```
public class Node
  {
  private int exp;
  private double coeff;
  private Node next;

public Node()
{}
public Node(double coff,int ex) {
    exp = ex;
    coeff = coff;
    next = null;
  } // end constructor
   public Node(double coff,int ex,Node newnode) {
    exp = ex;
    coeff = coff;
    next = newnode;
  } // endnd constructor

  public void setExp(int newItem)
  {
    exp = newItem;
  }
  public void setCoeff(double newitem)
{
            coeff = newitem;
}
  public int getExp() {
    return exp;
  } // end getItem
```

```
  public double getCoeff()
  {
    return coeff;
  }

  public void setNext(Node nextNode)
  {
    next = nextNode;
  } // end setNext*/

  public Node getNext()
  {
    return next;
   } // end getNext
}

public class Prog1
   {
```

```
public Node head;


public void add(int index, double coff,int ex) // add Node at given index
   {
     if (index == 1) //add at head
      {
       Node newNode = new Node(coff,ex, head);
       head = newNode;
      } // end if
   else
      {
       Node prev = find(index-1);
       Node newNode = new Node(coff,ex, prev.getNext());
       prev.setNext(newNode);
      } //end else

    } //end add

private Node find(int index)  // find the node at given index
   {

      Node curr = head;
      for (int skip = 1; skip < index; skip++)
       {
          curr = curr.getNext();
       } // end for
    return curr;
   } // end find

public void remove(int index) // remove node from given index
```

```
    {
      if (index >= 1 )
       {
         if(index == 1)
           {
     // delete the first node from the list
     head = head.getNext();
           }
         else
           {
      Node prev = find(index-1);
     // delete the node after the node that prev
     // references, save reference to node
     Node curr = prev.getNext();
     prev.setNext(curr.getNext());
           } // end if
   } // end if
}

  public void removeZeroNodes()  // remove nodes those have coeff as zero
    {
      Node        curr = head;
           int index = 1;
           while(curr != null)
            {
                    if(curr.getCoeff() == 0)
                    {
                      curr = curr.getNext();
                       remove(index);
                    }// end if
```

```
                    else
                    {
                      index++;
                            curr = curr.getNext();
                    } //end else
              } // end while
       }   // end remove
  } //end class


   import java.util.*;
import java.io.*;

public class ProgTest
{         Node head ;
          public static  void main(String args[]) throws IOException
          {
                    Prog1 prog1 = new Prog1();
                    Prog1 prog2 = new Prog1();

                    Writer output;
                    File file = new File("write.txt");
                    output = new BufferedWriter(new FileWriter(file));
                    File a = new File("1"+ ".txt");
              Scanner in = new Scanner( a );
                    int i=1;
                    String b = in.nextLine();
                    String result[];
                    while ( b.length()!= 0 )      // reads input from file and assign values to polynomial1(
linkedlist1)
```

```
                    {
                      result = b.split("\\s");
                     //  System.out.println( " value1 " +result[1]);
                      // System.out.println( " value2 " +result[3]);
                              prog1.add(i,Double.parseDouble(result[1]),Integer.parseInt(result[3]));

                              b = in.nextLine();
                              i++;
                    }
                      i=1;
                   while (  in.hasNextLine()  )    //    reads input from file and assign values to
polynomial2( linkedlist2)
                    {
                      b = in.nextLine();
                              result = b.split("\\s");
                    //        System.out.println( " value1 " +result[1]);
                    //        System.out.println( " value2 " +result[3]);
                              prog2.add(i,Double.parseDouble(result[1]),Integer.parseInt(result[3]));
                              i++;
               }

            ProgTest ob = new ProgTest();
            Node ref = ob.polyMult(prog1.head,prog2.head);
             double c =    ob.polyEval(ref,3);
             //System.out.println( "eval = " + c );

       //output = new BufferedWriter(new FileWriter(file));

           while( ref != null)
        {
```

```
                output.write(" ( " +ref.getCoeff() + "," + ref.getExp() + " ) \n"); // writes output to file
                ref = ref.getNext();
        }

    output.close();
        } // end main



double power(double x, int y)    // caluculate x^y;
  {
    double value = 1;
            int i;
    for(i = 0; i< y; i++)
    value = value * x;
    return value;
  }

double polyEval(Node p,double x)  // caluculates the value of polynomial at given value
         {
       Node cur = p;
       double sum = 0;
          while(cur != null)
          {
            sum = sum + cur.getCoeff() * (power(x,cur.getExp()));

           // System.out.println("sum " + sum + " " +cur.getCoeff() * (power(x,cur.getExp())) );
            cur = cur.getNext();
          }
       return sum;
```

```
    }

  public  Node polyMult(Node p1,Node p2)  // calucilates p1 * p2  and returns head of the resulting polynomial
  {
     Node ptr1 = p1;
     Node ptr2 = p2;
     Prog1 prog3   = new Prog1();

     if(ptr1 == null || ptr2 == null)
      return null;

     else
     {

       while(ptr1 != null)
       {
        ptr2 = p2;
         while(ptr2 !=null)
                  {
            int c = ptr1.getExp() + ptr2.getExp();
            double d = ptr1.getCoeff() * ptr2.getCoeff();

                int index = 1;
                Node  curr = prog3.head;


           while(curr != null )
                   {
              if(curr.getExp() == c)
```

```
                {
               curr.setCoeff  (curr.getCoeff() + d);
               break;
                }

               else if(curr.getExp() < c)
                {
                  curr = curr.getNext();
                  index++;
                }
               else if(curr.getExp() > c)
                {
                  prog3.add(index,d,c);
                     break;

                }
               else ;

             }//end while 3
           if( curr == null)
            {

             prog3.add(index,d,c);

            }
             ptr2 = ptr2.getNext();
     }//end while 2
  ptr1 = ptr1.getNext();
 }//end while 1
```

```
    prog3.removeZeroNodes();
    return prog3.head;
      }// end else
    }  // end polymult
}
```