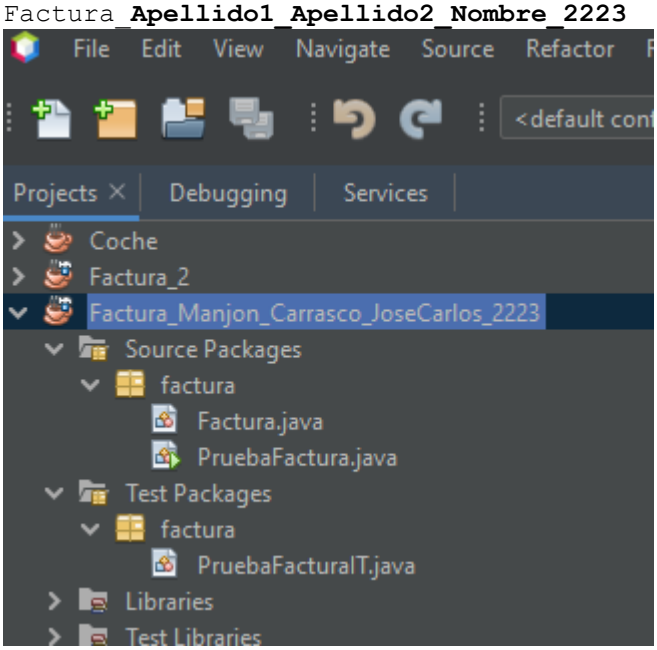
Tarea 3

Entornos de desarrollo

Jose Carlos Manjon Carrasco

1.- Renombra el proyecto con tu nombre quedando de la siguiente forma:

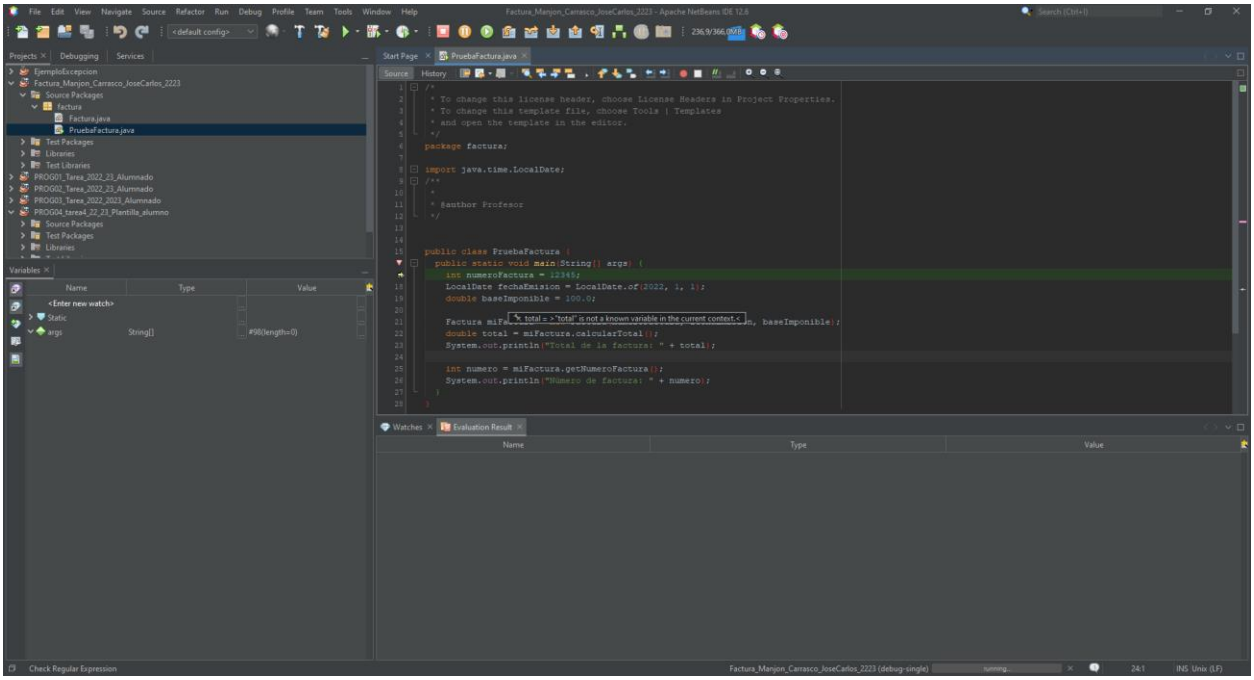
`Factura_`**`Apellido1_Apellido2_Nombre_2223`**



