



TECNOLÓGICO
NACIONAL DE MÉXICO



TECNOLÓGICO NACIONAL DE MÉXICO
INSTITUTO TECNOLÓGICO DE CIUDAD MADERO



TECNOLÓGICO NACIONAL DE MÉXICO

INSTITUTO TECNOLÓGICO DE CIUDAD MADERO

Carrera: Sistemas Computacionales

Tema: Práctica 5

Equipo 3:

Reyes Villar Luis Ricardo

Garcia Valles Roberto Carlos

Lara Hernández Juan Jesús

Rocha Suarez María Fernanda

Hernández del Ángel Ángel Ivan

Profesora: Claudia Lizeth Castillo Ramírez

Materia: Métodos Numéricos

Hora: 14:00 – 15:00hrs

Grupo: 5501B

Semestre: 4to

Ciclo Escolar: Enero 2023 – Junio 2023

Especificaciones del problema.

Para el problema de Interpolación se Newton, se busca resolver el problema por el método nombrado Diferencias divididas, este método ayuda a aproximar la derivada y la integral de una función, respectivamente en el polinomio de interpolación

Se representa con la siguiente formula:

$$f(X_i, X_{i+1}) = \frac{f(X_{i+1}) - f(X_i)}{X_{i+1} - X_i}$$

Para esto, tenemos que tener como dato una función y sus respectivos valores X_i , al resolver la función por cada valor, se procede a resolver la función inferior menos la función superior sobre la función superior, y esto se realiza dependiendo el numero de Valores X_i hasta que ya no se puede más.

Para el problema de Interpolación de Lagrange tenemos que nos permite construir fácilmente de forma explicita, el polinomio interpolador. Este método es simplemente una reformulación del polinomio de Newton, lo cual explica el porqué son necesarios los mismos datos. Sin embargo, este método evita los cálculo de las diferencias divididas.

Este método consiste en construir el polinomio interpolador de grado n que pasa por $n+1$ con punta (X_i, Y_i) de la forma:

$$P_n(x) = \sum_{\substack{i=0 \\ j \neq i}}^n \prod_{i=0}^n \frac{(X - X_i)}{(X_i - X_j)}$$

La siguiente formula correspondiente al polinomio interpolador es:

$$P(x) = \frac{(X - X_1)(X - X_2)}{(X_0 - X_1)(X_0 - X_2)}(f_0) + \frac{(X - X_0)(X - X_2)}{(X_1 - X_0)(X_1 - X_2)}(f_1) + \frac{(X - X_0)(X - X_1)}{(X_2 - X_0)(X_2 - X_1)}(f_2)$$

Análisis.

Diagrama de flujo:



Programación.

Para poder representar el algoritmo en un lenguaje de programación, se optó por utilizar java para obtener las soluciones del Polinomio de Interpolación de Newton y del Polinomio de Interpolación de Lagrange.

Método Main.

```
1 package practica5_mn;
2
3 import javafx.application.Application;
4 import static javafx.application.Application.launch;
5 import javafx.fxml.FXMLLoader;
6 import javafx.scene.Parent;
7 import javafx.scene.Scene;
8 import javafx.stage.Stage;
9
10 /**
11  *
12  * @author juani
13  */
14 public class Practica5_MN extends Application {
15
16     @Override
17     public void start(Stage stage) throws Exception {
18         Parent root =
19             FXMLLoader.load(getClass().getResource("Practica5_MN.fxml"));
20
21         Scene scene = new Scene(root); // attach scene graph to scene
22         stage.setTitle("Programa Interpolación y Ajuste de Funciones"); // displayed in window's title bar
23         stage.setScene(scene); // attach scene to stage
24         stage.show(); // display the stage
25     }
26
27     public static void main(String[] args) {
28         // create a Welcome object and call its start method
29         launch(args);
30     }
31 }
32
```

