```
import java.util.*;

public class DiffEq{
   public static double x0, y0, x;
   public static String DE[];
   public static String RHS[];
   public static int count;


   public DiffEq(double x0, double y0, double x, String DE, int count){
   /* Constructor to initialize variables */

      this.x0 = x0;
      this.y0 = y0;
      this.x = x;
      this.DE = DE.split(" ");
      RHS = new String[this.DE.length-2];
      for(int i=0; i<this.DE.length-2; i++){
         RHS[i] = this.DE[i+2];
      }//for
      this.count = count;

   }//DiffEq()


public static String[] f(String s, String arr[], char ch){
   /* Evaluates Function s symbolically at arr for variable ch */

      int count = 0;
      for(int i=0; i<s.length(); i++){
         if(s.charAt(i)==ch) count++;
      }//for
      if(count==0){//if s is not a function of variable ch
         String a[] = {s};
         return a;
      }//if

      String a[] = new String[arr.length];

      for(int i=0; i<arr.length; i++){

         int indexPow = 0;
         for(int j=0; j<arr[i].length(); j++){
            if(arr[i].charAt(j)=='^'){
               indexPow = j;
            }//if
         }//for
```

```
String ai = arr[i].substring(0,indexPow);

boolean b = false; //for s
boolean b1 = false; //for arr[i]
double d = 0;
try{// s is a number
   d = Double.parseDouble(s);
   String ar[] = {s};
   b = true; //s is a number
   return ar;
}//try
catch(Exception e){
}//catch

if(!b){// s is not a number
   int index = 0;

   for(int k=0; k<s.length(); k++){
      if( s.charAt(k)==ch ){
         index = k;
      }//if
   }//for

   if(index==0 && s.charAt(0)==ch ){// s is the variable, ch
      return arr;
   }//if
   else if(s.length()!=1){// s is of the form f(y) = ay, a = real number
      double coef = Double.parseDouble( f(s,1.0,ch) );

      try{// arr[i] is a number
         double d1 = Double.parseDouble(arr[i]);
         a[i] = f(s,d1,ch);
         b1 = true; //arr[i] is a number
         continue;
      }//try
      catch(Exception e){
      }//catch

      double aiCoef = 0;
      char ch1 = ' ';
      for(int k=0; k<ai.length();k++){
         if( Character.isLetter( ai.charAt(k) ) ){
            ch1 = ai.charAt(k);
            aiCoef = Double.parseDouble( f(arr[i], 1.0, ch1) );
         }//if
      }//for
```

```
            if(arr[i].length()==1 && !b1){ //arr[i] is not a number
               a[i] = coef + arr[i];
            }//if
            else if(indexPow!=0){
               a[i] = (coef * aiCoef ) + arr[i].substring(ai.length()-1);
            }//else if
            else{
               a[i] = (coef * aiCoef ) + arr[i].substring(arr[i].length()-1);
            }//else
         }//else if
      }//if
   }//for
   return a;
}//f()


public static String f(String s, double d, char ch){
/* Evaluates Function at d for variable ch*/

   if(s.length()==1 && Character.isDigit(s.charAt(0))){
      return Double.parseDouble(s.substring(0,1)) + "";
   }//if
   else if( (s.length()==1) && (s.charAt(0)==ch) ){
      return d + "";
   }//else

   int index=0;
   boolean flag = false;
   for(int i=0;i<s.length();i++){
      if(s.charAt(i)=='E') continue;
      else if(s.charAt(i)==ch){
         index=i;
         flag = true;
         break;
      }//else if
   }//for

   if(!flag && ( s.equals("0^0.0") || s.equals("0.0^0.0") ) ){
      return 0 + "";
   }//if
   else if(s.equals(null)){
      return 0 + "";
   }//else if
   try{
      if(s.substring(0,3).equals("0.0")){
         return 0 + "";
      }//else if
```

```
      }//try
      catch(StringIndexOutOfBoundsException e){
      }//catch()
      if(!flag){
