

# TAREA PARA ED04

---

**Nombre:** Ayoze Pestano de la Rosa

**Curso:** 1 de Ciclo Superior de Desarrollo de Aplicaciones Web a distancia

## ÍNDICE

- Introducción
  - Objetivos
  - Material empleado
  - Desarrollo
  - Conclusiones
- 

## **INTRODUCCIÓN:**

El presente trabajo tiene como objetivo describir el proceso de desarrollo de software utilizando herramientas como Netbeans, Github y Git. Estas herramientas son ampliamente utilizadas en el ámbito de la programación para facilitar la colaboración entre desarrolladores y mejorar la calidad del código generado.

## **OBJETIVOS:**

El principal objetivo de este trabajo es proporcionar una guía completa sobre cómo utilizar estas herramientas de forma efectiva para el desarrollo de software en equipo. También se busca destacar las ventajas de utilizar estas herramientas y cómo pueden mejorar la eficiencia en el desarrollo de proyectos de software.

## **MATERIAL EMPLEADO:**

Para el desarrollo de software utilizando estas herramientas, es necesario tener acceso a una computadora con el software de Netbeans instalado. También es necesario contar con una cuenta de Github para poder compartir el código generado y utilizar el sistema de control de versiones de Git para administrar los cambios en el código.

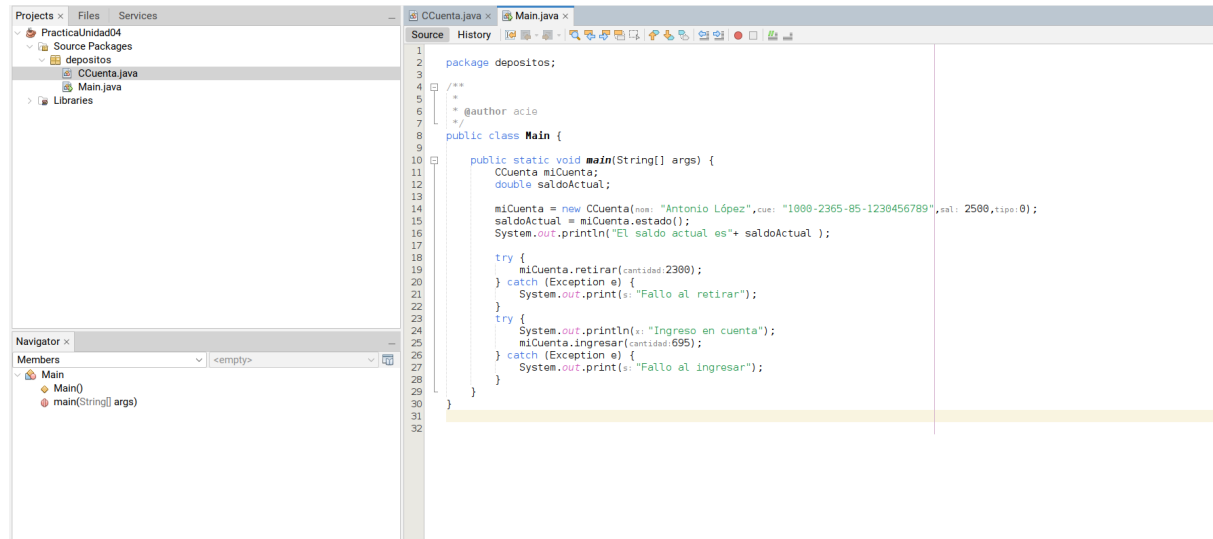
## **DESARROLLO:**

El proceso de desarrollo de software utilizando Netbeans, Github y Git comienza con la creación de un repositorio en Github, donde se almacenará todo el código generado. A continuación, se configura Netbeans para conectarse con Github y se clona el repositorio en la computadora local. A medida que se va escribiendo el código, se utiliza Git para hacer cambios y realizar commits, lo que permite registrar los cambios realizados en el código.

Una vez que se ha escrito el código y se ha realizado el commit correspondiente, se puede subir el código al repositorio de Github. De esta manera, el código queda disponible para que otros desarrolladores puedan acceder a él y realizar cambios o mejoras.

