**Part 1: Depuració del Codi**

**Objectiu:** *entendre com es gestionen les operacions de dipositar i retirar, com es modifiquen les variables i com es capturen les excepcions en cas d'errors.*

Documenteu els següents passos:

1. **Descripció inicial del codi:**

* **Què fa el codi? (Explicar breument).**

El código implementa un sistema bancario básico a través de una clase Account, que permite al usuario manipular el saldo mediante las operaciones de ingreso y retirada de dinero. La clase Main es la que construye una cuenta de cliente y realiza operaciones con ella, comprobando las posibles excepciones**.**

* **Quins són els mètodes més importants i què fan?**

**depositAmount(double amount)**

* **Afegeix un import al saldo del compte.**
* **Llença una excepció si l’import és negatiu.**

**withdrawAmount(double amount)**

* **Resta un import del saldo si hi ha prou diners.**
* **Llença excepcions si l’import és negatiu o si el saldo és insuficient.**

**getBalance()**

* **Retorna el saldo actual del compte.**
* **Quin és el valor inicial del saldo (balance) abans de realitzar qualsevol operació?**

El compte myAccount es crea amb un saldo inicial de **2500**.

2. **Posar punts de control (Breakpoints):** *Per depurar el codi, utilitza els punts de control (breakpoints). Això permet aturar l'execució del codi en determinats punts i examinar l'estat de les variables. Per afegir un punt de control, fes clic a la barra de l’esquerra de la línia on vols aturar el codi.*

* **On has col·locat els punts de control (breakpoints) i per què?**
* **Inclou una captura de pantalla de Eclipse amb els breakpoints activats abans de començar la depuració**.

**Línia 9 de Main.java → Just abans de cridar withdrawAmount(2300) per veure l'estat inicial del compte.**

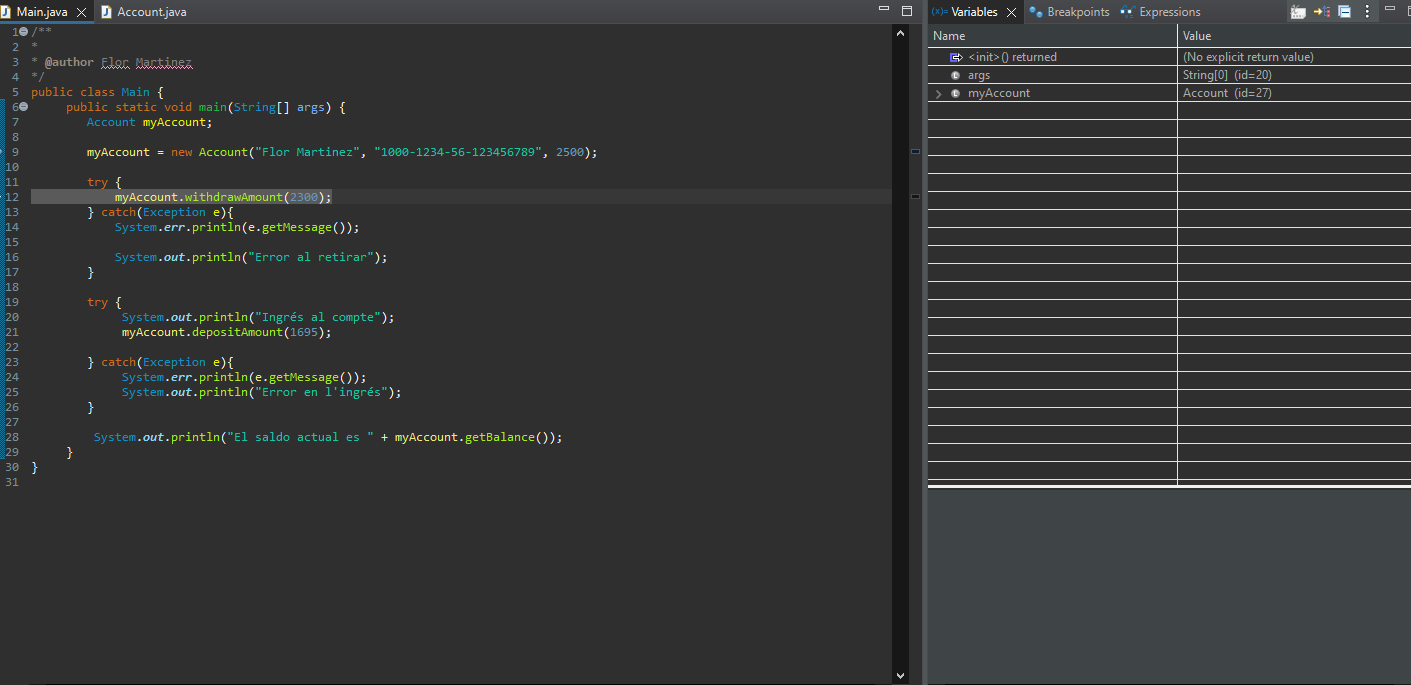
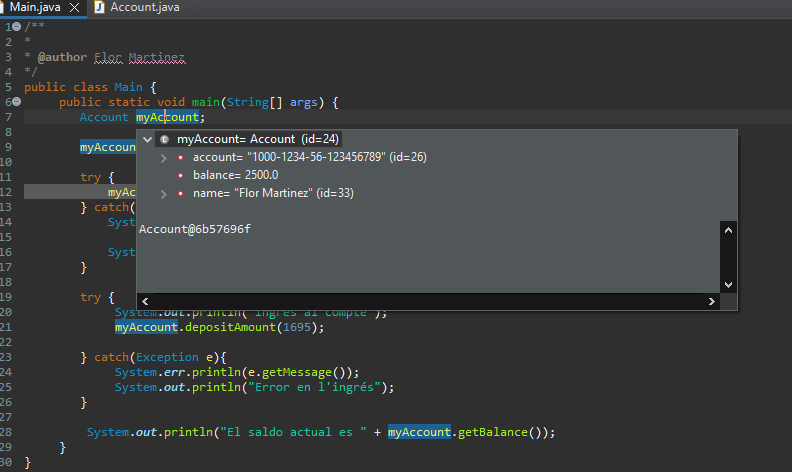
**Línia 16 de Main.java → Abans de fer el dipòsit per verificar l’estat després de la retirada.**