Clase Controladora.

```
1 package practica5_mn;
2 import javafx.event.ActionEvent;
3 import javafx.fxml.FXML;
4 import javafx.scene.control.Alert;
5 import javafx.scene.control.Alert.AlertType;
6 import javafx.scene.control.Button;
7 import javafx.scene.control.TextField;
8 import javafx.scene.image.ImageView;
9
10 public class Practica5_MNController {
11
12     @FXML
13     private ImageView funcionInterpolacionNewtonImageView;
14
15     @FXML
16     private TextField fx0TextField;
17
18     @FXML
19     private TextField fx1TextField;
20
21     @FXML
22     private TextField fx2TextField;
23
24     @FXML
25     private Button newtonInterpolationButton;
26
27     @FXML
28     private Button lagrangeInterpolationButton;
29
30     @FXML
31     private TextField x0TextField;
32
33     @FXML
34     private TextField x1TextField;
35
36     @FXML
37     private TextField x2TextField;
38
39     @FXML
40     void newtonInterpolationCalc(ActionEvent event) {
41         NodeFunctionOfX x0 = new NodeFunctionOfX(new FunctionOfX(valueOfX: 0));
42         NodeFunctionOfX x1 = new NodeFunctionOfX(new FunctionOfX(valueOfX: 1));
43         NodeFunctionOfX x2 = new NodeFunctionOfX(new FunctionOfX(valueOfX: 2));
44         NodeFunctionOfX x3 = new NodeFunctionOfX(new FunctionOfX(valueOfX: 3));
45
46         NodeFunctionOfX x0x1 = new NodeFunctionOfX(new DivDiffFunctionOfX(valueXm:x0.getData(), valueXn:x1.getData()));
47         x0x1.setLeft(left: x0);
48         x0x1.setRight(right: x1);
49     }
```

```

50 NodeFunctionOfX x1x2 = new NodeFunctionOfX(new DivDiffFunctionOfX(valueOf(x1.getData()), valueOf(x2.getData())));
51 x1x2.setLeft(left: x1);
52 x1x2.setRight(right: x2);
53
54 NodeFunctionOfX x2x3 = new NodeFunctionOfX(new DivDiffFunctionOfX(valueOf(x2.getData()), valueOf(x3.getData())));
55 x2x3.setLeft(left: x2);
56 x2x3.setRight(right: x3);
57
58 NodeDivDiffFunctionOfX x0x1x2 = new NodeDivDiffFunctionOfX((new DivDiffFunctionOfX((DivDiffFunctionOfX) x0x1.getData(), (DivDiffFunctionOfX) x1x2.getData())));
59 x0x1x2.setLeft(left: x0x1);
60 x0x1x2.setRight(right: x1x2);
61
62 NodeDivDiffFunctionOfX x1x2x3 = new NodeDivDiffFunctionOfX((new DivDiffFunctionOfX((DivDiffFunctionOfX) x1x2.getData(), (DivDiffFunctionOfX) x2x3.getData())));
63 x1x2x3.setLeft(left: x1x2);
64 x1x2x3.setRight(right: x2x3);
65
66 NodeDivDiffFunctionOfX x0x1x2x3 = new NodeDivDiffFunctionOfX((new DivDiffFunctionOfX(valueOf(x0x1x2.getData()), valueOf(x1x2x3.getData()))));
67 x0x1x2x3.setLeft(left: x0x1x2);
68 x0x1x2x3.setRight(right: x1x2x3);
69
70 Alert alert = new Alert(AlertType.INFORMATION);
71 alert.setTitle(getString("Resultados de Polinomio de Interpolación de Newton"));
72 alert.setHeaderText(getString("null"));
73 alert.setContentText(getString((new InterpolacionNewton(x0x1x2x3)).toString()));
74 alert.showAndWait();
75
76
77
78 @FXML
79 void lagrangeInterpolationButtonCalc(ActionEvent event) {
80
81     int[] valoresX = new int [3];
82     int[] valoresFx = new int [3];
83
84     valoresX[0] = Integer.parseInt(x0TextField.getText());
85     valoresX[1] = Integer.parseInt(x1TextField.getText());
86     valoresX[2] = Integer.parseInt(x2TextField.getText());
87
88     valoresFx[0] = Integer.parseInt(fx0TextField.getText());
89     valoresFx[1] = Integer.parseInt(fx1TextField.getText());
90     valoresFx[2] = Integer.parseInt(fx2TextField.getText());
91
92
93     InterpolacionLagrange interpolacionL = new InterpolacionLagrange(valoresX, valoresFx);
94     PolinomioCuadratico polinomioLagrange = interpolacionL.PolinomioInterpolador(valoresX, valoresFx);
95
96     Alert alert = new Alert(AlertType.INFORMATION);
97     alert.setTitle(getString("Resultados de Polinomio de Interpolación de Lagrange"));
98
99     alert.setHeaderText(getString("null"));
100     alert.setContentText((PolinomioLagrange.toString()));
101     alert.showAndWait();
102
103 }
104 public void initialize() {
105
106
107
108 }
109