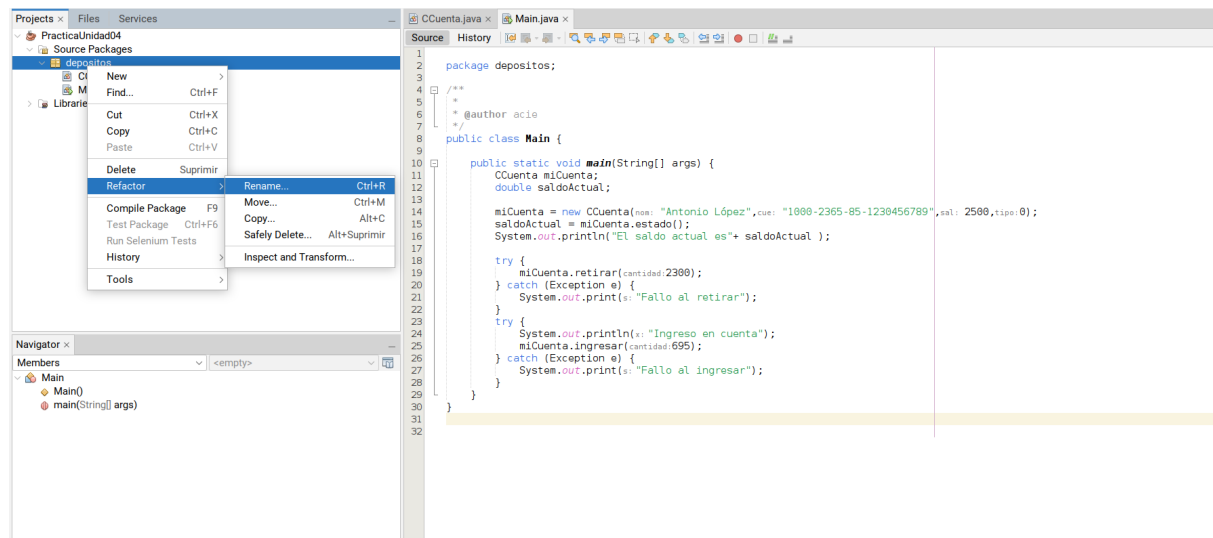
# TAREA PARA ED04

Primero Importamos el proyecto a nuestro IDE Netbeans:

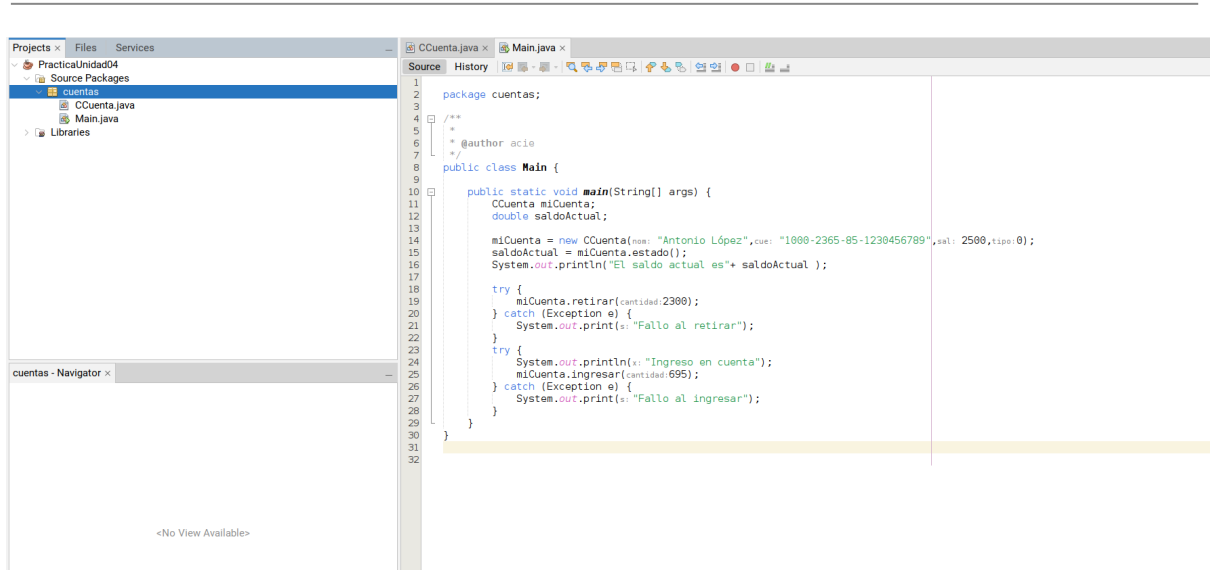


## REFACTORIZACIÓN

1. Las clases deberán de formar parte del paquete cuentas:  
Para ello, modificamos el paquete “depósitos” a “cuentas” con el refactor:



