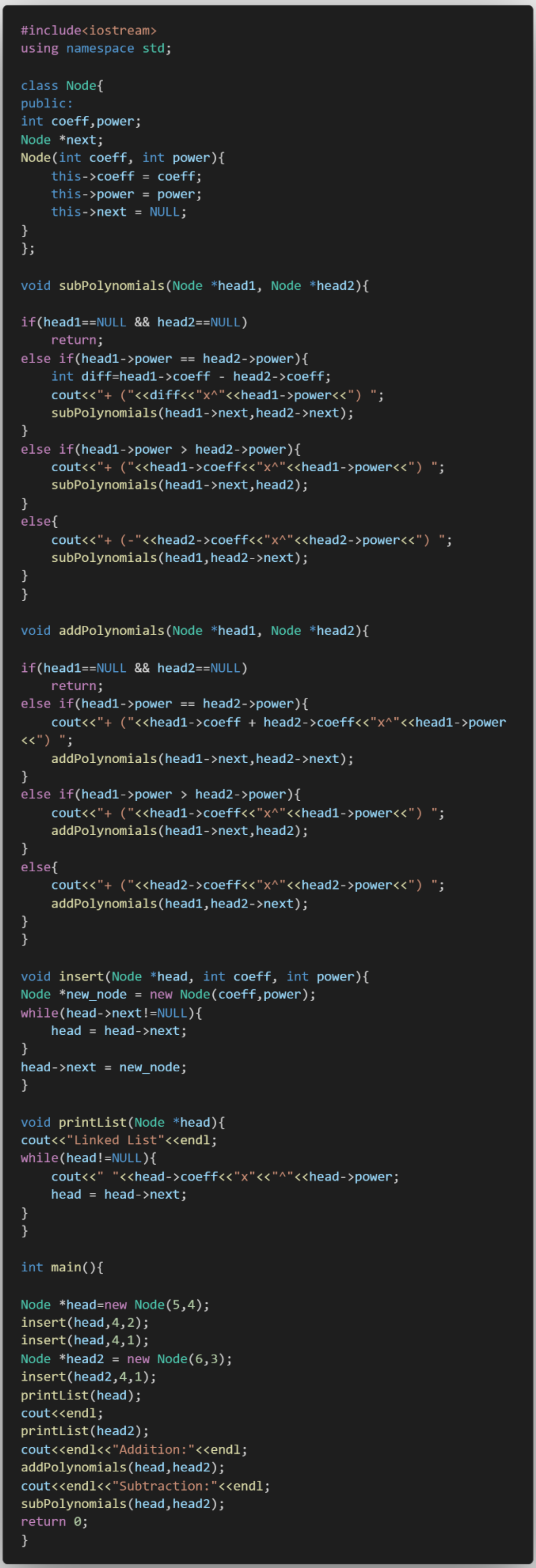
PRN: 2020BTEIT00041

NAME: Om Vivek Gharge

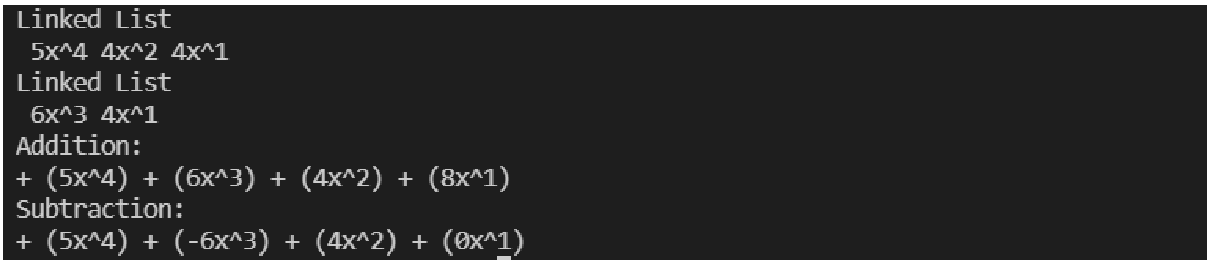
Q.) Evaluating value of Polynomial by adding the value of variable x

