



# REPORTE – PROYECTO PRIMER PARCIAL-PAR 5

PROGRAMACIÓN ORIENTADA A OBJETOS  
II PAO 2023-2024

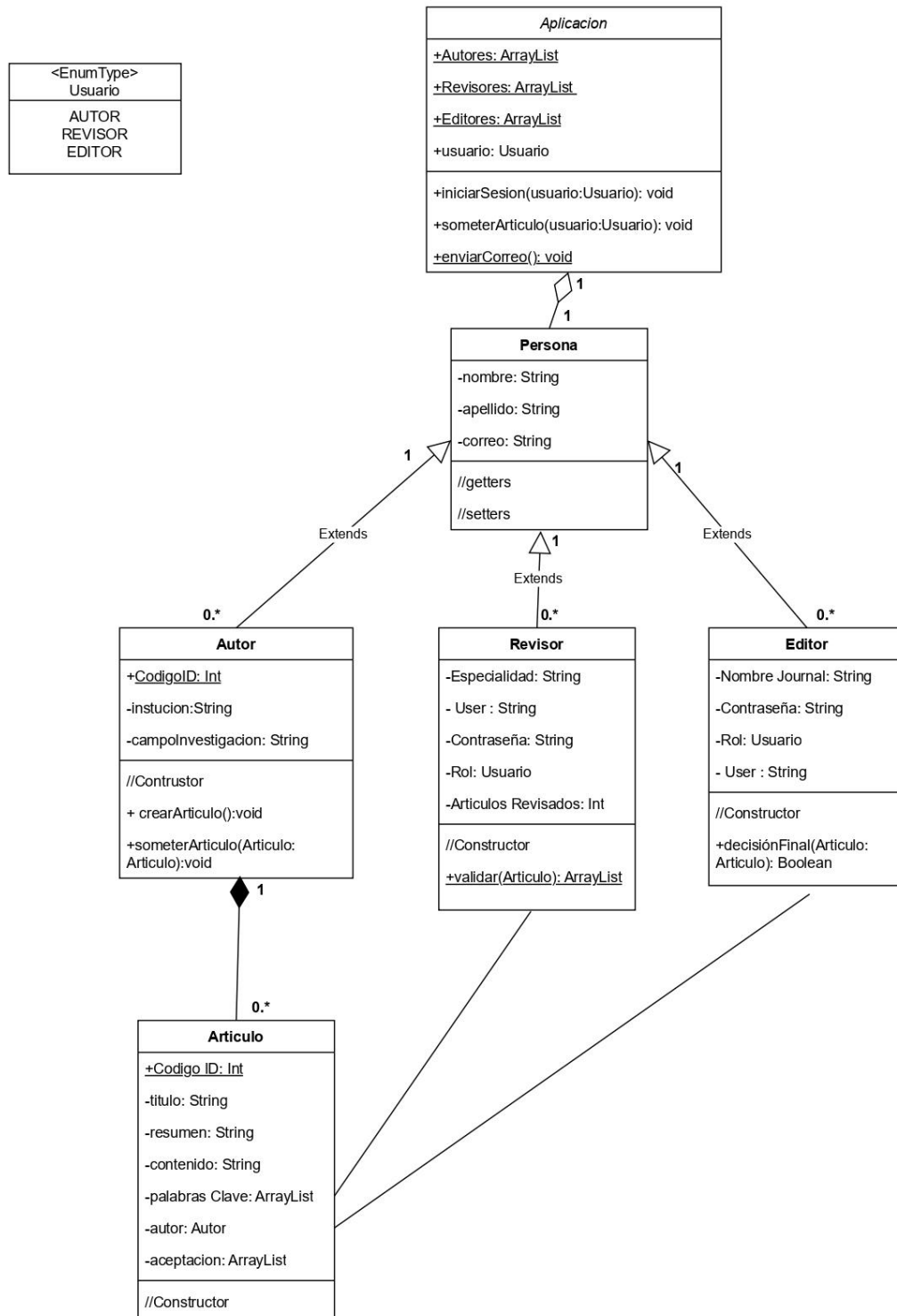
## Integrantes:

Valdiviezo Chancay Victor Manuel  
Desintonio Leon Jesus Alfonso  
Vega Jacome Mauro Fernando