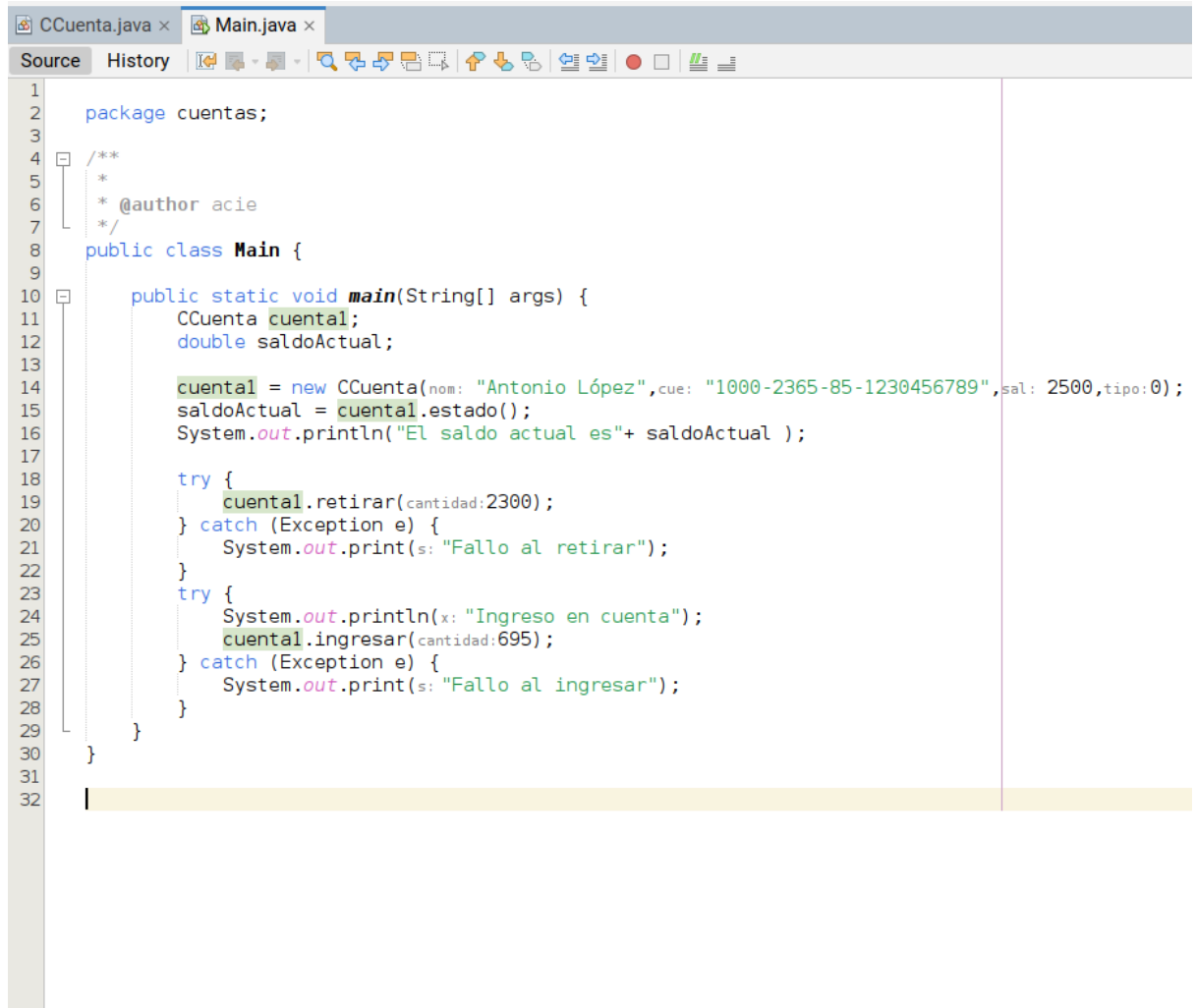
# TAREA PARA ED04



## 2. Cambiar el nombre de la variable "miCuenta" por "cuenta1".

Para ello, modificamos la variable y los métodos donde esté siendo usada y/o llamada:

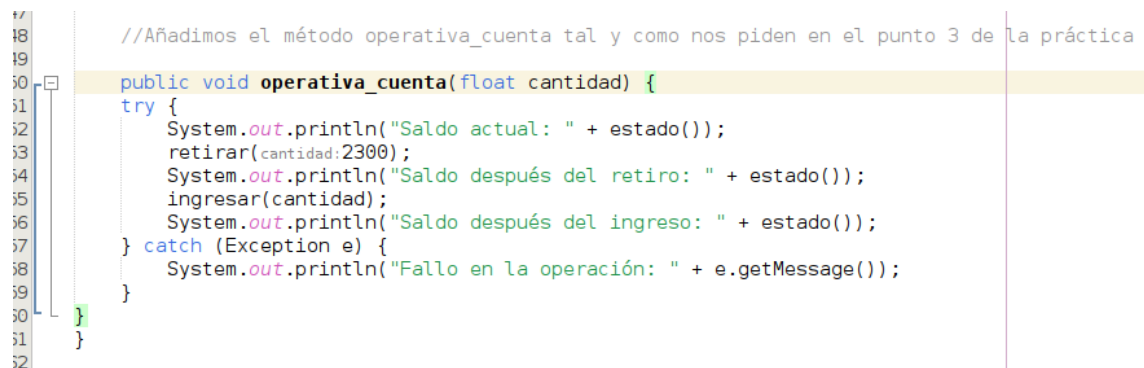
# TAREA PARA ED04



```
1 package cuentas;
2
3
4 /**
5  *
6  * @author acie
7  */
8 public class Main {
9
10     public static void main(String[] args) {
11         CCuenta cuenta1;
12         double saldoActual;
13
14         cuenta1 = new CCuenta(nom: "Antonio López", cue: "1000-2365-85-1230456789", sal: 2500, tipo: 0);
15         saldoActual = cuenta1.estado();
16         System.out.println("El saldo actual es"+ saldoActual );
17
18         try {
19             cuenta1.retirar(cantidad:2300);
20         } catch (Exception e) {
21             System.out.print(s: "Fallo al retirar");
22         }
23         try {
24             System.out.println(x: "Ingreso en cuenta");
25             cuenta1.ingresar(cantidad:695);
26         } catch (Exception e) {
27             System.out.print(s: "Fallo al ingresar");
28         }
29     }
30 }
31
32
```