         return s;
      }//else if
      else if(index==0 && flag){
         return Math.pow(d,Double.parseDouble(s.substring(2))) + "";
      }//else if
      else if(index==s.length()-1){
         if(s.charAt(0)=='+' && index==1){
            return d + "";
         }//if
         else if(s.charAt(0)=='-' && index==1){
         return (-1*d) + "";
         }//else if
         else{
            return Double.parseDouble(s.substring(0,index))*d + "";
         }//else
      }//else if
      else{
         if(s.charAt(0)=='+' && index==1){
            return Math.pow(d,Double.parseDouble(s.substring(index+2))) + "";
         }//if
         else if(s.charAt(0)=='-' && index==1){
         return ( (-1) * Math.pow( d,Double.parseDouble( s.substring(index+2) ) ) ) + "";
         }//else if
         return
Double.parseDouble(s.substring(0,index))*Math.pow(d,Double.parseDouble(s.substring(ind
ex+2))) + "";
      }//else
   }//f()


   public static String[] function(String RHS[], String arr[], char ch){ //***
   /* Calls f() */
      if(RHS.length==1) return f(RHS[0], arr, ch); //***
      String a[] = function(Arrays.copyOfRange(RHS, 1, RHS.length), arr, ch);
      return f(RHS[0], arr, ch); //***
   }//function


   public static String function(String arr[], double d, char ch){
   /* Calls f() */

      if(arr.length==1) return f(arr[0], d, ch)+"";
```

```
      return (Double.parseDouble(f(arr[0], d, ch)) +
Double.parseDouble(function(Arrays.copyOfRange(arr, 1, arr.length),d, ch))) + "";
   }//function


   public static String[] StrIntegrate(String integrand[]){
   /* Function to Symbolically Integrate */

      int l = integrand.length;
      double coef[] = new double[l];
      char var[] = new char[l];
      double power[] = new double[l];
      for(int i=0; i<l; i++){
         for(int j=0; j<integrand[i].length(); j++){
            if( Character.isLetter(integrand[i].charAt(j)) ){
               var[i] = integrand[i].charAt(j);

               if( (integrand[i].charAt(0)=='+') && (j==1) ) coef[i] = +1;
               else if( (integrand[i].charAt(0)=='-') && (j==1) ) coef[i] = -1;
               else if(j!=0) coef[i] = Double.parseDouble(integrand[i].substring(0,j));
               else coef[i] = 1;

               if(j==integrand[i].length()-1) power[i] = 1;
               else if(integrand[i].charAt(j+1)=='^'){
                  power[i] = Double.parseDouble(integrand[i].substring(j+2));
               }//else if
               else power[i] = 1;
            }//if
         }//for

         if( integrand[i].equals("0^0.0") || integrand[i].equals("0.0^0.0") ){
            coef[i] = 0;
            var[i] = 'x';
            power[i] = 0;
         }//if

         try{
            coef[i] = Double.parseDouble(integrand[i]);
            var[i] = 'x';
            power[i] = 0;
         }//try
         catch(NumberFormatException e){
            continue;
         }//catch

      }//for
```

```
   /* Now Integrate Symbolically! */
   for(int i=0; i<l; i++){
      if(var[i]=='x'){
         power[i] = power[i] + 1;
         coef[i] = coef[i] / power[i];
      }//if
   }//for

   String result[] = new String[l+1];
   for(int i=0; i<l; i++){
      result[i] = coef[i] + "" + var[i] + "^" + power[i];
   }//for

   double constant = 0;
   for(int i=0; i<l; i++){
      if(var[i]=='x'){
         constant -=  Double.parseDouble(f(result[i],x0,'x'));
      }//if
   }//for
   result[l] = constant + "x^0";

   return result;
}//StrIntegrate()


public static String[] y_nMinus1(String RHS[], int count){
   if(count==1){
      String a[] = {"" + y0};
      return a;
   }//if