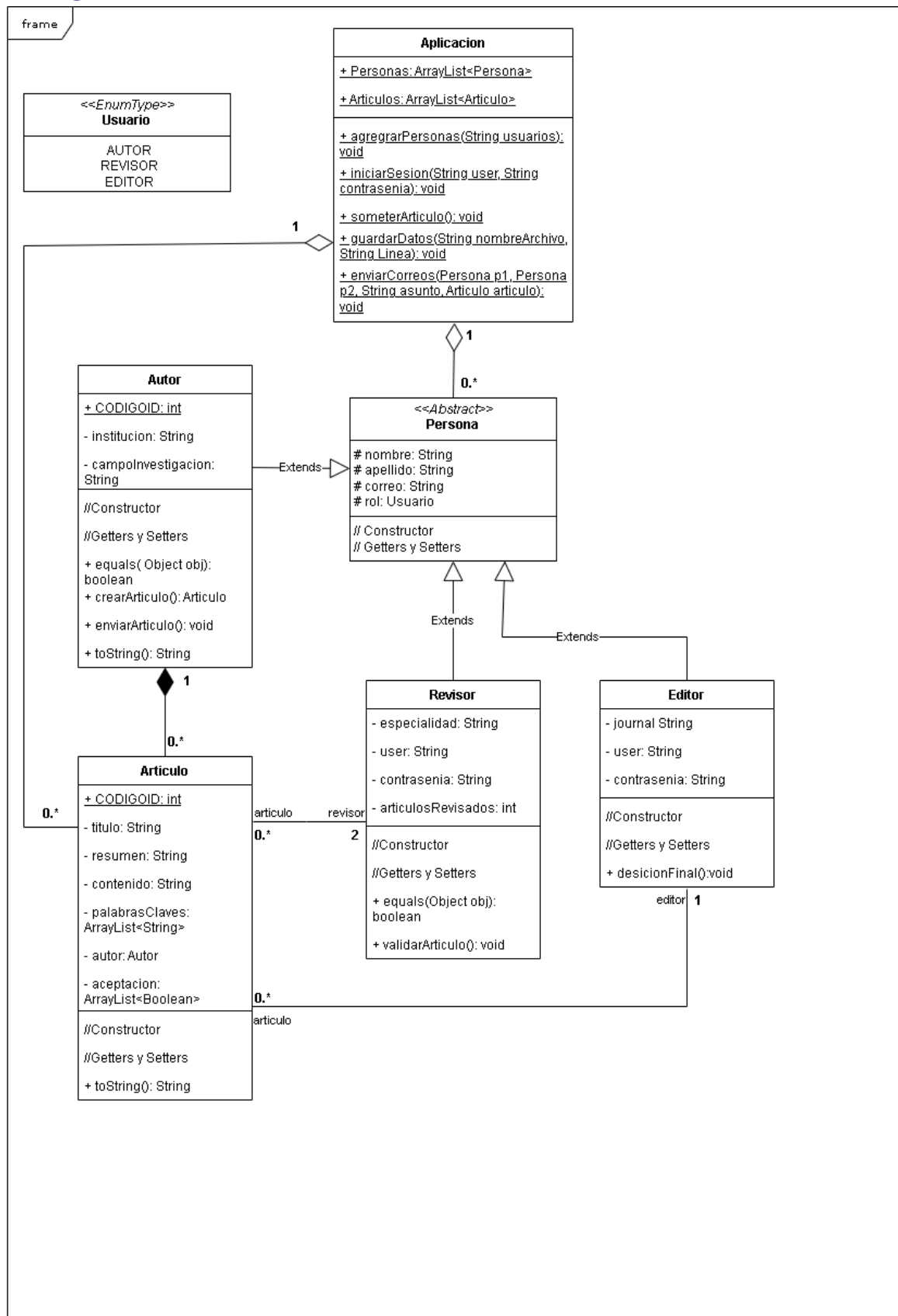
URL Repositorio:

Fecha: 07/07/2024

## 1. Diagrama de clases versión 1



## 2. Diagrama de clases versión 2



### 3. Tareas

En esta sección incluirán el detalle de las tareas que se asignó a cada estudiante.

Estudiante (Mauro Fernando Vega Jacome):

1. Enum usuario
2. Métodos y atributos Autor, Revisor y Editor

Estudiante (Victor Manuel Valdiviezo Chancay):

1. Clase Aplicacion
2. Clase Main
3. Metodo para enviar Correo

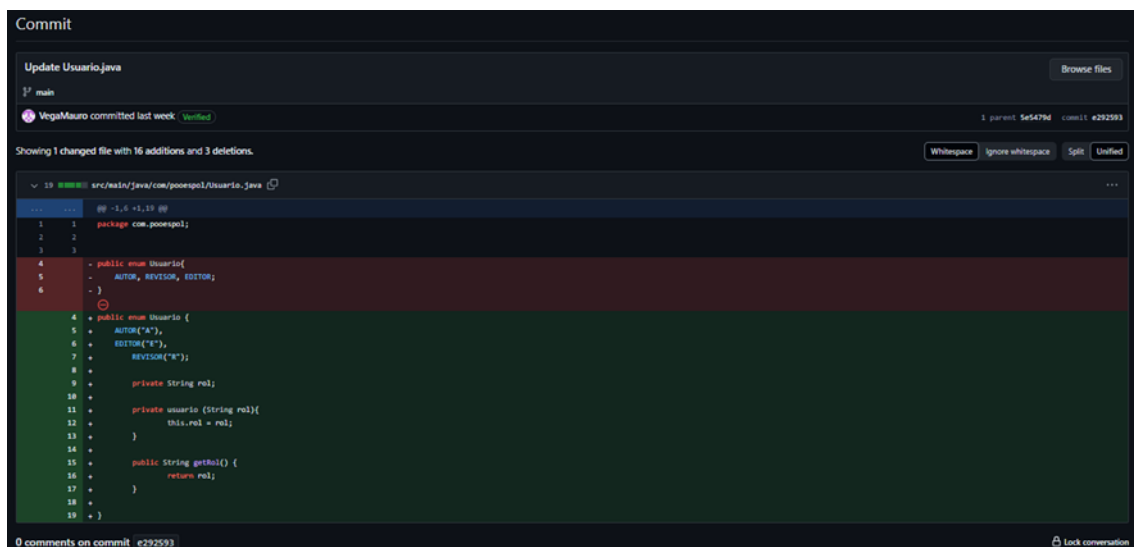
Estudiante (Jesus Alfonso Desintonio Leon):

1. Clase Artículo (toda)
2. Clase Autor (método crear articulo)

### 4. Evidencias de Tareas

Estudiante (Mauro Fernando Vega Jacome):