1. Introducir el método operativa\_cuenta, que englobe las sentencias de la clase Main que operan con el objeto cuenta1.

Para ello, creamos el método en la clase CCuenta:



```
18 //Añadimos el método operativa_cuenta tal y como nos piden en el punto 3 de la práctica
19
20 public void operativa_cuenta(float cantidad) {
21     try {
22         System.out.println("Saldo actual: " + estado());
23         retirar(cantidad:2300);
24         System.out.println("Saldo después del retiro: " + estado());
25         ingresar(cantidad);
26         System.out.println("Saldo después del ingreso: " + estado());
27     } catch (Exception e) {
28         System.out.println("Fallo en la operación: " + e.getMessage());
29     }
30 }
31
32
```

y luego lo llamamos en la Main:

# TAREA PARA ED04

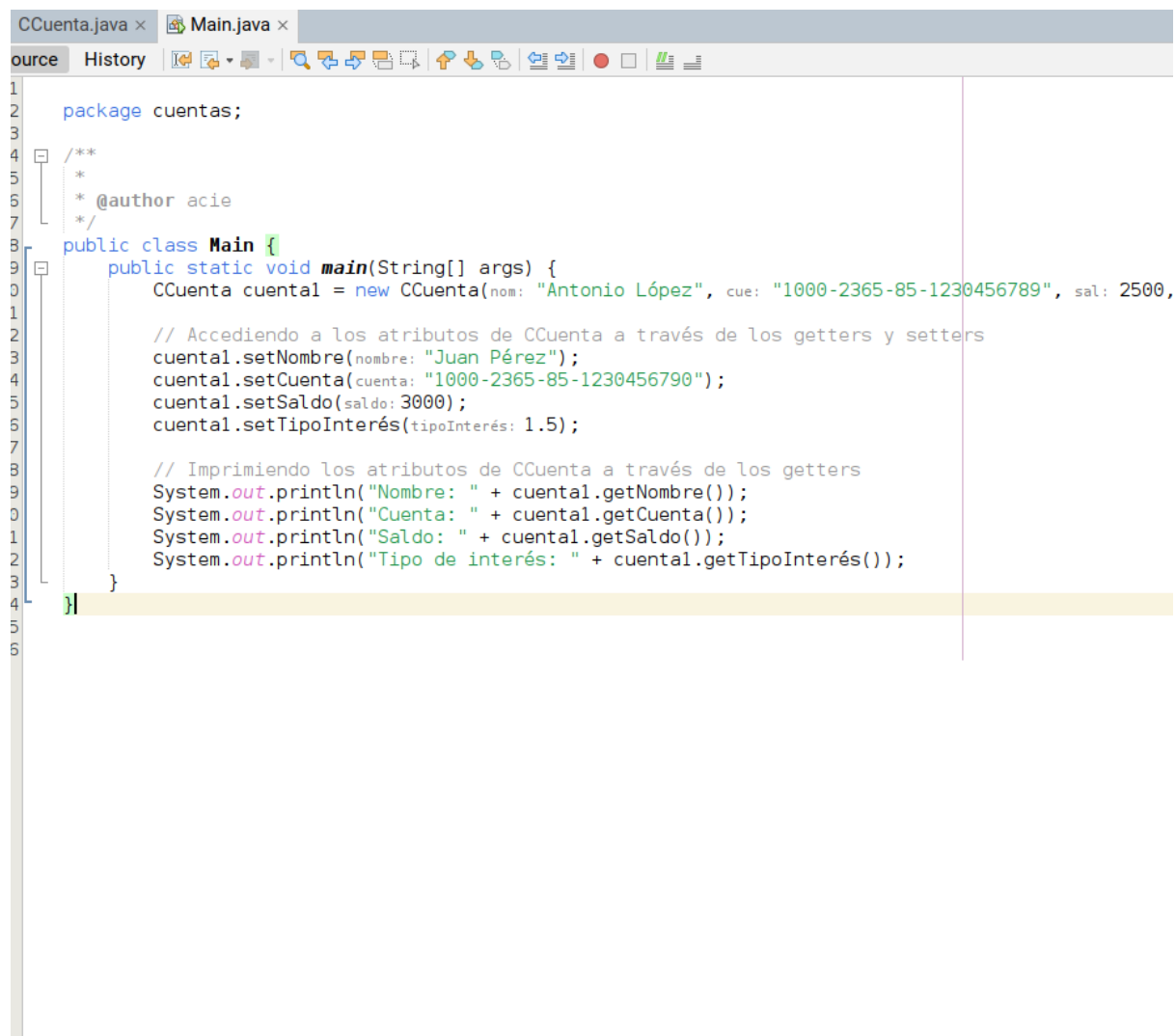
```
6 |  /* @author acie
7 |  */
8 |  public class Main {
9 |
10 |      public static void main(String[] args) {
11 |          CCuenta cuenta1;
12 |          double saldoActual;
13 |
14 |          cuenta1 = new CCuenta(nom: "Antonio López", cue: "1000-2365-85-1230456789", sal: 2500, tipo:
15 |          cuenta1.operativa_cuenta(cantidad: 695);
16 |
17 |          saldoActual = cuenta1.estado();
18 |          System.out.println("El saldo actual es"+ saldoActual );
19 |
20 |          try {
21 |              cuenta1.retirar(cantidad: 2300);
```