```

Archivo FXML

```
1 <?xml version="1.0" encoding="UTF-8"?>
2
3 <?import java.lang.*?>
4 <?import javafx.geometry.*?>
5 <?import javafx.scene.control.*?>
6 <?import javafx.scene.image.*?>
7 <?import javafx.scene.layout.*?>
8 <?import javafx.scene.text.*?>
9 <?import javafx.geometry.Insets?>
10 <?import javafx.scene.control.Button?>
11 <?import javafx.scene.control.Label?>
12 <?import javafx.scene.control.TextField?>
13 <?import javafx.scene.image.Image?>
14 <?import javafx.scene.image.ImageView?>
15 <?import javafx.scene.layout.BorderPane?>
16 <?import javafx.scene.layout.ColumnConstraints?>
17 <?import javafx.scene.layout.GridPane?>
18 <?import javafx.scene.layout.RowConstraints?>
19 <?import javafx.scene.layout.VBox?>
20 <?import javafx.scene.text.Font?>
21
22 <VBox alignment="CENTER" prefHeight="500.0" prefWidth="700.0" xmlns="http://javafx.com/javafx/8" xmlns:fx="http://javafx.com/fxml/1" fx:controller="practica5_mn.Practica5_MNController">
23     <children>
24         <BorderPane prefHeight="200.0" prefWidth="200.0">
25             <bottom>
26                 <Button fx:id="newtonInterpolationButton" mnemonicParsing="false" onAction="#newtonInterpolationCalc" text="Calcular" BorderPane.alignment="CENTER">
27                     <BorderPane.margin>
28                         <Insets bottom="10.0" left="10.0" right="10.0" top="10.0" />
29                     </BorderPane.margin>
30                 </Button>
31             </bottom>
32             <top>
33                 <Label text="Polinomio de Interpolación de Newton" BorderPane.alignment="CENTER">
34                     <padding>
35                         <Insets bottom="10.0" left="10.0" right="10.0" top="10.0" />
36                     </padding>
37                     <font>
38                         <font name="Book Antiqua" size="24.0" />
39                     </font>
40                 </Label>
41             </top>
42             <left>
43                 <Label alignment="CENTER" contentDisplay="CENTER" BorderPane.alignment="CENTER">
44                     <font>
45                         <font name="Berlin Sans FB" size="18.0" />
46                     </font>
47                     <BorderPane.margin>
48                         <Insets bottom="10.0" left="10.0" right="10.0" top="10.0" />
49                     </BorderPane.margin>
50                 </Label>
51             </left>
52             <center>
53                 <ImageView fx:id="functionInterpolacionNewtonImageView" fitHeight="150.0" fitWidth="200.0" pickOnBounds="true" preserveRatio="true" BorderPane.alignment="CENTER">
54                     <BorderPane.margin>
55                         <Insets bottom="10.0" left="10.0" right="10.0" top="10.0" />
56                     </BorderPane.margin>
57                     <image>
58                         <Image url="@../res/Function_Practica5_MN.JPG" />
59                     </image>
60                 </ImageView>
61             </center>
62             <VBox.margin>
63                 <Insets />
64             </VBox.margin>
65             <padding>
66                 <Insets bottom="10.0" left="10.0" right="10.0" top="10.0" />
67             </padding>
68         </BorderPane>
69         <BorderPane prefHeight="200.0" prefWidth="200.0">
70             <top>
71                 <Label text="Polinomio de Interpolación de Lagrange" BorderPane.alignment="CENTER">
72                     <font>
73                         <font name="Book Antiqua" size="24.0" />