Commit 1



```
Commit

Update Usuario.java

main
VegaMauro committed last week Verified
1 parent 5e5479d commit e292593

Showing 1 changed file with 16 additions and 3 deletions.
Whitespace Ignore whitespace Split Unified

src/main/java/com/pooespol/Usuario.java
@@ -1,6 +1,19 @@
1 package com.pooespol;
2
3
4 - public enum Usuario{
5 -     AUTOR, REVISOR, EDITOR;
6 - }
7
8 + public enum Usuario {
9 +     AUTOR("A"),
10 +     EDITOR("E"),
11 +     REVISOR("R");
12 +
13 +     private String rol;
14 +
15 +     private usuario (String rol){
16 +         this.rol = rol;
17 +     }
18 +
19 +     public String getRol() {
20 +         return rol;
21 +     }
22 + }
```

Commit 2

Update Revisor.java

metodo validar articulo

Browse files

main

VegaMauro committed last week Verified

1 parent c417d87 commit dad289a

Showing 1 changed file with 7 additions and 7 deletions.

Whitespace Ignore whitespace Split Unified

src/main/java/com/posopol/Revisor.java

@@ -43,16 +43,16 @@ public void setuser(String user){  
43 this.user = user;  
44 }  
45  
46 - public ArrayList validarArticulo(Articulo articulo){  
46 + public ArrayList<Boolean> validarArticulo(Articulo articulo){  
47 Scanner sc = new Scanner(System.in);  
48 - ArrayList<String> validation = new ArrayList();  
48 + ArrayList<String> validation = new ArrayList();  
49 String respuesta = sc.nextLine();  
50 - if (respuesta.toLowerCase().equals("aceptado")) {  
51 - validation.add(respuesta);  
52 - } else if (respuesta.toLowerCase().equals("rechazado")) {  
53 - validation.add(respuesta);  
49 + if (respuesta.toLowerCase().equals("aprobado")) {  
50 + articulo.aceptacion.add(true);  
51 + } else if (respuesta.toLowerCase().equals("rechazado")) {  
52 + articulo.aceptacion.add(false);  
53 + }  
54 - return validation;  
54 + sc.close();  
55 + return articulo.aceptacion;  
56 }  
57  
58 }  
+  
0 comments on commit dad289a Lock conversation

## Commit 3

Update Editor.java

metodo decision final

Browse files

main

VegaMauro committed last week Verified

1 parent 2d83c77 commit c417d87

Showing 1 changed file with 18 additions and 0 deletions.

Whitespace Ignore whitespace Split Unified

src/main/java/com/posopol/Editor.java

@@ -35,4 +35,22 @@ public void setuser(String user) {  
35 public String getuser() {  
36 return user;  
37 }  
38 + public Boolean decisionFinal(Articulo articulo) {  
39 + Boolean decisionFinal=false;  
40 + ArrayList<Boolean> validation = articulo.getAceptacion();  
41 + if (validation.get(0) == true && validation.get(1) == true) {  
42 + decisionFinal = true;  
43 + } else if (validation.contains(false)) {  
44 + Scanner sc = new Scanner(System.in);  
45 + String respuesta = sc.nextLine();  
46 + if (respuesta.toLowerCase().equals("aprobado")) {  
47 + decisionFinal = true;  
48 + } else if (respuesta.toLowerCase().equals("rechazado")) {  
49 + decisionFinal = false;  
50 + }  
51 + sc.close();  
52 + }  
53 + return decisionFinal;  
54 + }  
55 + }  
38 56 }  
0 comments on commit c417d87 Lock conversation

Estudiante (Victor Manuel Valdiviezo Chancay):

Commit 1

Avance 4 Valdiviezo

main

Manuel0811 committed last week

1 parent 46e8ea1 commit 883899a

Showing 6 changed files with 46 additions and 18 deletions.

src/main/java/com/pooesp/

Application.java

Editor.java

target/classes/com/pooesp/

Application.class

Articulo.class

Editor.class

Usuario.class

@@ -13,6 +13,17 @@

13 13 public class Aplicacion {

14 14 public static ArrayList<Persona> personas;

15 15

16 + public static void main(String[] args){

17 + Scanner sc = new Scanner(System.in);

18 + //AgregarPersonas(usuario.txt);

19 + int opcion = 0;

20 + do{

21 + System.out.println("Someter Articulo: 1 \n Iniciar Sesión: 2 \n Salir: 3 \n Escriba la opcion que desea realizar:");

22 + opcion = sc.nextInt();

23 + sc.nextLine();

24 + switch (opcion) {

25 + case 1:

26 + someterArticulo();

27 + break;

28 + case 2:

29 + System.out.println("Ingrese su User: ");

30 + String user = sc.nextLine();

31 + System.out.println("Ingrese su contraseña: ");

32 + String contrasena = sc.nextLine();

33 + iniciarSesion(user, contrasena);

34 + break;

35 + default:

36 + System.out.println("Opcion Incorrecta")

37 + break;

38 + }

39 + }while(opcion!=3);

40 + }

41 + }

42 + }

Commit 2

Avance 5 Valdiviezo

main

Manuel0811 committed last week

1 parent 88f5ba1 commit c9d88ef

Showing 15 changed files with 168 additions and 87 deletions.