## 4. Encapsular los atributos de la clase CCuenta.

Para encapsularlos, deberemos de modificar los atributos para que sean privados y crear los metodos getter y setters para poder llamarlos desde la main:

```
7 |  /* ~
8 |  public class CCuenta {
9 |      private String nombre;
10 |      private String cuenta;
11 |      private double saldo;
12 |      private double tipoInterés;
13 |
14 |      public CCuenta() {
15 |      }
16 |
17 |      public CCuenta(String nom, String cue, double sal, double tipo) {
18 |          nombre = nom;
19 |          cuenta = cue;
20 |          saldo = sal;
21 |          tipoInterés = tipo;
22 |      }
23 |
24 |      public double getSaldo() {
25 |          return saldo;
26 |      }
27 |
28 |      public void setSaldo(double saldo) {
29 |          this.saldo = saldo;
30 |      }
31 |
32 |      public String getNombre() {
33 |          return nombre;
34 |      }
35 |
36 |      public void setNombre(String nombre) {
37 |          this.nombre = nombre;
38 |      }
39 |
40 |      public String getCuenta() {
41 |          return cuenta;
42 |      }
43 |
44 |      public void setCuenta(String cuenta) {
45 |          this.cuenta = cuenta;
46 |      }
47 |
48 |      public double getTipoInterés() {
49 |          return tipoInterés;
50 |      }
51 |
52 |      public void setTipoInterés(double tipoInterés) {
```

# TAREA PARA ED04



```
1 package cuentas;
2
3
4 /**
5  *
6  * @author acie
7  */
8 public class Main {
9     public static void main(String[] args) {
10         CCuenta cuental = new CCuenta(nom: "Antonio López", cue: "1000-2365-85-1230456789", sal: 2500,
11
12         // Accediendo a los atributos de CCuenta a través de los getters y setters
13         cuental.setNombre(nombre: "Juan Pérez");
14         cuental.setCuenta(cuenta: "1000-2365-85-1230456790");
15         cuental.setSaldo(saldo: 3000);
16         cuental.setTipoInterés(tipoInterés: 1.5);
17
18         // Imprimiendo los atributos de CCuenta a través de los getters
19         System.out.println("Nombre: " + cuental.getNombre());
20         System.out.println("Cuenta: " + cuental.getCuenta());
21         System.out.println("Saldo: " + cuental.getSaldo());
22         System.out.println("Tipo de interés: " + cuental.getTipoInterés());
23     }
24 }
```

5. Añadir un nuevo parámetro al método operativa\_cuenta, de nombre cantidad y de tipo float.

Para ello añadimos el nuevo parámetro "cantidad" de tipo float al método operativa\_cuenta:

# TAREA PARA ED04

```
//Añadimos el método operativa_cuenta tal y como nos piden en el punto 3 de la práctica
//Añadimos el parámetro "cantidad" de tipo float

public static void operativa_cuenta(CCuenta cuenta, float cantidad) {
    double saldoActual;
    try {
        System.out.println(x: "Ingreso en cuenta");
        cuenta.ingresar(cantidad);
    } catch (Exception e) {
        System.out.print(s: "Fallo al ingresar");
    }
    saldoActual = cuenta.estado();
    System.out.println("El saldo actual es: " + saldoActual);
    try {
        System.out.println(x: "Retirada de efectivo");
        cuenta.retirar(cantidad);
    } catch (Exception e) {
        System.out.print(s: "Fallo al retirar");
    }
    saldoActual = cuenta.estado();
    System.out.println("El saldo actual es: " + saldoActual);
}

}
```

## GIT

1. Configurar GIT para el proyecto. Crear un repositorio público en GitHub.

Creamos el repositorio desde nuestra Terminal, como ya lo teníamos creado desde antes, lo obviamos, pero la práctica estará en el T4 dentro del directorio Entornos\_Desarrollo:

