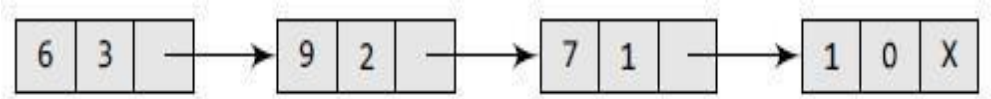
Polynomial Manipulation

Linked lists can be used to represent polynomials and the different operations that can be performed on them

Polynomial Representation

Consider a polynomial 6x3 + 9x2 + 7x + 1. Every individual term in a polynomial consists of two parts, a coefficient and a power. Here, 6, 9, 7, and 1 are the coefficients of the terms that have 3, 2, 1, and 0 as their powers respectively. Every term of a polynomial can be represented as a node of the linked list

Linked representation of a polynomial



Polynomial manipulations such as addition, subtraction & differentiation etc.. can be performed using linked list

Declaration for Linked list implementation of Polynomial ADT

```
struct poly
{
int coeff;
int power;
struct poly * Next;
}
*list1,*list2,*list3;
```

Creation of the Polynomial

```
poly create(poly *head, poly *newnode)
{
```

```
poly*ptr;
if(head==NULL)
{
head=newnode;
return(head);
}
else
{
ptr=head;
while(ptr-> next!=NULL)
ptr=ptr->next;
ptr->next=newnode;
}
return(head);
}
```

Addition of Polynomials:

To add two polynomials, we need to scan them once. If we find terms with the same exponent in the two polynomials, then we add the coefficients; otherwise, we copy the term of larger exponent into the sum and go on. When we reach at the end of one of the polynomial, then remaining part of the other is copied into the sum.

To add two polynomials, follow the following steps:

- Read two polynomials.
- Add them.

- **Display the resultant polynomial.**

Addition of Polynomials:

```
void add()
{
poly *ptr1, *ptr2, *newnode;
ptr1=list1;
ptr2=list2;
while(ptr1!=NULL && ptr2!= NULL)
{
newnode=malloc(sizeof(struct poly));
if(ptr1->power==ptr2->power)
{
newnode->coeff = ptr1->coeff + ptr2->coeff;
newnode->power=ptr1->power;
newnode->next=NULL;
list3=create(list3,newnode);
ptr1=ptr->next;
ptr2=ptr2->next;
}
else
{
if(ptr1->power > ptr2->power)
{
newnode->coeff = ptr1->coeff
newnode->power=ptr1->power;
newnode->next=NULL;
list3=create(list3,newnode);
ptr1=ptr1->next;
}
else
{
newnode->coeff = ptr2->coeff
```

```
newnode->power=ptr2->power;
newnode->next=NULL;
list3=create(list3,newnode);
ptr2=ptr2->next;
}
}
}
```

FOR SUBTRACTION OF POLYNOMIALS ,

add this statement in the above program newnode->coeff = ptr1->coeff - ptr2->coeff