74                     </font>
75                     <BorderPane.margin>
76                         <Insets />
77                     </BorderPane.margin>
78                 </Label>
79             </top>
80             <center>
81                 <GridPane alignment="CENTER" hgap="10.0" maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity" minWidth="-Infinity" vgap="10.0" BorderPane.alignment="CENTER">
82                     <columnConstraints>
83                         <ColumnConstraints hgrow="SOMETIMES" minWidth="10.0" prefWidth="100.0" />
84                         <ColumnConstraints hgrow="SOMETIMES" minWidth="10.0" prefWidth="100.0" />
85                         <ColumnConstraints hgrow="SOMETIMES" minWidth="10.0" prefWidth="100.0" />
86                     </columnConstraints>
87                     <rowConstraints>
88                         <RowConstraints minHeight="10.0" prefHeight="30.0" vgrow="SOMETIMES" />
89                         <RowConstraints minHeight="10.0" prefHeight="30.0" vgrow="SOMETIMES" />
90                         <RowConstraints minHeight="10.0" prefHeight="30.0" vgrow="SOMETIMES" />
91                         <RowConstraints minHeight="10.0" prefHeight="30.0" vgrow="SOMETIMES" />
92                     </rowConstraints>
93                     <children>
94                         <TextField alignment="CENTER" GridPane.columnIndex="1" GridPane.rowIndex="1" fx:id="x0TextField">
95                             <GridPane.margin>
96                                 <Insets />
97                             </GridPane.margin>
98                         </TextField>
```

```

99 <TextField fx:id="fx0TextField" alignment="CENTER" GridPane.columnIndex="2" GridPane.rowIndex="1">
100     <GridPane.margin>
101         <Insets />
102     </GridPane.margin>
103 </TextField>
104 <TextField alignment="CENTER" GridPane.columnIndex="1" GridPane.rowIndex="2" fx:id="x1TextField">
105     <GridPane.margin>
106         <Insets />
107     </GridPane.margin>
108 </TextField>
109 <TextField fx:id="fx1TextField" alignment="CENTER" GridPane.columnIndex="2" GridPane.rowIndex="2">
110     <GridPane.margin>
111         <Insets />
112     </GridPane.margin>
113 </TextField>
114 <TextField alignment="CENTER" GridPane.columnIndex="1" GridPane.rowIndex="3" fx:id="x2TextField">
115     <GridPane.margin>
116         <Insets />
117     </GridPane.margin>
118 </TextField>
119 <TextField fx:id="fx2TextField" alignment="CENTER" GridPane.columnIndex="2" GridPane.rowIndex="3">
120     <GridPane.margin>
121         <Insets />
122     </GridPane.margin>
123 </TextField>
124 <Label text="Xi" GridPane.columnIndex="1">
125     <padding>
126         <Insets bottom="10.0" left="10.0" right="10.0" top="10.0" />
127     </padding>
128     <GridPane.margin>
129         <Insets bottom="10.0" left="10.0" right="10.0" top="10.0" />
130     </GridPane.margin>
131     <font>
132         <Font name="Bookman Old Style" size="14.0" />
133     </font>
134 </Label>
135 <Label text="f(Xi)" GridPane.columnIndex="2">
136     <padding>
137         <Insets bottom="10.0" left="10.0" right="10.0" top="10.0" />
138     </padding>
139     <GridPane.margin>
140         <Insets bottom="10.0" left="10.0" right="10.0" top="10.0" />
141     </GridPane.margin>
142     <font>
143         <Font name="Bookman Old Style" size="14.0" />
144     </font>
145 </Label>
146 <Label text="i = 0" GridPane.rowIndex="1">
147     <padding>