pom.xml

src/main/java/com/pooesp/

Aplicacion.java

Articulo.java

Autor.java

Editor.java

Persona.java

Revisor.java

Usuario.java

target/classes/com/pooesp/

Aplicacion.class

Articulo.class

Autor.class

Editor.class

Persona.class

Revisor.class

Usuario.class

@@ -13,4 +13,16 @@

13 13 <plugin>org.apache.maven.plugins:maven-compiler-plugin</plugin>

14 14 </properties>

15 15

16 + <dependencies>

17 + <dependency>

18 + <groupId>javax.mail</groupId>

19 + <artifactId>javax.mail-api</artifactId>

20 + <version>1.6.2</version>

21 + </dependency>

22 + <dependency>

23 + <groupId>com.sun.mail</groupId>

24 + <artifactId>javax.mail</artifactId>

25 + <version>1.6.2</version>

26 + </dependency>

27 + </dependencies>

28 </project>

@@ -9,6 +9,11 @@

9 9 import java.nio.charset.StandardCharsets;

10 10 import java.util.ArrayList;

11 11 import java.util.Scanner;

12 + import java.util.Properties;

13 + import javax.mail.\*;

14 + import javax.mail.internet.InternetAddress;

15 + import javax.mail.internet.MimeMessage;

16 +

17 17

## Commit 3

Avance 6 Valdiviezo

main

Manuel0811 committed last week

2 parents c968aef + 3b84bce commit 6eb4c38

Showing 13 changed files with 172 additions and 42 deletions.

Filter changed files

src/main/java/com/poospol

Aplicacion.java

Articulo.java

Autor.java

Editor.java

Main.java

Revisor.java

target/classes/com/poospol

Aplicacion\$1.class

Aplicacion.class

Articulo.class

Autor.class

Editor.class

Main.class

Revisor.class

src/main/java/com/poospol/Aplicacion.java

18 18 public class Aplicacion {

19 19 public static ArrayList<Personas> personas;

20 20

21 21 public static void main(String[] args){

22 22 Scanner sc = new Scanner(System.in);

23 23 //agregar personas(usar idio.txt);

24 24 int opcion = 0;

25 25 do{

26 26 System.out.println("Someter Articulo: 1 \n Iniciar Sesión: 2 \n Salir: 3 \n Escriba la opcion que desea realizar:");

27 27 opcion = sc.nextInt();

28 28 sc.nextLine();

29 29 switch (opcion) {

30 30 case 1:

31 31 someterArticulo();

32 32 break;

33 33 case 2:

34 34 System.out.println("Ingrese su User: ");

35 35 String user = sc.nextLine();

36 36 System.out.println("Ingrese su contraseña: ");

37 37 String contraseña = sc.nextLine();

38 38 iniciarSesion(user, contraseña);

39 39 break;

40 40 default:

41 41 System.out.println("Opcion Incorrecta");

42 42 break;

43 43 }

44 44 }while(opcion!=2);

45 45 }

46 46 }

47 47 }

21 21

Estudiante (Jesus Alfonso Desintonio Leon):

## Commit 1

Update Articulo.java

main

JesusDeLe05 committed last week Verified

1 parent e292593 commit 46e8ea1

Showing 1 changed file with 18 additions and 2 deletions.

Filter changed files

src/main/java/com/poospol/Articulo.java

1 1 package com.poospol;

2 2 import java.util.ArrayList;

3 3

4 4 public class Articulo {

5 5

6 6 public int codigoId;

7 7 private String titulo;

8 8 private String resumen;

9 9 private String contenido;

10 10 private ArrayList<String> palabrasClaves;

11 11 private Autor autor;

12 12 private ArrayList<boolean> aceptacion;

13 13

14 14 public Articulo(int codigoId, String titulo, String resumen, String contenido, ArrayList<String> palabrasClaves, Autor autor, ArrayList<boolean> aceptacion) {

15 15 this.codigoId = codigoId;

16 16 this.titulo = titulo;

17 17 this.resumen = resumen;

18 18 this.contenido = contenido;

19 19 this.palabrasClaves = palabrasClaves;

20 20 this.autor = autor;

21 21 this.aceptacion = aceptacion;

22 22 }

0 comments on commit 46e8ea1

Lock conversation

## Commit 2

Update Autor.java

main

JesusDeleOS committed last week

1 parent e5164f5 commit 2db3c77

Showing 1 changed file with 26 additions and 1 deletion.

Whitespace Ignore whitespace Split Unified

src/main/java/com/poespol/Autor.java

```
42 42 }
43 43 }
44 44
45 - public Artículo crearArticulo(){
45 + public Artículo crearArticulo(Autor autor){
46 +
47 +     System.out.println("Ingrese el Código ID");
48 +     int codId = sc.nextInt();
49 +
50 +     System.out.println("Ingrese el Título");
51 +     String titulo = sc.nextLine();
52 +
53 +     System.out.println("Ingrese el Resumen");
54 +     String res = sc.nextLine();
55 +
56 +     System.out.println("Ingrese el contenido");
57 +     String cont = sc.nextLine();
58 +
59 +     System.out.println("Ingrese cuantas palabras clave va a insertar");
60 +     int nPalabras = sc.nextInt();
61 +     ArrayList<String> palClaves = new ArrayList<String>();
62 +     for(int i=1;i<=nPalabras;i++){
63 +         System.out.println("Ingrese la palabra clave N° "+i);
64 +         String pal = sc.nextLine();
65 +         palClaves.add(pal);
66 +     }
67 +
68 +     Artículo art = new Artículo(codId,titulo,res,cont,palClaves,autor);
69 +
70 +     return art;
71 + }
```

## Commit 3

Commit

Update Artículo.java

main

JesusDeleOS committed last week

1 parent edc88e1 commit e5164f5

Showing 1 changed file with 1 addition and 1 deletion.

