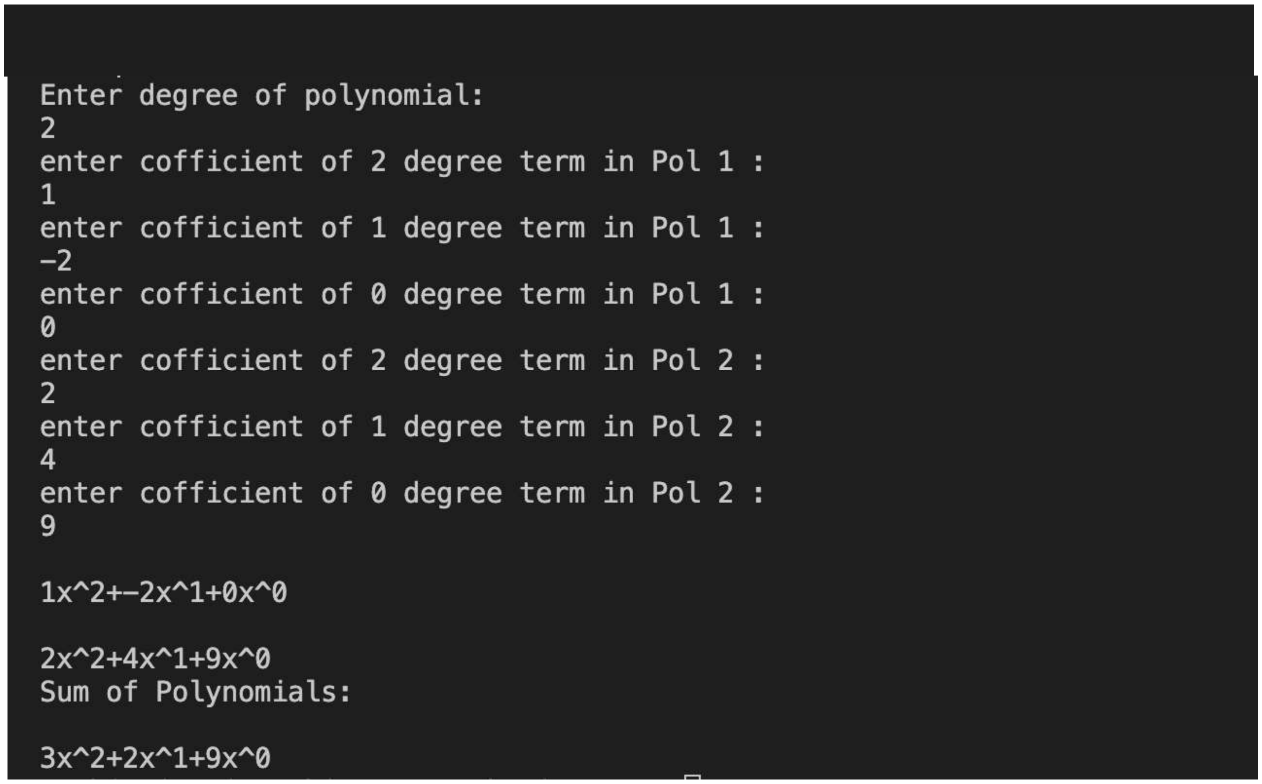
```
Enter no. of elements to be inserted:
6
Enter elements:
1
2
3
4
5
6
6 5 4 3 2 1

count=3

After swapping:

After swap 1
1 5 4 3 2 6

After swap 2
1 2 4 3 5 6

After swap 3
1 2 3 4 5 6
archittiwari@Archits-MacBook-Air DSA %
```

7)

```
#include<iostream>

#include<cstring>

using namespace std;
```

```cpp
struct node{

int data;

int degree;

struct node* next;

};

void print(struct node*head){

struct node*ptr=head;

cout<<endl;

while(ptr->next!=NULL){

// cout<<"Degree: "<<ptr->degree<<" Coefficient: "<<ptr->data<<endl;

// ptr=ptr->next;

cout<<ptr->data<<"x^"<<ptr->degree<<"+";

ptr=ptr->next;

}

cout<<ptr->data<<"x^"<<ptr->degree;

cout<<endl;

}

struct node* insertatend(struct node*head, int data, int degree)

{

   struct node*p=new struct node;

   struct node*ptr=head;

    p->data=data;

    p->degree=degree;

   if(ptr==NULL)

   {

       p->next=NULL;
       head=p;

       return head;

   }

  while(ptr->next!=NULL){

   ptr=ptr->next;

  }

  ptr->next=p;

  p->next=NULL;

   return head;

}

struct node* addpol(struct node*h1, struct node*h2)

{

   int degree,data;

   struct node*sum=new struct node;

   sum=NULL;

   struct node*p=h1;

   struct node*q=h2;

   while(p!=NULL)

   {

    degree=p->degree;