```



```

148         <Insets bottom="10.0" left="10.0" right="10.0" top="10.0" />
149     </padding>
150     <GridPane.margin>
151         <Insets bottom="10.0" left="10.0" right="10.0" top="10.0" />
152     </GridPane.margin>
153     <font>
154         <Font name="Bookman Old Style" size="14.0" />
155     </font>
156 </Label>
157 <Label text="i = 1" GridPane.rowIndex="2">
158     <GridPane.margin>
159         <Insets bottom="10.0" left="10.0" right="10.0" top="10.0" />
160     </GridPane.margin>
161     <padding>
162         <Insets bottom="10.0" left="10.0" right="10.0" top="10.0" />
163     </padding>
164     <font>
165         <Font name="Bookman Old Style" size="14.0" />
166     </font>
167 </Label>
168 <Label text="i = 2" GridPane.rowIndex="3">
169     <GridPane.margin>
170         <Insets bottom="10.0" left="10.0" right="10.0" top="10.0" />
171     </GridPane.margin>
172     <padding>
173         <Insets bottom="10.0" left="10.0" right="10.0" top="10.0" />
174     </padding>
175     <font>
176         <Font name="Bookman Old Style" size="14.0" />
177     </font>
178 </Label>
179 </children>
180 <BorderPane.margin>
181     <Insets bottom="10.0" left="10.0" right="10.0" top="10.0" />
182 </BorderPane.margin>
183 </GridPane>
184 </center>
185 <bottom>
186     <Button fx:id="lagrangeInterpolationButton" mnemonicParsing="false" onAction="#lagrangeInterpolationButtonCalc" text="Calcular" BorderPane.alignment="CENTER">
187         <BorderPane.margin>
188             <Insets bottom="10.0" left="10.0" right="10.0" top="10.0" />
189         </BorderPane.margin>
190     </Button>
191 </bottom>
192 </BorderPane>
193 </children>
194 </VBox>
195

```

Clase Polinomio Cuadrático.

```
1
2 package practica5_mn;
3 public class PolinomioCuadratico {
4     private double coeficienteCuadratico;
5     private double coeficienteLineal;
6     private double coeficienteIndependiente;
7
8     public PolinomioCuadratico() {
9     }
10
11     public PolinomioCuadratico(double coeficienteCuadratico, double coeficienteLineal, double coeficienteIndependiente) {
12         this.coeficienteCuadratico = coeficienteCuadratico;
13         this.coeficienteLineal = coeficienteLineal;
14         this.coeficienteIndependiente = coeficienteIndependiente;
15     }
16
17     public double getCoeficienteCuadratico() {
18         return coeficienteCuadratico;
19     }
20
21     public void setCoeficienteCuadratico(double coeficienteCuadratico) {
22         this.coeficienteCuadratico = coeficienteCuadratico;
23     }
24
25     public double getCoeficienteLineal() {
26         return coeficienteLineal;
27     }
28
29     public void setCoeficienteLineal(double coeficienteLineal) {
30         this.coeficienteLineal = coeficienteLineal;
31     }
32
33     public double getCoeficienteIndependiente() {
34         return coeficienteIndependiente;
35     }
36
37     public void setCoeficienteIndependiente(double coeficienteIndependiente) {
38         this.coeficienteIndependiente = coeficienteIndependiente;
39     }
40
41     @Override
42     public String toString() {
43         return "Polinomio Interpolador de Lagrange: " +
44             coeficienteCuadratico + " x² + " +
45             coeficienteLineal + " x + " +
46             coeficienteIndependiente;
47     }
48 }
49
```

Clase Node.

```
1
2 package practica5_mn;
3
4 class Node {
5     private Object data;
6     private Node left;
7     private Node right;
8
9     //Constructor
10    public Node(Object data) {
11        this.data = data;
12    }
13
14    //Getters And Setters
15    public Object getData() {
16        return data;
17    }
18
19    public void setData(Object data) {
20        this.data = data;
21    }
22
23    public Node getLeft() {
24        return left;
25    }
26
27    public void setLeft(Node left) {
28        this.left = left;
29    }
30
31    public Node getRight() {
32        return right;
33    }
34
35    public void setRight(Node right) {
36        this.right = right;
37    }
38 }
```