Whitespace Ignore whitespace Split Unified

src/main/java/com/poespol/Articulo.java

```
1 1 @@ -2,7 +2,7 @@
2 2 import java.util.ArrayList;
3 3
4 - public class Artículo {
4 + public class Artículo {
5 -     public static int codigoId;
5 +     public int codigoId;
6 6     private String titulo;
7 7     private String resumen;
8 8     private String contenido;
```



## 5. Identificación de pilares de la programación orientada a objetos.

### Abstracción

```
public abstract class Persona {  
    protected String nombre;  
    protected String apellido;  
    protected String correo;  
    protected Usuario rol;  
  
    public Persona(String nombre, String apellido, String correo, Usuario rol){  
        this.nombre = nombre;  
        this.apellido = apellido;  
        this.correo = correo;  
        this.rol = rol;  
    }  
}
```

Sera una clase padre cuyos métodos y variables de instancia serán utilizados por las clases hijas: autor, revisor y editor. Por esta razón se la declaró una clase abstracta

### Encapsulamiento

```
3 public abstract class Persona {  
4     protected String nombre;  
5     protected String apellido;  
6     protected String correo;  
7     protected Usuario rol;  
  
7 public class Autor extends Persona{  
8     public static int CODIGOID;  
9     private String institucion;  
10    private String campoInvestigacion;  
11  
7 public class Revisor extends Persona{  
8     private String especialidad;  
9     private String user;  
10    private String contrasenia;  
11    private int articulosRevisados;  
12  
4 public class Artículo {  
5     public static int CODIGOID;  
6     private String titulo;  
7     private String resumen;  
8     private String contenido;  
9     private ArrayList<String> palabrasClaves;  
10    private Autor autor;  
11    private ArrayList<Boolean> aceptacion;  
12
```

```
6 public class Editor extends Persona{
7     private String journal;
8     private String contrasenia;
9     private String user;
```

```
18 public class Aplicacion {
19     public static ArrayList<Persona> personas;
20     public static ArrayList<Articulo> articulos;
21 }
```

```
16 /**
17  * Este metodo permite mirar el Nombre de la Persona
18  * @return Retorna un String del Nombre de la Persona
19  */
20 public String getNombre(){
21     return nombre;
22 }
23
24 /**
25  * Este metodo permite mirar el Apellido de la Persona
26  * @return Retornara un String del Apellido de la Persona
27  */
28 public String getApellido(){
29     return apellido;
30 }
31
32 /**
33  * Este metodo permite mirar el Correo de la Persona
34  * @return Retornara el String del Correo de la Persona
35  */
36 public String getCorreo(){
37     return correo;
38 }
```

```
40 /**
41  * Este metodo permite mirar el Rol de la Persona
42  * @return Retornara el Usuario del Rol de la Persona
43  */
44 public Usuario getRol(){
45     return rol;
46 }
47
48 /**
49  * Este metodo permite modificar el Nombre de la Persona
50  * @param nombre recibe el nuevo Nombre de la Persona
51  */
52 public void setNombre(String nombre){
53     this.nombre = nombre;
54 }
55
56 /**
57  * Este metodo permite modificar el Apellido de la Persona
58  * @param apellido recibe el nuevo Apellido de la Persona
59  */
60 public void setApellido(String apellido){
61     this.apellido = apellido;
62 }
```

```
64 /**
65  * Este metodo permite modificar el Correo de la Persona
66  * @param correo recibe el nuevo Correo de la Persona
67  */
68 public void setCorreo(String correo){
69     this.correo = correo;
70 }
71
72 /**
73  * Este metodo permite modificar el Rol de la Persona
74  * @param rol recibe el nuevo Rol de la Persona
75  */
76 public void setRol(Usuario rol){
77     this.rol = rol;
78 }
```

```
19      /**
20       * Este metodo permite mirar la Institucion del Autor
21       * @return Sera el nombre de la Institucion del Autor
22       */
23      public String getInstitucion(){
24          return institucion;
25      }
26  
```

```
27      /**
28       * Este metodo permite mirar el Campo de Investigacion del Autor
29       * @return Sera el nombre del Campo de Investigacion del Autor
30       */
31      public String getCampoInvestigacion(){
32          return campoInvestigacion;
33      }
34  
```

```
35      /**
36       * Este metodo permite modificar la Institucion del Autor
37       * @param institucion Sera la nueva Institucion del Autor
38       */
39      public void setInstitucion(String institucion){
40          this.institucion = institucion;
41      }
42
43      /**
44       * Este metodo permite modificar el Campo de Investigacion del Autor
45       * @param campoInvestigacion Sera el nuevo Campo de Investigacion del Autor
46       */
47      public void setCampoInvestigacion(String campoInvestigacion){
48          this.campoInvestigacion= campoInvestigacion;
49      }

```

```
22      /**
23       * Este metodo retornara la Especialidad del Revisor
24       * @return Retorna el String de la Especialidad del Editor
25       */
26      public String getEspecialidad(){
27          return especialidad;
28      }
29
30      /**
31       * Este metodo retornara la Contraseña del Revisor
32       * @return Retornara un String de la Contraseña del Editor
33       */
34      public String getContrasenia(){
35          return contrasenia;
36      }
37
38      /**
39       * Este metodo retornara el User del Revisor
40       * @return Retornara un String del User del Editor
41       */
42      public String getUser(){
43          return user;
44      }