    data=p->data + q->data;

    sum=insertatend(sum,data,degree)    ;

    p=p->next;

    q=q->next;

   }

   return sum;
```

```
}



int main(){

struct node*h1=NULL;

struct node*h2= NULL;

int a,k,s;

cout<<"Enter degree of polynomial:\n";

cin>>s;

for(int i=s;i>=0;i--){

   cout<<"enter cofficient of "<<i<<" degree term in Pol 1 :\n";

   cin>>a;

   h1=insertatend(h1,a,i);

}

for(int i=s;i>=0;i--){

   cout<<"enter cofficient of "<<i<<" degree term in Pol 2 :\n";

   cin>>a;

   h2=insertatend(h2,a,i);

}

print(h1);

print (h2);

cout<<"Sum of Polynomials:\n";

struct node*sum=addpol(h1,h2);

print(sum);

}
```



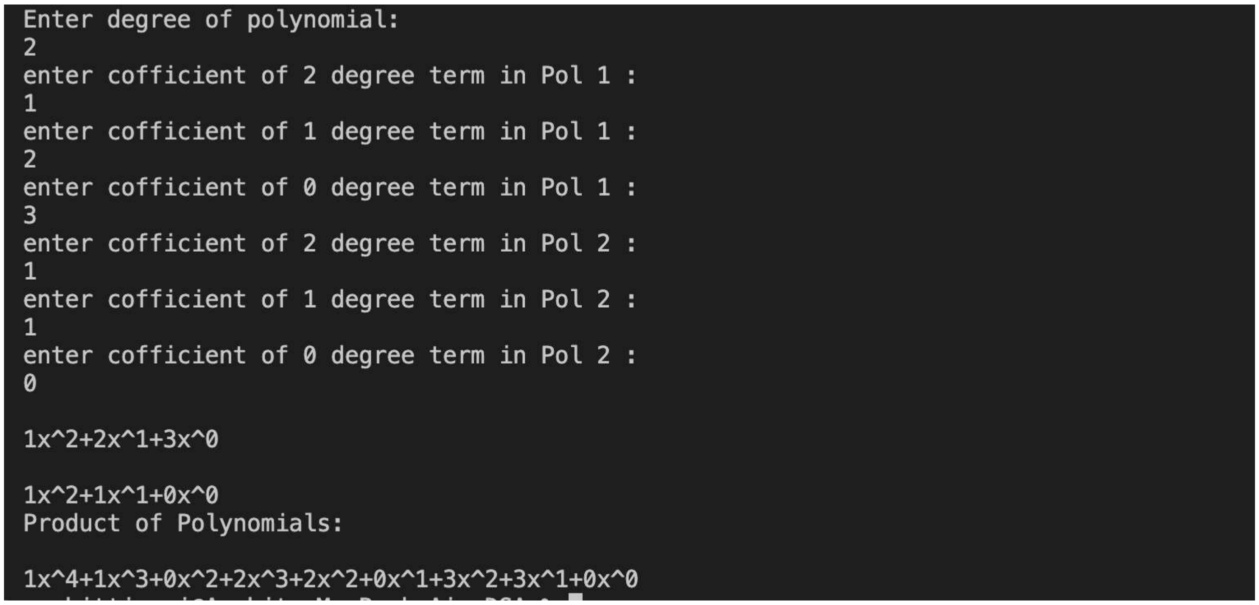
```
Enter degree of polynomial:
2
enter cofficient of 2 degree term in Pol 1 :
1
enter cofficient of 1 degree term in Pol 1 :
-2
enter cofficient of 0 degree term in Pol 1 :
0
enter cofficient of 2 degree term in Pol 2 :
2
enter cofficient of 1 degree term in Pol 2 :
4
enter cofficient of 0 degree term in Pol 2 :
9

1x^2+-2x^1+0x^0

2x^2+4x^1+9x^0
Sum of Polynomials:

3x^2+2x^1+9x^0
```

8)

```
// Write a program to implement multiplication of two polynomials. Each node must

// contain the value of the coefficient as well as its power as data components. Take

// care of law of exponent multiplication.