# TAREA PARA ED04

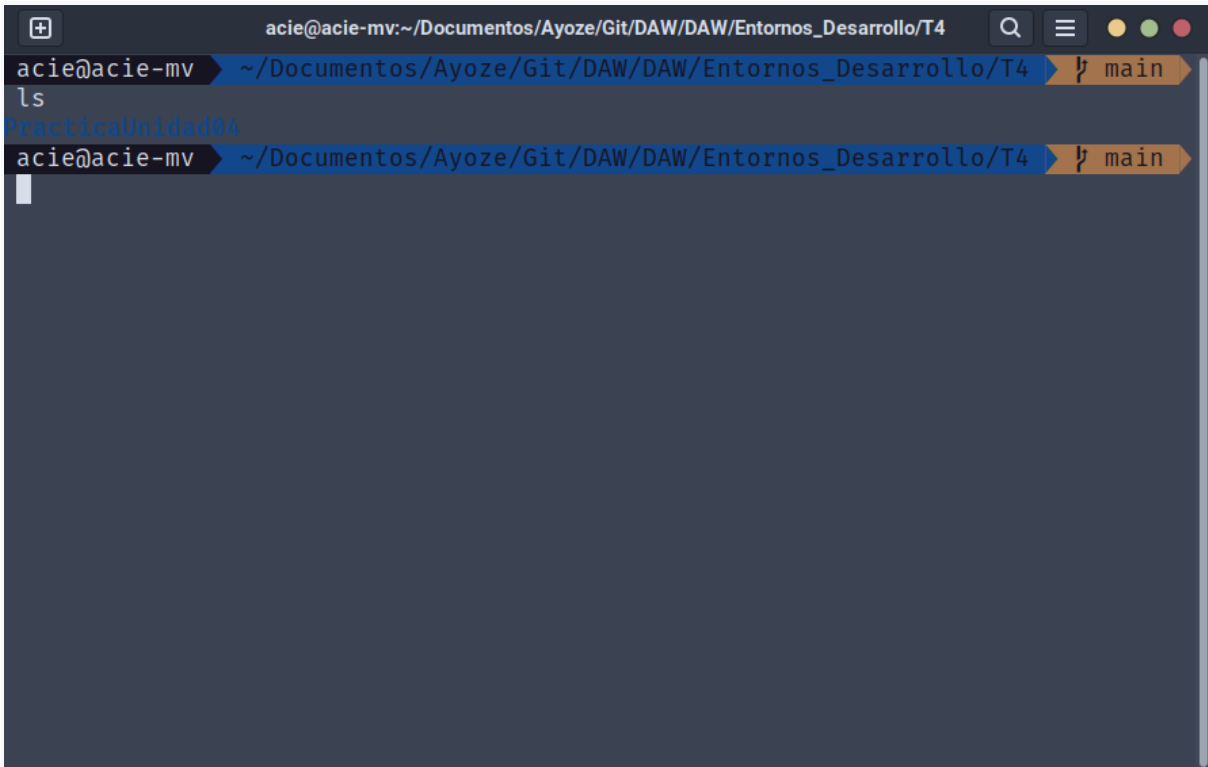
```
acie@acie-mv:~/Documentos/Ayoze/Git/DAW/DAW/Entornos_Desarrollo
acie@acie-mv ~$ cd Documentos
acie@acie-mv ~/Documentos$ cd Ayoze
acie@acie-mv ~/Documentos/Ayoze$ cd Git
acie@acie-mv ~/Documentos/Ayoze/Git$ ls
DAW  JavaCodeLearning
acie@acie-mv ~/Documentos/Ayoze/Git$ cd DAW
acie@acie-mv ~/Documentos/Ayoze/Git/DAW$ ls
DAW
acie@acie-mv ~/Documentos/Ayoze/Git/DAW$ cd DAW
acie@acie-mv ~/Documentos/Ayoze/Git/DAW/DAW$ ls
zsh: command not found: LS
❌ acie@acie-mv ~/Documentos/Ayoze/Git/DAW/DAW$ ls
Entornos_Desarrollo  LND  PROG  README.md  SSIT
acie@acie-mv ~/Documentos/Ayoze/Git/DAW/DAW$ cd Entornos_Desarrollo
acie@acie-mv ~/Documentos/Ayoze/Git/DAW/DAW/Entornos_Desarrollo$ ls
'Guión prácticas.docx'  T1  T2  T3  T4
acie@acie-mv ~/Documentos/Ayoze/Git/DAW/DAW/Entornos_Desarrollo$ cd
```

2. Realizar, al menos, una operación commit. Comentando el resultado de la ejecución.

Antes que nada, tenemos que subir el proyecto a nuestro Github, y después hacerle un commit -m para comentarlo.