```

```

46  /**
47   * Este metodo retornara la cantidad de Articulos Revisados del Revisor
48   * @return Retornara un int de la cantidad de Articulos Revisados
49   */
50  public int getArticulosRevisados(){
51      return articulosRevisados;
52  }
53
54  /**
55   * Este metodo permite modificar la Especialidad del Revisor
56   * @param especialidad Sera la nueva especialidad del Revisor
57   */
58  public void setEspecialidad(String especialidad){
59      this.especialidad = especialidad;
60  }
61
62  /**
63   * Este metodo permite modificar la Contraseña del Revisor
64   * @param contrasenia Sera la nueva Contraseña del Revisor
65   */
66  public void setContrasenia(String contrasenia){
67      this.contrasenia = contrasenia;
68  }

```

```

70  /**
71   * Este metodo permite modificar el User del Revisor
72   * @param user Sera el nuevo User del Revisor
73   */
74  public void setUser(String user){
75      this.user = user;
76  }
77

```

```

23  /**
24   * Este metodo permite acceder al Titulo del Articulo
25   * @return retorna un String con el Titulo del Articulo
26   */
27  public String getTitulo() {
28      return titulo;
29  }
30
31  /**
32   * Este metodo permite modificar el Titulo del Articulo
33   * @param titulo recibe el nuevo Titulo del Articulo
34   */
35  public void setTitulo(String titulo) {
36      this.titulo = titulo;
37  }
38
39  /**
40   * Este metodo permite accede al Resumen del Articulo
41   * @return retornara un String con el Resumen del Articulo
42   */
43  public String getResumen() {
44      return resumen;
45  }

```

```

47  /**
48   * Este metodo permite modificar el Resumen del Articulo
49   * @param resumen recibe el nuevo Resumen del Articulo
50   */
51  public void setResumen(String resumen) {
52      this.resumen = resumen;
53  }
54
55  /**
56   * Este metodo permite acceder al Contenido del Articulo
57   * @return retornara un String con el Contenido del Articulo
58   */
59  public String getContenido() {
60      return contenido;
61  }
62
63  /**
64   * Este metodo permite modificar el Contenido del Articulo
65   * @param contenido recibe el nuevo Contenido del Articulo
66   */
67  public void setContenido(String contenido) {
68      this.contenido = contenido;
69  }
70

```

```

72     /**
73      * Este metodo permite acceder a la lista de las Palabras Claves del Articulo
74      * @return retornara la lista de las Palabras Claves del Articulo
75      */
76     public ArrayList<String> getPalabrasClave() {
77         return palabrasClaves;
78     }
79
80     /**
81      * Este metodo permite modificar la lista de las Palabras Claves del Articulo
82      * @param palabrasClaves recibe la nueva lista de Palabras Claves del Articulo
83      */
84     public void setPalabrasClave(ArrayList<String> palabrasClaves) {
85         this.palabrasClaves = palabrasClaves;
86     }
87
88     /**
89      * Este metodo permite acceder al Autor del Articulo
90      * @return retorna el Autor del Articulo
91      */
92     public Autor getAutor() {
93         return autor;
94     }
95
96     /**
97      * Este metodo permite modificar el Autor del Articulo
98      * @param autor recibe el nuevo Autor del Articulo
99      */
100    public void setAutor(Autor autor) {
101        this.autor = autor;
102    }
103
104    /**
105     * Este metodo permite acceder a la lista de Aceptacion de los Revisores sobre el Articulo
106     * @return retorna la lista de Aceptacion del Articulo
107     */
108    public ArrayList<Boolean> getAceptacion(){
109        return aceptacion;
110    }
111
112    /**
113     * Este metodo permite modificar la lista de Aceptacion del Articulo
114     * @param aceptacion recibe la lista de Aceptacion del Articulo
115     */
116    public void setAceptacion(ArrayList<Boolean> aceptacion){
117        this.aceptacion = aceptacion;
118    }

```

En estos casos Encapsulamos estos atributos que se utilizaran en la aplicación, también en los métodos Getters y Setters, ya que a través de ellos accedemos a los atributos.

## Herencia

```
7 public class Autor extends Persona{
8     public static int CODIGOID;
9     private String institucion;
10    private String campoInvestigacion;
11
12    public Autor(String nombre, String apellido, String correo, Usuario rol, String institucion, String campoInvestigacion){
13        super(nombre,apellido,correo,rol);
14        this.institucion = institucion;
15        this.campoInvestigacion = campoInvestigacion;
16        CODIGOID++;
17    }
18 }
19
20 public class Revisor extends Persona{
21     private String especialidad;
22     private String user;
23     private String contrasenia;
24     private int articulosRevisados;
25
26     public Revisor(String nombre, String apellido, String correo, Usuario rol, String especialidad, String user, String contrasenia){
27         super(nombre, apellido, correo, rol);
28         this.especialidad = especialidad;
29         this.user = user;
30         this.contrasenia = contrasenia;
31         this.articulosRevisados = articulosRevisados;
32     }
33 }
34
35 public class Editor extends Persona{
36     private String journal;
37     private String contrasenia;
38     private String user;
39
40     public Editor(String nombre, String apellido, String correo, String journal, String contrasena, String user, Usuario rol) {
41         super(nombre,apellido,correo,rol);
42         this.journal = journal;
43         this.contrasenia = contrasena;
44         this.user = user;
45     }
46 }
47 }
```

Todas estas son clases que heredan de la clase abstracta persona, ya que necesitan y hacen uso de sus variables y métodos.

