PIMPRI CHINCHWAD EDUCATION TRUST's.

PIMPRI CHINCHWAD COLLEGE OF ENGINEERING

(An Autonomous Institute)

S.Y. B. TECH **Year:** 2024 – 25 **Semester:** I

Name: Abhishek Joshi **PRN: 123B1B150**

Department: Computer Engineering **Division:** C (C1)

Course: Data Structures Laboratory **Course Code:** BCE23PC02

Date:

Assignment – 6

- **Aim:**

Consider two polynomial expressions of any degree. Design solution to perform addition of these two polynomials with suitable data structure and display results.

- **Source Code :**

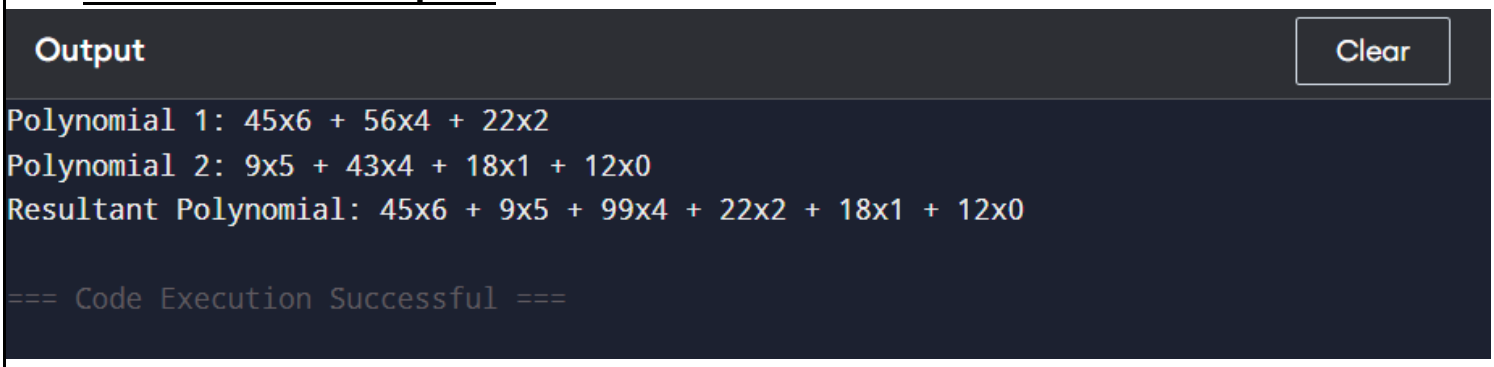
```
#include<iostream>
using namespace std;

class node{
   public:
      int  coeff;
      int pow;
      node* next;
      node(){
         coeff = 0;
         pow = 0;
         next = NULL;
      }
      node(int x, int y){
         coeff = x;
         pow = y;
         next = NULL;
      }
};
class LL{
   public:
      node* head = NULL;
      void create_node(int x, int y);
```

```
    void print_poly();
    void polyAdd(LL l1, LL l2);
};

void LL::create_node(int x, int y){
  node* nn = new node();
  nn->coeff = x;
  nn->pow = y;
  if(head == NULL){
    head = nn;
  }
  else{
    node* temp = new node();
    temp = head;
    while(temp->next != NULL){
      temp = temp->next;
    }
    temp->next = nn;
  }
}

void LL::polyAdd(LL l1, LL l2) {
  node* p1 = l1.head;
  node* p2 = l2.head;

  while (p1 != NULL && p2 != NULL) {
    if (p1->pow == p2->pow) {
      create_node(p1->coeff + p2->coeff, p1->pow);
      p1 = p1->next;
      p2 = p2->next;
    } else if (p1->pow > p2->pow) {
      create_node(p1->coeff, p1->pow);
      p1 = p1->next;
    } else {
      create_node(p2->coeff, p2->pow);
      p2 = p2->next;
    }
  }

  while (p1 != NULL) {
    create_node(p1->coeff, p1->pow);
```

```
        p1 = p1->next;
    }
    while (p2 != NULL) {
        create_node(p2->coeff, p2->pow);
        p2 = p2->next;
    }
}

void LL::print_poly(){
    if(head != NULL){
        node* temp = head;
        while(temp->next != NULL){
            if(temp->pow==0){

            }
            else{
                cout<<temp->coeff<<"x"<<temp->pow<<" + ";
            }
            temp=temp->next;
        }
        cout<<temp->coeff<<"x"<<temp->pow;
    }
}

int main(){
    LL l1;
    LL l2;
    LL result;
    l1.create_node(45, 6);
    l1.create_node(56, 4);
    l1.create_node(22, 2);
        cout<<"Polynomial 1: ";
        l1.print_poly();
        cout<<endl;
    l2.create_node(9, 5);
    l2.create_node(43, 4);
    l2.create_node(18, 1);
    l2.create_node(12, 0);
        cout<<"Polynomial 2: ";
        l2.print_poly();
        cout<<endl;
```

```
    result.polyAdd(l1, l2);
    cout << "Resultant Polynomial: ";
    result.print_poly();

  return 0;
  }
```

- **Screen Shot of Output :**



- **Conclusion:**

  Hence, we studied about application of Generalized Linked List as Polynomial Addition by addition of two linked lists and displaying the resultant polynomial expression with their algorithm and programs.