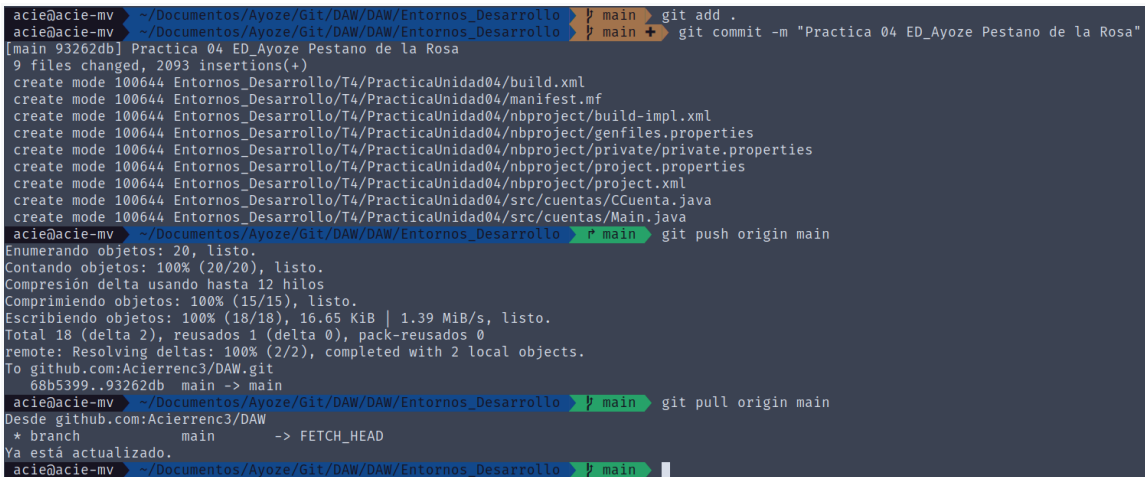


# TAREA PARA ED04



```
acie@acie-mv: ~/Documentos/Ayoze/Git/DAW/DAW/Entornos_Desarrollo/T4
ls
PracticaUnidad04
acie@acie-mv: ~/Documentos/Ayoze/Git/DAW/DAW/Entornos_Desarrollo/T4
```

Realizamos un: **"git add ."** para que nos añada toda la carpeta a nuestro repositorio.  
Posteriormente realizamos un: **"git commit -m "comentario"** para comentar lo que hemos hecho, después realizamos un **"git push origin ....."** para subirlo, y luego es recomendable (cuando estemos desarrollando con más compañeros) realizar un **"git pull origin ....."** para actualizar la maquina local.



```
acie@acie-mv: ~/Documentos/Ayoze/Git/DAW/DAW/Entornos_Desarrollo
acie@acie-mv: ~/Documentos/Ayoze/Git/DAW/DAW/Entornos_Desarrollo
[main 93262db] Practica 04 ED_Ayoze Pestano de la Rosa
9 files changed, 2093 insertions(+)
 create mode 100644 Entornos_Desarrollo/T4/PracticaUnidad04/build.xml
 create mode 100644 Entornos_Desarrollo/T4/PracticaUnidad04/manifest.mf
 create mode 100644 Entornos_Desarrollo/T4/PracticaUnidad04/nbproject/build-impl.xml
 create mode 100644 Entornos_Desarrollo/T4/PracticaUnidad04/nbproject/genfiles.properties
 create mode 100644 Entornos_Desarrollo/T4/PracticaUnidad04/nbproject/private/private.properties
 create mode 100644 Entornos_Desarrollo/T4/PracticaUnidad04/nbproject/project.properties
 create mode 100644 Entornos_Desarrollo/T4/PracticaUnidad04/nbproject/project.xml
 create mode 100644 Entornos_Desarrollo/T4/PracticaUnidad04/src/cuentas/CCuenta.java
 create mode 100644 Entornos_Desarrollo/T4/PracticaUnidad04/src/cuentas/Main.java
acie@acie-mv: ~/Documentos/Ayoze/Git/DAW/DAW/Entornos_Desarrollo
Enumerando objetos: 20, listo.
Contando objetos: 100% (20/20), listo.
Compresión delta usando hasta 12 hilos
Comprimiendo objetos: 100% (15/15), listo.
Escribiendo objetos: 100% (18/18), 16.65 KiB | 1.39 MiB/s, listo.
Total 18 (delta 2), reusados 1 (delta 0), pack-reusados 0
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To github.com:Acierrenc3/DAW.git
 68b5399..93262db main -> main
acie@acie-mv: ~/Documentos/Ayoze/Git/DAW/DAW/Entornos_Desarrollo
Desde github.com:Acierrenc3/DAW
* branch      main      -> FETCH_HEAD
Ya está actualizado.
acie@acie-mv: ~/Documentos/Ayoze/Git/DAW/DAW/Entornos_Desarrollo
```

Como comprobamos, hemos subido la Práctica 04 a nuestro Github personal:

# TAREA PARA ED04

DAW / Entornos_Desarrollo /			Add file	
Acierrenc3 Practica 04_ED_Ayoze Pestano de la Rosa			93262db - 2 minutes ago	History
Name	Last commit message	Last commit date		
..				
T1	Entornos de Desarrollo	yesterday		
T2	Entornos de Desarrollo	yesterday		
T3	Entornos de Desarrollo	yesterday		
T4/PracticaUnidad04	Practica 04_ED_Ayoze Pestano de la Rosa	2 minutes ago		
Guión prácticas.docx	Entornos de Desarrollo	yesterday		