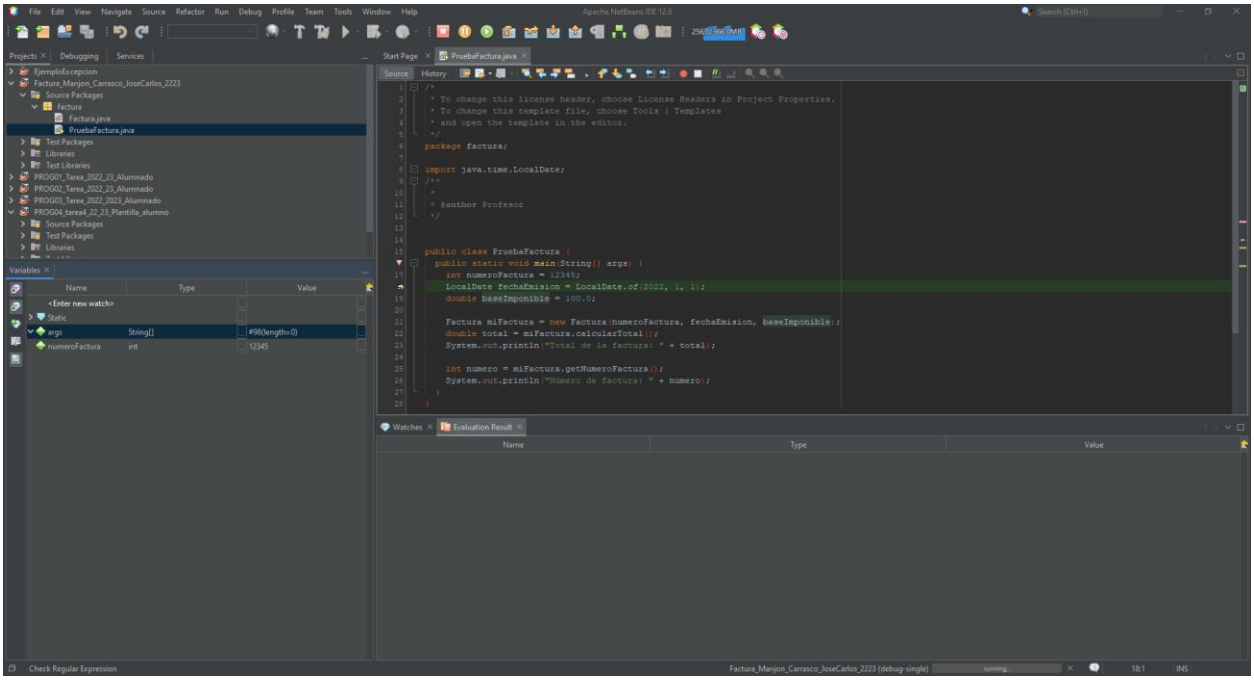
2.- Pon un punto de ruptura en una línea de la clase ***PruebaFactura***. Realiza una ejecución paso a paso, que verifique el correcto funcionamiento de la aplicación. Muestra los valores que marca la inspección de variables tras ejecutar las instrucciones en la función main, visualizando el valor de las variables cuando llegue al punto de ruptura.

Copia en un documento de texto la captura de pantalla en la que se visualice el punto de ruptura, el avance paso a paso y el valor de las variables después de la llamada a cada método