## Polimorfismo

```
124 public String toString(){
125     String palabras = "";
126     for(String e: palabrasClaves){
127         palabras+= " "+e;
128     }
129     String s= titulo+ "-"+resumen+"-"+contenido+"-"+palabras+"-"+autor.toString();
130     return s;
131 }
132
133 /**
134  * Este metodo permite comparar un Artículo con otro objeto
135  * @param obj recibe un objeto con el se comparara
136  * @return retorna true si el Artículo y el objeto son de la misma clase y sus atributos son iguales
137  */
138 @Override
139 public boolean equals(Object obj){
140     if (this == obj)
141         return true;
142     if (obj == null)
143         return false;
144     if (this.getClass() == obj.getClass()){
145         Artículo articulo2 = (Artículo)obj;
146         if((titulo.equals(articulo2.getTitulo())) && (resumen.equals(articulo2.getResumen())) && (contenido.equals(articulo2.getContenido())) && (palabrasClaves.equals(articulo2.getPalabrasClaves()))){
147             return true;
148         }
149     }
150     return false;
151 }
```

En este caso se hace uso del polimorfismo de inclusión ya que es necesario sobrescribir los métodos de *toString* y *equals* para el uso propio de cada clase según lo necesite.

```

38 String linea;
39 while ((linea = br.readLine()) != null){
40     String[] lin = linea.split(",");
41     String nombre = lin[0], apellido = lin[1], correo = lin[2], rol = lin[3];
42     if (rol.equals("AUTOR")){
43         Usuario rol1 = Usuario.valueOf(rol);
44         String institucion = lin[4], campoInvestigacion = lin[5];
45         Autor autor = new Autor(nombre, apellido, correo, rol1, institucion, campoInvestigacion);
46         personas.add(autor);
47     } else if (rol.equals("REVISOR")){
48         Usuario rol1 = Usuario.valueOf(rol);
49         String especialidad = lin[4], user=lin[5], contrasenia= lin[6], articulosRevisados =lin[7];
50         int articulosR= Integer.parseInt(articulosRevisados);
51         Revisor revisor = new Revisor(nombre, apellido, correo, rol1, especialidad, user, contrasenia, articulosR);
52         personas.add(revisor);
53     } else if (rol.equals("EDITOR")){
54         Usuario rol1 = Usuario.valueOf(rol);
55         String nombreJournal = lin[4], user = lin[5], contrasenia = lin[6];
56         Editor editor = new Editor(nombre, apellido, correo, nombreJournal, contrasenia, user, rol1);
57         personas.add(editor);
58     }
59 }

```

En este caso es Polimorfismo Puro, ya que se están guardando Autores, Revisores y Editores en una lista de Personas.

## 6. Identificación de otros conceptos

### ArrayLists

```

1 package com.pooespol;
2 import java.util.ArrayList;
3
4 public class Artículo {
5     public int codigoId;
6     private String titulo;
7     private String resumen;
8     private String contenido;
9     private ArrayList<String> palabrasClaves;
10    private Autor autor;
11    private ArrayList<Boolean> aceptacion;
12
13    public Artículo(String titulo, String resumen, String contenido, ArrayList<String> palabrasClaves, Autor autor) {
14        this.titulo = titulo;
15        this.resumen = resumen;
16        this.contenido = contenido;
17        this.palabrasClaves = palabrasClaves;
18        this.autor = autor;
19        aceptacion = new ArrayList<>();
20        codigoId ++;
21    }

```

En la clase Artículo se utilizó dos ArrayList para las variables de instancia: “palabrasClaves” y “aceptacion”.

```

73 public Artículo crearArticulo(){
74     Scanner sc = new Scanner(System.in);
75     System.out.println("Ingresa el título del artículo: ");
76     String titulo = sc.nextLine();
77     System.out.println("Ingresa un resumen del artículo: ");
78     String resumen = sc.nextLine();
79     System.out.println("Ingresa el contenido del artículo: ");
80     String contenido = sc.nextLine();
81     System.out.println("Ingresa cuantas palabras claves del artículo vas a agregar: ");
82     int cantidad = sc.nextInt();
83     sc.nextLine();
84     ArrayList<String> palabrasClaves = new ArrayList<>();
85     for(int i =0; i<cantidad;i++){
86         System.out.println("Ingresa la palabra clave del artículo: ");
87         String palabra = sc.nextLine();
88         palabrasClaves.add(palabra);
89     }
90     Artículo art = new Artículo(titulo, resumen, contenido, palabrasClaves, this);
91
92     String linea = art.toString();
93     Aplicacion.guardarDatos("articulos", linea);
94     Aplicacion.articulos = new ArrayList<>();
95     Aplicacion.articulos.add(art);
96     return art;
97 }
98
99

```

En la clase Autor, en el método crearArticulo(), se usó un ArrayList para guardar las palabras claves insertadas por el usuario

```

18 public class Aplicacion {
19     public static ArrayList<Persona> personas;
20     public static ArrayList<Articulo> articulos;
21
22
23     /**
24      * Este metodo agrega las Personas de un .txt a un ArrayList
25      * @param usuarios un .txt con todas las personas que acceden al programa
26      */
27     public static void agregarPersonas(String usuarios){
28         personas = new ArrayList<>();
29         File archivo = null;
30         FileReader fr = null;
31         BufferedReader br = null;