1. Mostrar el historial de versiones para el proyecto mediante un comando desde consola.

Para mostrar el historial de versiones del proyecto desde la consola, se puede utilizar el comando "git log". Para ello, es necesario tener instalado Git en el equipo y haber inicializado el repositorio Git en el directorio donde se encuentra el proyecto:

```
git log
commit 93262db59a3bd7da61eaf4d3b49456830c7ecd07 (HEAD -> main, origin/main, origin/HEAD)
Author: Acierrenc3 <Ayopestano@gmail.com>
Date: Mon May 8 19:49:56 2023 +0100

    Practica 04_ED_Ayoze Pestano de la Rosa

commit 68b53999232b8ee71b9e435669a7cc02dfbcb67
Author: Acierrenc3 <Ayopestano@gmail.com>
Date: Mon May 8 18:40:56 2023 +0100

    .

commit 9974747fda85f33c961db3e3b39e3e5ecac39dd8
Author: Acierrenc3 <Ayopestano@gmail.com>
Date: Mon May 8 18:39:36 2023 +0100

    prueba

commit 8b5bdbddd715e580d88d590d952e411ce64b99cb
Author: Acierrenc3 <Ayopestano@gmail.com>
Date: Sun May 7 15:32:02 2023 +0100

    Programacion Java

commit f15821d12abab0641b160d54420d54a38422f50a
Author: Acierrenc3 <Ayopestano@gmail.com>
Date: Sun May 7 15:31:04 2023 +0100

    SSII

commit fc2e4a2e67b6deb8dc0614aa292e8673f9de9e43
Author: Acierrenc3 <Ayopestano@gmail.com>
Date: Sun May 7 15:30:24 2023 +0100

    LND

commit 6d2ca8320fb907f07e721f0140a8782c39c09e42
Author: Acierrenc3 <Ayopestano@gmail.com>
Date: Sun May 7 15:29:40 2023 +0100

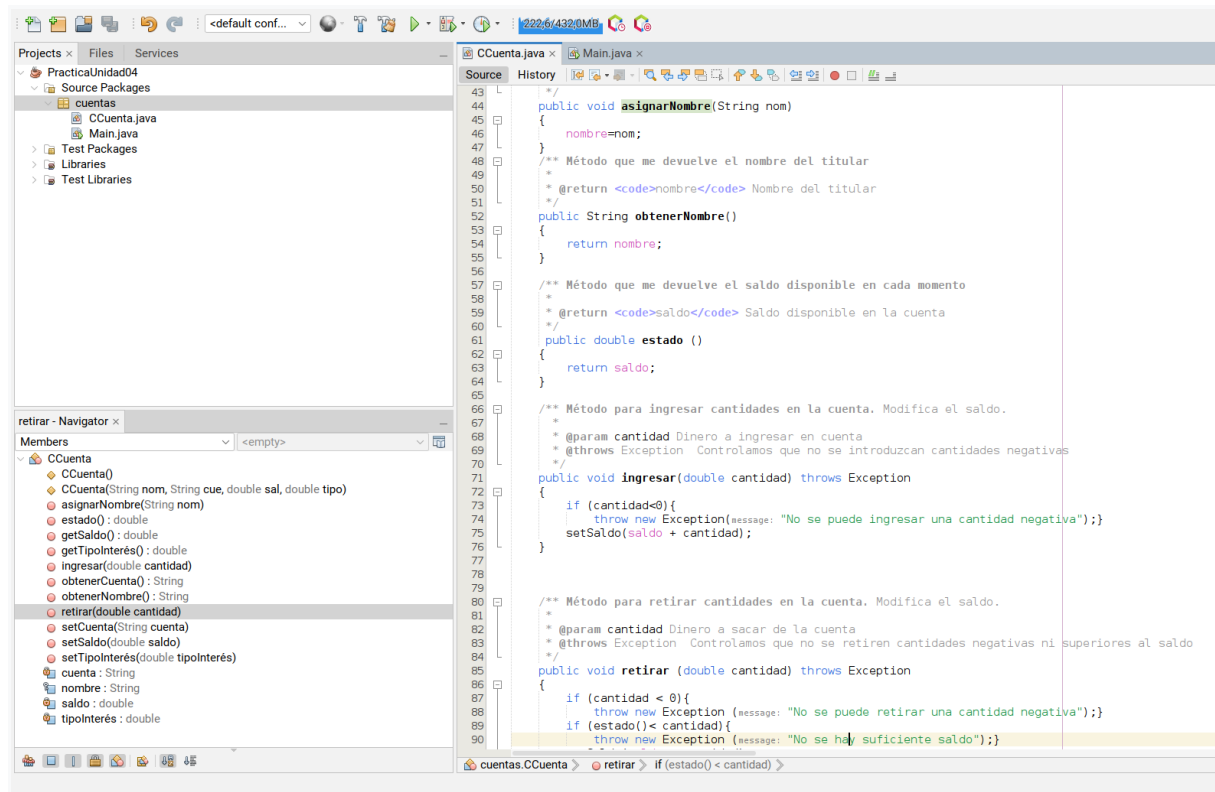
    Entornos de Desarrollo

commit 1de3618c1ded38c36270cacb8ee0d8292176e2f3
Author: Acierrence <126329342+Acierrenc3@users.noreply.github.com>
Date: Sun May 7 15:15:52 2023 +0100
```

# TAREA PARA ED04

## JAVADOC

1. Insertar comentarios JavaDoc en la clase CCuenta.
2. Generar documentación JavaDoc para todo el proyecto y comprueba que abarca todos los métodos y atributos de la clase CCuenta.



## CONCLUSIÓN:

En resumen, el uso de herramientas como Netbeans, Github y Git puede mejorar significativamente el proceso de desarrollo de software en equipo. Estas herramientas permiten una mayor colaboración entre desarrolladores, una mejor gestión del código y una mayor eficiencia en el desarrollo de proyectos de software.

# TAREA PARA ED04

---