#include<iostream>

#include<cstring>

using namespace std;

struct node{

int data;

int degree;

struct node* next;

};
```

```cpp
void print(struct node*head){

struct node*ptr=head;

cout<<endl;

while(ptr->next!=NULL){

// cout<<"Degree: "<<ptr->degree<<" Coefficient: "<<ptr->data<<endl;

// ptr=ptr->next;

cout<<ptr->data<<"x^"<<ptr->degree<<"+";

ptr=ptr->next;

}

cout<<ptr->data<<"x^"<<ptr->degree;

cout<<endl;

}

struct node* insertatend(struct node*head, int data, int degree)

{

   struct node*p=new struct node;

   struct node*ptr=head;

    p->data=data;

    p->degree=degree;

   if(ptr==NULL)

   {

       p->next=NULL;

       head=p;

       return head;

   }

  while(ptr->next!=NULL){

   ptr=ptr->next;
  }

  ptr->next=p;

  p->next=NULL;

   return head;

}

struct node* prodpol(struct node*h1, struct node*h2)

{

   int degree,data;

   struct node*product=new struct node;

   product=NULL;

   struct node*p=h1;

   struct node*q=h2;

   while(p!=NULL)

   {

    while(q!=NULL)

    {

    degree=p->degree+q->degree;

    data=(p->data)*(q->data);

    product=insertatend(product,data,degree)    ;

    q=q->next;

    }

    q=h2;

    p=p->next;

   }

   return product;
```

```
}



int main(){

struct node*h1=NULL;

struct node*h2= NULL;

int a,k,s;

cout<<"Enter degree of polynomial:\n";

cin>>s;

for(int i=s;i>=0;i--){

   cout<<"enter cofficient of "<<i<<" degree term in Pol 1 :\n";

   cin>>a;

   h1=insertatend(h1,a,i);

}

for(int i=s;i>=0;i--){

   cout<<"enter cofficient of "<<i<<" degree term in Pol 2 :\n";

   cin>>a;

   h2=insertatend(h2,a,i);

}

print(h1);

print (h2);

cout<<"Product of Polynomials:\n";

struct node*product=prodpol(h1,h2);

print(product);

return 0;

}
```



```
Enter degree of polynomial:
2
enter cofficient of 2 degree term in Pol 1 :
1
enter cofficient of 1 degree term in Pol 1 :
2
enter cofficient of 0 degree term in Pol 1 :
3
enter cofficient of 2 degree term in Pol 2 :
1
enter cofficient of 1 degree term in Pol 2 :
1
enter cofficient of 0 degree term in Pol 2 :
0

1x^2+2x^1+3x^0

1x^2+1x^1+0x^0
Product of Polynomials:

1x^4+1x^3+0x^2+2x^3+2x^2+0x^1+3x^2+3x^1+0x^0
```