Class NodeFunctionOfX

```
1 |
2 package practica5_mn;
3 @ public class NodeFunctionOfX extends Node {
4
5     public NodeFunctionOfX(FunctionOfX data) {
6         super(data);
7     }
8
9     @Override
10    public FunctionOfX getData() {
11        return (FunctionOfX) super.getData();
12    }
13 }
14
```

Class NodeDivDiffFunctionOfX

```
1 |
2 package practica5_mn;
3 public class NodeDivDiffFunctionOfX extends NodeFunctionOfX {
4
5     public NodeDivDiffFunctionOfX(DivDiffFunctionOfX data) {
6         super(data);
7     }
8
9     @Override
10    public DivDiffFunctionOfX getData() {
11        return (DivDiffFunctionOfX) super.getData();
12    }
13 }
14
```

Clase FunctionOfX

```
1 package practica5_mn;
2 @
3 public class FunctionOfX {
4
5     private String functionString;
6     private int valueOfX;
7     private String stringValueOfX;
8     private double valueOfFunction;
9
10    private int level;
11
12    public FunctionOfX(int valueOfX) {
13        this.setValueOfX(valueOfX);
14        this.setValueOfFunction(xSustitution(xValue: valueOfX));
15        this.setFunctionString(functionString: String.format(format: "f(%d)", args: this.valueOfX));
16        this.setStringValueOfX(stringValueOfX: String.valueOf(i: this.valueOfX()));
17        this.setLevel(level: 1);
18    }
19
20    public FunctionOfX(double valueOfFunction) {
21        this.setValueOfFunction(valueOfFunction);
22    }
23
24    private static double xSustitution(double xValue) {
25        double result = Math.pow(x: Math.E, ((3 * xValue) - 2));
26        return result;
27    }
28
29    public String getFunctionString() {
30        return this.functionString;
31    }
32
33    public void setFunctionString(String functionString) {
34        this.functionString = functionString;
35    }
36
37    public int getValueOfX() {
38        return this.valueOfX;
39    }
40
41    public void setValueOfX(int valueOfX) {
42        this.valueOfX = valueOfX;
43    }
44
45    public double getValueOfFunction() {
46        return this.valueOfFunction;
47    }
48
49    public void setValueOfFunction(double valueOfFunction) {
```

```

50 |         this.valueOfFunction = valueOfFunction;
51 |     }
52 |
53 | [-] public int getLevel() {
54 |     return level;
55 | }
56 |
57 | [-] public void setLevel(int level) {
58 |     this.level = level;
59 | }
60 |
61 | [-] public String getStringValueOfX() {
62 |     return stringValueOfX;
63 | }
64 |
65 | [-] public void setStringValueOfX(String stringValueOfX) {
66 |     this.stringValueOfX = stringValueOfX;
67 | }
68 | }
69 |