CODE:



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

class Node{
public:
int coeff,power;
Node *next;
Node(int coeff, int power){
    this->coeff = coeff;
    this->power = power;
    this->next = NULL;
}
};

void subPolynomials(Node *head1, Node *head2){

if(head1==NULL && head2==NULL)
    return;
else if(head1->power == head2->power){
    int diff=head1->coeff - head2->coeff;
    cout<<"+ ("<<diff<<"x^"<<head1->power<<") ";
    subPolynomials(head1->next,head2->next);
}
else if(head1->power > head2->power){
    cout<<"+ ("<<head1->coeff<<"x^"<<head1->power<<") ";
    subPolynomials(head1->next,head2);
}
else{
    cout<<"+ (-"<<head2->coeff<<"x^"<<head2->power<<") ";
    subPolynomials(head1,head2->next);
}
}

void addPolynomials(Node *head1, Node *head2){

if(head1==NULL && head2==NULL)
    return;
else if(head1->power == head2->power){
    cout<<"+ ("<<head1->coeff + head2->coeff<<"x^"<<head1->power
<<") ";
    addPolynomials(head1->next,head2->next);
}
else if(head1->power > head2->power){
    cout<<"+ ("<<head1->coeff<<"x^"<<head1->power<<") ";
    addPolynomials(head1->next,head2);
}
else{
    cout<<"+ ("<<head2->coeff<<"x^"<<head2->power<<") ";
    addPolynomials(head1,head2->next);
}
}

void insert(Node *head, int coeff, int power){
Node *new_node = new Node(coeff,power);
while(head->next!=NULL){
    head = head->next;
}
head->next = new_node;
}

void printList(Node *head){
cout<<"Linked List"<<endl;
while(head!=NULL){
    cout<<" "<<head->coeff<<"x"<<"^"<<head->power;
    head = head->next;
}
}

int main(){

Node *head=new Node(5,4);
insert(head,4,2);
insert(head,4,1);
Node *head2 = new Node(6,3);
insert(head2,4,1);
printList(head);
cout<<endl;
printList(head2);
cout<<endl<<"Addition:"<<endl;
addPolynomials(head,head2);
cout<<endl<<"Subtraction:"<<endl;
subPolynomials(head,head2);
return 0;
}
```

OUTPUT:



```
Linked List
 5x^4 4x^2 4x^1
Linked List
 6x^3 4x^1
Addition:
+ (5x^4) + (6x^3) + (4x^2) + (8x^1)
Subtraction:
+ (5x^4) + (-6x^3) + (4x^2) + (0x^1)
```

ALGORITHM:

Step 1: Algorithm to add two polynomials using linked list: Let p and q be the two polynomials represented by linked lists.

> Step 1. while p and q are not null, repeat step 2.
>
> Step 2. If powers of the two terms are equal then if the terms do not cancel then insert the sum of the terms into the sum Polynomial Advance p Advance q Else if the power of the first polynomial> power of second Then insert the term from first polynomial into sum polynomial Advance p Else insert the term from second polynomial into sum polynomial Advance q.
>
> Step 3. Copy the remaining terms from the non-empty polynomial into the sum polynomial. step 3 of the algorithm is to be processed till the end of the polynomials has not been reached.

Step 2: Algorithm to subtract two polynomials using linked list: Let p and q be the two polynomials represented by linked lists.

> Step 1. while p and q are not null, repeat step 2.
>
> Step 2. If powers of the two terms are equal then if the terms do not cancel then insert the difference of the terms into the difference Polynomial Advance p Advance q Else if the power of the first polynomial> power of second Then insert the term from first polynomial into difference polynomial Advance p Else insert the term from second polynomial into difference polynomial Advance q.
>
> Step 3. Copy the remaining terms from the non-empty polynomial into the difference polynomial. Step 3 of the algorithm is to be processed till the end of the polynomials has not been reached.