```

En la clase Aplicación se usó ArrayList para las variables de instancia "personas" y "artículos"

## Creación de objetos a partir de datos de archivo

```

33 try {
34     archivo = new File(usuarios);
35     fr = new FileReader(archivo, StandardCharsets.UTF_8);
36     br = new BufferedReader(fr);
37
38     String linea;
39     while ((linea = br.readLine()) != null){
40         String[] lin = linea.split(",");
41         String nombre = lin[0], apellido = lin[1], correo = lin[2], rol = lin[3];
42         if (rol.equals("AUTOR")){
43             Usuario rol1 = Usuario.valueOf(rol);
44             String institucion = lin[4], campoInvestigacion = lin[5];
45             Autor autor = new Autor(nombre, apellido, correo, rol1, institucion, campoInvestigacion);
46             personas.add(autor);
47         } else if (rol.equals("REVISOR")){
48             Usuario rol1 = Usuario.valueOf(rol);
49             String especialidad = lin[4], user=lin[5], contrasenia= lin[6], articulosRevisados =lin[7];
50             int articulosR= Integer.parseInt(articulosRevisados);
51             Revisor revisor = new Revisor(nombre, apellido, correo, rol1, especialidad, user, contrasenia,articulosR);
52             personas.add(revisor);
53         }else if(rol.equals("EDITOR")){
54             Usuario rol1 = Usuario.valueOf(rol);
55             String nombreJournal = lin[4], user = lin[5], contrasenia = lin[6];
56             Editor editor = new Editor(nombre, apellido, correo, nombreJournal, contrasenia, user, rol1);

```

Se leen los datos del archivo para identificar si se trata de un autor, revisor o editor y posteriormente crear un objeto dependiendo de que tipo sea y agregarlos a una lista



## Sobrescritura

```

124  public String toString(){
125      String palabras = "";
126      for(int i =0; i<palabrasClaves.size();i++){
127          palabras+= "\nPalabra "+(i+1)+": "+palabrasClaves.get(i);
128      }
129      String s= "Titulo: "+titulo+ "\nResumen: "+resumen+"\nContenido: "+contenido+palabras+"\nAutor:"+autor.toString();
130      return s;
131  }
132
133  /**
134   * Este metodo permite comparar un Articulo con otro objeto
135   * @param obj recibe un objeto con el se comparara
136   * @return retorna true si el Articulo y el objeto son de la misma clase y sus atributos son iguales
137   */
138  @Override
139  public boolean equals(Object obj){
140      if (this == obj)
141          return true;
142      if (obj == null)
143          return false;
144      if (this.getClass() == obj.getClass()){
145          Articulo articulo2 = (Articulo)obj;
146          if((titulo.equals(articulo2.getTitulo())) && (resumen.equals(articulo2.getResumen())) && (contenido.equals(articulo2.getContenido())))
147              return true;
148      }
149      return false;
150  }

```

En el programa solamente fue necesario sobrescribir los métodos *toString* y *equals* los cuales se heredan implícitamente en todas las clases.

## 7. Programa en ejecución

[illegible]

```
////////////////////////////////////
Vamos a registrar su articulo
Ingrese el titulo del articulo:
La IA
Ingrese un resumen del articulo:
La inteligencia artificial es muy util...
Ingrese el contenido del articulo:
Que es la IA
Ingrese cuantas palabras claves del articulo va a agregar:
2
Ingre la palabra clave del articulo:
Inteligencia
Ingre la palabra clave del articulo:
Artificial
Articulo creado
Ahora se enviara a dos revisores para su aceptacion
Se envio su articulo exitosamente
////////////////////////////////////
Bienvenido a la Aplicacion para publicacion de articulos
Escriba el numero de la accion que desea realizar
Someter Articulo: 1
Iniciar Sesion: 2
Salir: 3
Escriba la opcion que desea realizar:
2
Ingrese su User:
plop
```

```
Ingrese su User:
plop
Ingrese su contraseña:
lopez12
////////////////////////////////////
Revision de articulo
Articulo por designar
Titulo: La IA
Resumen: La inteligencia artificial es muy util...
Contenido: Que es la IA
Palabra 1: Inteligencia
Palabra 2: Artificial
Autor:
Nombre: Juan
Apellido: Valdez
Correo: juanv@gmail.com
Institucion: Espol
Campo de Invesigacion: Matematicas
////////////////////////////////////
Desea asignar el articulo a un Editor
1. Aceptar
2. Rechazar
Escriba su opcion:
1
Asignacion Aceptada
Hemos enviado el articulo al Editor
Siguiendo Articulo por asignar
```



```
////////////////////////////////////  
Usuario o contraseña invalida  
////////////////////////////////////  
Bienvenido a la Aplicacion para publicacion de articulos  
Escriba el numero de la accion que desea realizar  
Someter Articulo: 1  
Iniciar Sesion: 2  
Salir: 3  
Escriba la opcion que desea realizar:  
3  
Saliendo de la Aplicacion  
PS C:\Users\Manuel\Documents\POO\poo5_1p_valdiviezo_desintonio_vega> █
```

## 8. JAVADOC

Agregar la documentación JAVADOC como una carpeta adicional a este reporte. Los métodos deben estar siempre comentados con el formato explicado en clase.

Referencia:<https://www.youtube.com/watch?v=KChdcRscFt0>