```

Clase DivDiffFunctionOfX

```

1 | package practica5_mn;
2 |
3 | public class DivDiffFunctionOfX extends FunctionOfX {
4 |
5 |
6 |     //Function of 2 Functions (1 Level above Function)
7 | [-] public DivDiffFunctionOfX(FunctionOfX valueXn, FunctionOfX valueXm) {
8 |     super(valueOfFunction: DivDiffFunctionOfX.dividedDifference(valueXn, valueXm));
9 |     this.setFunctionString(functionString: String.format(format: "f(%d,%d)", args: valueXn.getValueOfX(), args: valueXm.getValueOfX()));
10 |     this.setLevel(valueXn.getLevel() + 1);
11 | }
12 |
13 |     //Function of 2 DividedDiffFunctions (Upper Levels)
14 | [-] public DivDiffFunctionOfX(DivDiffFunctionOfX valueXn, DivDiffFunctionOfX valueXm) {
15 |     super(valueOfFunction: DivDiffFunctionOfX.dividedDifference(valueXn, valueXm));
16 |     this.setFunctionString(functionString: String.format(format: "f(%s,%s)", args: valueXn.getStringValueOfX(), args: valueXm.getStringValueOfX()));
17 |     this.setLevel(valueXn.getLevel() + 1);
18 | }
19 |
20 |     //Divided difference Method: f(Xi) - f(Xi+1) /
21 | [-] private static double dividedDifference(FunctionOfX valueXn, FunctionOfX valueXm) {
22 |
23 |     double functionOfXn_Xm;
24 |     functionOfXn_Xm = (valueXm.getValueOfFunction() - valueXn.getValueOfFunction()) / valueXn.getLevel();
25 |     return functionOfXn_Xm;
26 | }
27 |
28 |
29 | }
30 |

```

Clase Interpolación Lagrange

```
1 package practica5_mn;
2 public class InterpolacionLagrange {
3
4     private int[] valoresXi;
5     private int[] valoresFxi;
6
7     public InterpolacionLagrange(int[] valoresXi, int[] valoresFxi) {
8         this.valoresXi = valoresXi;
9         this.valoresFxi = valoresFxi;
10    }
11
12    public PolinomioCuadratico PolinomioInterpolador(int[] valoresX, int[] valoresFxi) {
13        double[] cuadratico = new double[3];
14        double[] lineal = new double[3];
15        double[] independiente = {1, 1, 1};
16        double[] denominadores = {1, 1, 1};
17
18        PolinomioCuadratico[] Polinomios = new PolinomioCuadratico[3];
19
20        PolinomioCuadratico PolinomioLagrange = new PolinomioCuadratico();
21
22        for (int i = 0; i < valoresX.length; i++) {
23            cuadratico[i] = 1;
24            for (int j = 0; j < valoresX.length; j++) {
25                if (i != j) {
26                    lineal[i] += (-1.0 * valoresX[j]);
27                    independiente[i] *= (-1.0 * valoresX[j]);
28                    denominadores[i] *= (valoresX[i] - valoresX[j]);
29                }
30            }
31            Polinomios[i] = new PolinomioCuadratico((cuadratico[i] / denominadores[i] * valoresFxi[i]), (lineal[i] / denominadores[i] * valoresFxi[i]), (independiente[i] / denominadores[i] * valoresFxi[i]));
32            PolinomioLagrange.setCoeficienteCuadratico(PolinomioLagrange.getCoeficienteCuadratico() + Polinomios[i].getCoeficienteCuadratico());
33            PolinomioLagrange.setCoeficienteLineal(PolinomioLagrange.getCoeficienteLineal() + Polinomios[i].getCoeficienteLineal());
34            PolinomioLagrange.setCoeficienteIndependiente(PolinomioLagrange.getCoeficienteIndependiente() + Polinomios[i].getCoeficienteIndependiente());
35        }
36        return PolinomioLagrange;
37    }
38
39    public int[] getValoresXi() {
40        return valoresXi;
41    }
42
43    public void setValoresXi(int[] valoresXi) {
44        this.valoresXi = valoresXi;
45    }
46
47    public int[] getValoresFxi() {
48        return valoresFxi;
49    }
50
51
52    public void setValoresFxi(int[] valoresFxi) {
53        this.valoresFxi = valoresFxi;
54    }
55
56
57 }
```

