**Java Assignment 5**

Polynomial using Linked List and Polynomial Addition and Multiplication Operation

By:

Vaibhav Jaiswal

PRN:17070122071

Batch:CS-4

Branch: CSE

SY CSE (Batch of 2017-2021)

**CODE**

import java.util.\* ;

import java.util.Collections;

import java.util.LinkedList;

class Term{

int coeff ;

int pow ;

Term(){

coeff = 0 ;

pow = 0 ;

}

Term(int coeff , int pow){

this.coeff = coeff ;

this.pow = pow ;

}

public String toString(){

return (this.coeff + "x^" + this.pow ) ;

}

public static boolean samePow(Term t1 , Term t2){

return (t1.pow == t2.pow) ;

}

}

class Polynomial{

public static LinkedList<Term> multiplyPoly(LinkedList<Term> poly1 , LinkedList<Term> poly2 ){

LinkedList<Term> mulpoly = new LinkedList<Term>() ;

for (Term t1 : poly1){

for (Term t2 : poly2){

Term temp = new Term((t1.coeff\*t2.coeff) , (t1.pow + t2.pow)) ;

mulpoly.add(temp) ;

}

}

Polynomial.removeDuplicates(mulpoly);

return mulpoly ;

}

public static void removeDuplicates(LinkedList<Term> poly){

Collections.sort(poly , new SortPoly());

for (int i = 0 ; i < poly.size()-1 ; i++){

Term t1 = poly.get(i) ;

Term t2 = poly.get(i+1) ;

if(Term.samePow(t1, t2)){

int indx = poly.indexOf(t1) ;

poly.remove(t1) ;

poly.remove(t2) ;

t1.coeff = t1.coeff+t2.coeff ;

poly.add(indx , t1);

i--;

}

}

Collections.sort(poly , new SortPoly());

}

public static LinkedList<Term> addpoly(LinkedList<Term> poly1 , LinkedList<Term> poly2 ){

LinkedList<Term> addpoly = new LinkedList<Term>() ;

addpoly.addAll(poly1) ;

addpoly.addAll(poly2) ;

Polynomial.removeDuplicates(addpoly);

return addpoly;

}

}

class SortPoly implements Comparator<Term>{

public int compare(Term t1 , Term t2){

if(t1.pow < t2.pow)

return 1 ;

else return -1 ;

}

}

class Driver{

public static void main(String[] args) {

LinkedList<Term> poly1 = new LinkedList<Term>() ;

LinkedList<Term> poly2 = new LinkedList<Term>() ;

char another = 'y' ;

Scanner ip = new Scanner(System.in) ;

int coeff ,pow = 0 ;

System.out.println("\nEnter The Terms of Polynomial 1 : \n");

while (another=='y'){

System.out.println("Enter the coeff of term :");

coeff = ip.nextInt() ;

System.out.println("Enter the power of term :");

pow = ip.nextInt() ;

poly1.add(new Term(coeff , pow )) ;

System.out.println("Do you want to enter another term (y/n) :") ;

another = ip.next().charAt(0) ;

}

Collections.sort(poly1 , new SortPoly());

another = 'y' ;

System.out.println("Enter The Terms of Polynomial 2 : \n");

while (another=='y'){

System.out.println("Enter the coeff of term :");

coeff = ip.nextInt() ;

System.out.println("Enter the power of term :");

pow = ip.nextInt() ;

poly2.add(new Term(coeff , pow )) ;

System.out.println("Do you want to enter another term (y/n) :") ;

another = ip.next().charAt(0) ;

}

Collections.sort(poly2 , new SortPoly());

System.out.println("Polynomial 1 : " + poly1);

System.out.println("\nPolynomial 2 : " + poly2);

LinkedList<Term> addpoly = Polynomial.addpoly(poly1 , poly2) ;

System.out.println("\nResult of Addition of Both Polynomials : " + addpoly);

LinkedList<Term> mulpoly = Polynomial.multiplyPoly(poly1 , poly2 ) ;

System.out.println("\nResult of Multiplication of Both Polynomials : " + mulpoly);

ip.close();

}

}

**Output**