   else{
      String a[] = Picard(RHS, 1, --count);
      return a;
   }//else
}//y_nMinus1


public static String[] Picard(String RHS[], int count, int count1){
/* Function implementing Picard's iteration using upward recursion */

   if(count==count1){// y(count)
      String[] y_nMinus1 = y_nMinus1(RHS, count);
      int j=0;
      for(int i=0; i<RHS.length; i++){
         j += f( RHS[i], y_nMinus1, 'y').length;
      }//for
      String arr[] = new String[j]; //arr = Integrated Array to store terms evaluted at y(n-1)
```

```
      int k=0;
      for(int i=0; i<RHS.length && k<j; i++){
         int l=0;
         String ar[] = f( RHS[i], y_nMinus1, 'y');
         while(l<ar.length){
            arr[k] = ar[l];
            k++;
            l++;
         }//while
         l=0;
      }//for
      String a[] = StrIntegrate(arr);
      int l = a.length;
      for(int i=0; i<l; i++){
         a[i] = ( Double.parseDouble(f(a[i], x, 'x')) - Double.parseDouble(f(a[i], x0, 'x')) ) + "";
      }//for
      a[l-1] = (Double.parseDouble(f(a[l-1], x, 'x')) + y0) + "";
      return a;
   }//if
   else{// y(n)
      String[] y_nMinus1 = y_nMinus1(RHS, count);
      int l1 = y_nMinus1.length;
      int j=0;
      for(int i=0; i<RHS.length; i++){
         j += f( RHS[i], y_nMinus1, 'y').length;
      }//for
      String arr[] = new String[j]; //arr = Integrated Array to store terms evaluted at y(n-1)
      int k=0;
      for(int i=0; i<RHS.length && k<j; i++){
         int l=0;
         String ar[] = f( RHS[i], y_nMinus1, 'y');
         while(l<ar.length){
            arr[k] = ar[l];
            k++;
            l++;
         }//while
         l=0;
      }//for
      String a[] = StrIntegrate(arr);
      int l = a.length;
      a[l-1] = (Double.parseDouble(f(a[l-1], x, 'x')) + y0) + "";
      System.out.println(Arrays.toString(a)); //***
      return Picard(RHS, ++count, count1); //***
   }//else if
}//Picard()
```

```
   public static void main(String [] args){
   /* Main method to call functions */

      Scanner in = new Scanner(System.in);
      System.out.println("Enter x0 of initial data (y(x0)=y0)");
      double x0 = in.nextDouble();
      System.out.println("\nEnter y0 of initial data (y(x0)=y0)");
      double y0 = in.nextDouble();
      System.out.println("\nEnter the value of x for which y is to be calculated");
      double x = in.nextDouble();
      System.out.println("\nInput the differential equation. Look at the input requirements
below");

      System.out.println("\nThe differential equation (given in the exact format as the
example)");
      System.out.println("with + and - operators separated by a space before but not after");
      System.out.println("(Example: y' = y^3 +3y^2 -5, gap before and after =). Note: y' is");
      System.out.println("expressed as the sole term on the left-hand side and its
coefficient");
      System.out.println("is 1. Define the derivative as y' only. The differential equation
should");
      System.out.println("only involve variables x and y and their powers. Special functions
should not be");
      System.out.println("used. Example: Don't use sin, log or e\n");

      String DE1 = in.nextLine();
      String DE = in.nextLine();
      System.out.println("\nEnter the number of Picard iterations to solve the differential
equation\n");
      int count = in.nextInt();

      DiffEq de = new DiffEq(x0, y0, x, DE, count);
      String a[] = de.Picard(de.RHS, 1, count);
      double yN = 0;
      for(int i=0; i<a.length; i++){
         yN += Double.parseDouble(a[i]);
      }//for
      System.out.println(yN);
   }//main()
}//class
```