**Línia 22 de Main.java → Abans d'imprimir el saldo final per veure si s'ha actualitzat correctament.**

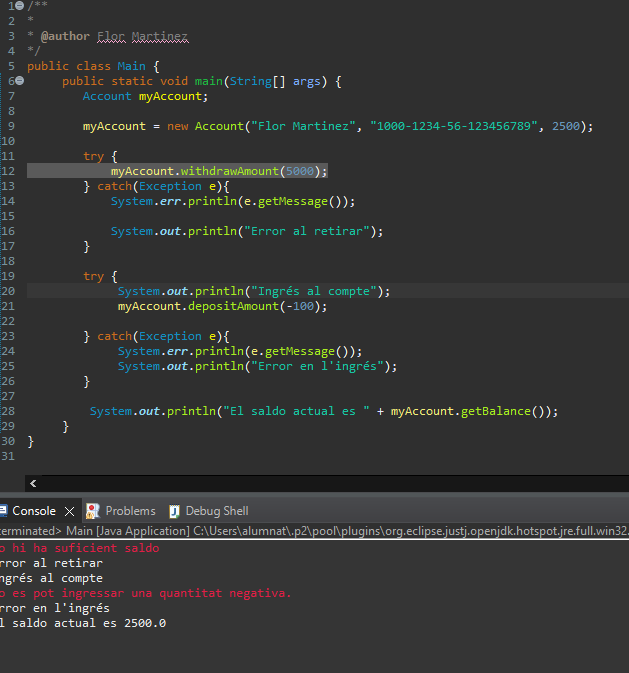
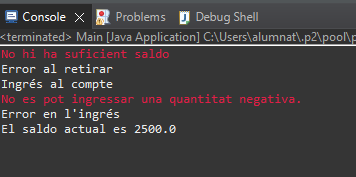


**3. Examina les variables i el flux d'execució:**

* *A mesura que el codi s'atura a cada punt de control, observa el valor de les variables name, account i balance. Inclou una captura de pantalles dels valors de les variables a mesura que avancen les operacions.*



**4. Explora les excepcions:**

* *Feu els canvis necessaris al Main per fer saltar les excepcions. Inclou la captura de pantalla d'un missatge d'error generat per una excepció i com es visualitza al terminal o a la consola de Eclipse.*

*Activitat 3-*

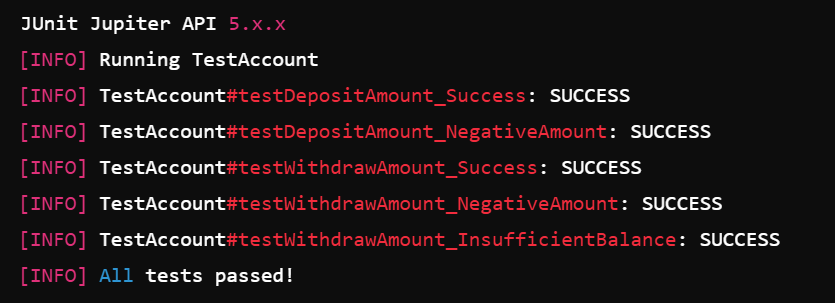
*Objectiu: Escriure proves unitàries per als mètodes depositAmount() i withdrawAmount() per garantir que el codi funciona correctament i gestionar les excepcions adequadament.*

*1. Crear una classe de proves que es digui “AccountTest” i contingui dos mètodes: un per provar els dipòsits i un altre per provar les retirades.*

*Texto

Descripción generada automáticamente*

*2. Les proves han de verificar casos d'èxit, així com casos amb errors (quantitat negativa, saldo insuficient). Utilitzeu el mètode assertEquals.*

**

*REPOSITORI GITHUB ->* [*https://github.com/Aarongp13/SimpleBankingSystem.git*](https://github.com/Aarongp13/SimpleBankingSystem.git)