Clase Interpolación Newton

```
1 package practica5_mn;
2
3 public class InterpolacionNewton {
4
5     private NodeDivDiffFunctionOfX root;
6
7     private float xValue;
8
9     public InterpolacionNewton(NodeDivDiffFunctionOfX root) {
10         this.setRoot(root);
11         this.setXValue((this.getRoot().getData().getLevel() - 1) / 2);
12     }
13
14     public double Forward(double xValue) {
15         NodeFunctionOfX iNode = this.getRoot();
16         double result = 0;
17         while (iNode != null) {
18             result += (iNode.getData().getValueOfFunction()) * Math.pow(xValue, iNode.getData().getLevel() - 1);
19             iNode = (NodeFunctionOfX) iNode.getLeft();
20         }
21         return result;
22     }
23
24     public double Backward(double xValue) {
25         NodeFunctionOfX iNode = this.getRoot();
26         double result = 0;
27         while (iNode != null) {
28             result += iNode.getData().getValueOfFunction() * Math.pow(xValue, iNode.getData().getLevel() - 1);
29             iNode = (NodeFunctionOfX) iNode.getRight();
30         }
31         return result;
32     }
33
34     public NodeDivDiffFunctionOfX getRoot() {
35         return root;
36     }
37
38     public void setRoot(NodeDivDiffFunctionOfX root) {
39         this.root = root;
40     }
41
42     public float getXValue() {
43         return xValue;
44     }
45
46     public void setXValue(float xValue) {
47         this.xValue = xValue;
48     }
49
50     @Override
51     public String toString() {
52         return " InterpolacionNewton" +
53             "\n f (X0 , X1 , X2 , X3) = " + this.getRoot().getData().getValueOfFunction() +
54             "\nInterpolación " +
55             "\nHacia adelante: " + this.Forward(xValue: 1.5) +
56             "\nHacia atrás: " + this.Backward(xValue: 1.5);
57     }
58 }
59
```


Programa Interpolación y Ajuste de Funciones

Polinomio de Interpolación de Newton

Calcular

Polinomio de Interpolación de Lagrange

	X_i	$f(X_i)$
$i = 0$	<input type="text"/>	<input type="text"/>
$i = 1$	<input type="text"/>	<input type="text"/>
$i = 2$	<input type="text"/>	<input type="text"/>

Calcular

Una vez abierto el programa se muestra una interfaz gráfica la cual se divide en 2 partes, la mitad hacia arriba para el Polinomio de Interpolación de Newton y la mitad hacia abajo para el Polinomio de Interpolación de Lagrange.

Para calcular la ecuación dada por medio del polinomio de interpolación de Newton, basta con presionar el primer botón que dice “Calcular” y se mostrará el resultado de la ecuación resuelta por medio del método dicho.

Programa Interpolación y Ajuste de Funciones

Polinomio de Interpolación de Newton

Resultados de Polinomio de Interpolación de Newton

InterpolacionNewton
 $f(X_0, X_1, X_2, X_3) = 156.80970308852767$
 Interpolación
 Hacia adelante: 588.7015398917466
 Hacia atrás: 4302.842951659667

Aceptar

Polinomio de Interpolación de Lagrange

	X_i	$f(X_i)$
$i = 0$	<input type="text"/>	<input type="text"/>
$i = 1$	<input type="text"/>	<input type="text"/>
$i = 2$	<input type="text"/>	<input type="text"/>

Calcular

Posteriormente, pasamos a la parte del Polinomio de Interpolación de Lagrange. Para este, basta con ingresar los datos de X_0 , X_1 y X_2 , posteriormente se introducen los datos de las funciones para cada valor de X anteriormente ingresado.


Polinomio de Interpolación de Lagrange

	X_i	$f(X_i)$
$i = 0$	<input type="text" value="3"/>	<input type="text" value="4"/>
$i = 1$	<input type="text" value="4"/>	<input type="text" value="0"/>
$i = 2$	<input type="text" value="5"/>	<input type="text" value="1"/>

Calcular

Posteriormente, se presiona el botón que dice “Calcular”, este siendo el botón que está por debajo de todo, una vez se presione, mostrará el resultado.

Polinomio de Interpolación de Newton

 Resultados de Polinomio de Interpolación de Lagrange

Polinomio Interpolador de Lagrange: $2.5 x^2 + -21.5 x + 46.0$

Aceptar

Calcular

Polinomio de Interpolación de Lagrange

	X_i	$f(X_i)$
$i = 0$	3	4
$i = 1$	4	0
$i = 2$	5	1

Calcular