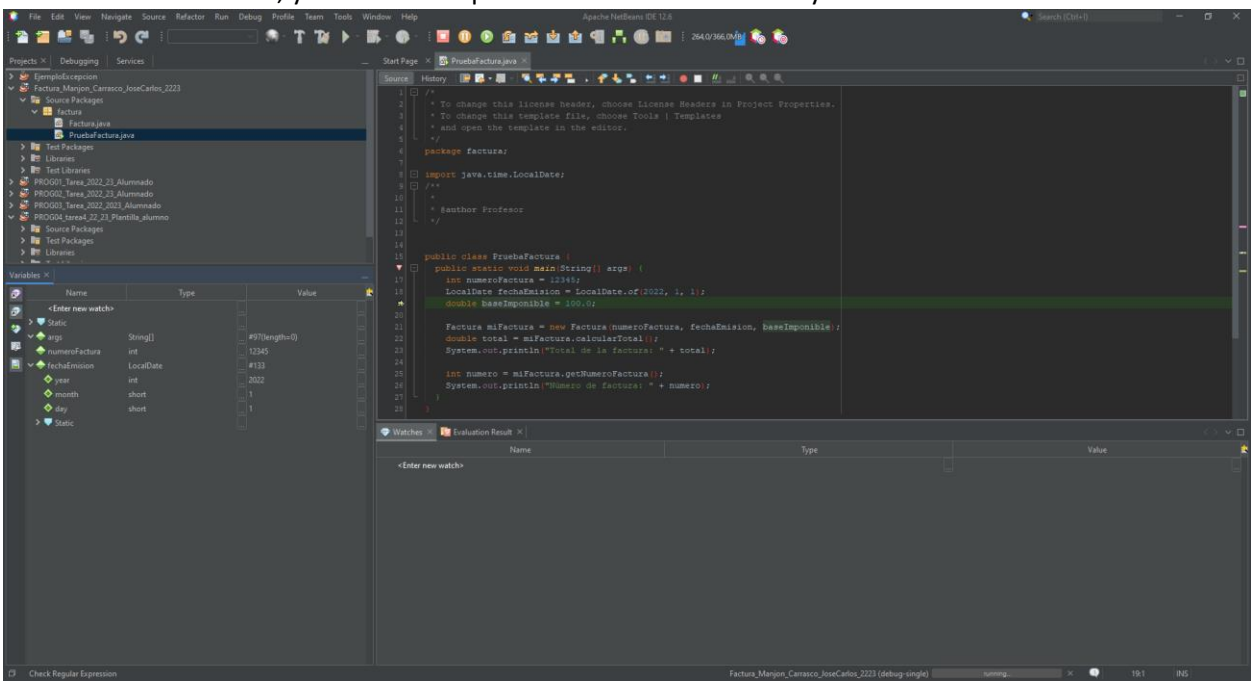
Debugger para en línea 16



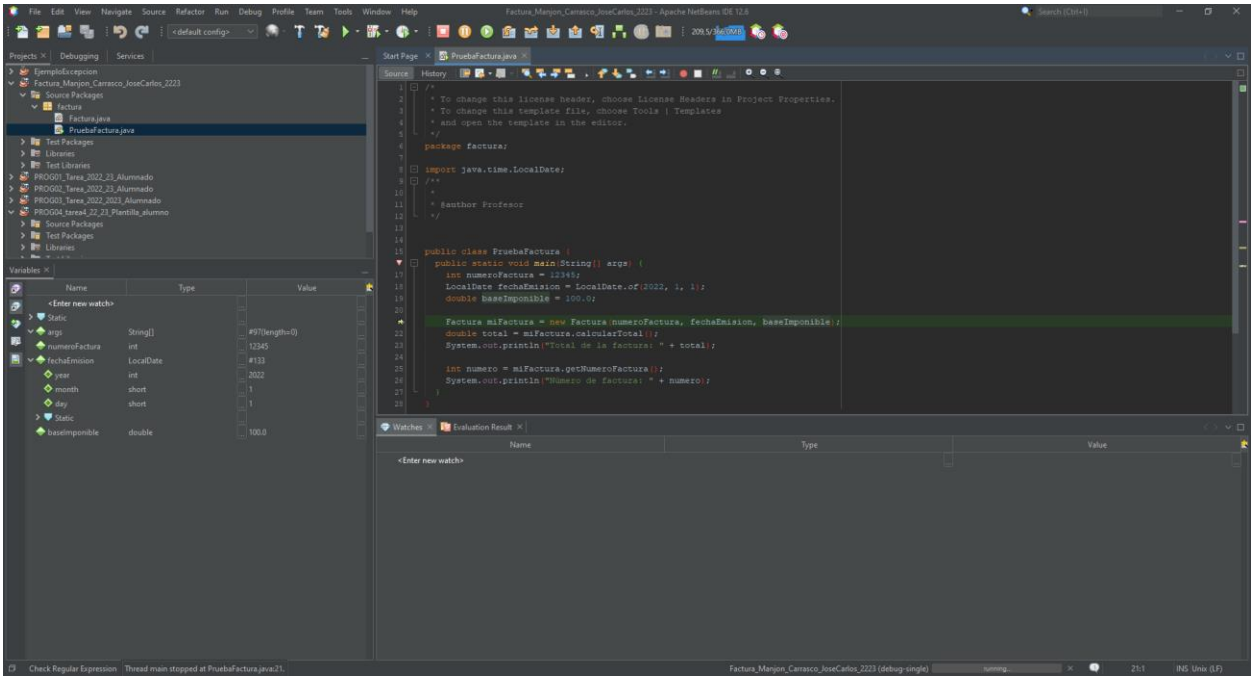
Paramos en línea 18, y observamos que se ha reconocido el valor de la variable numeroFactura con un valor 12345



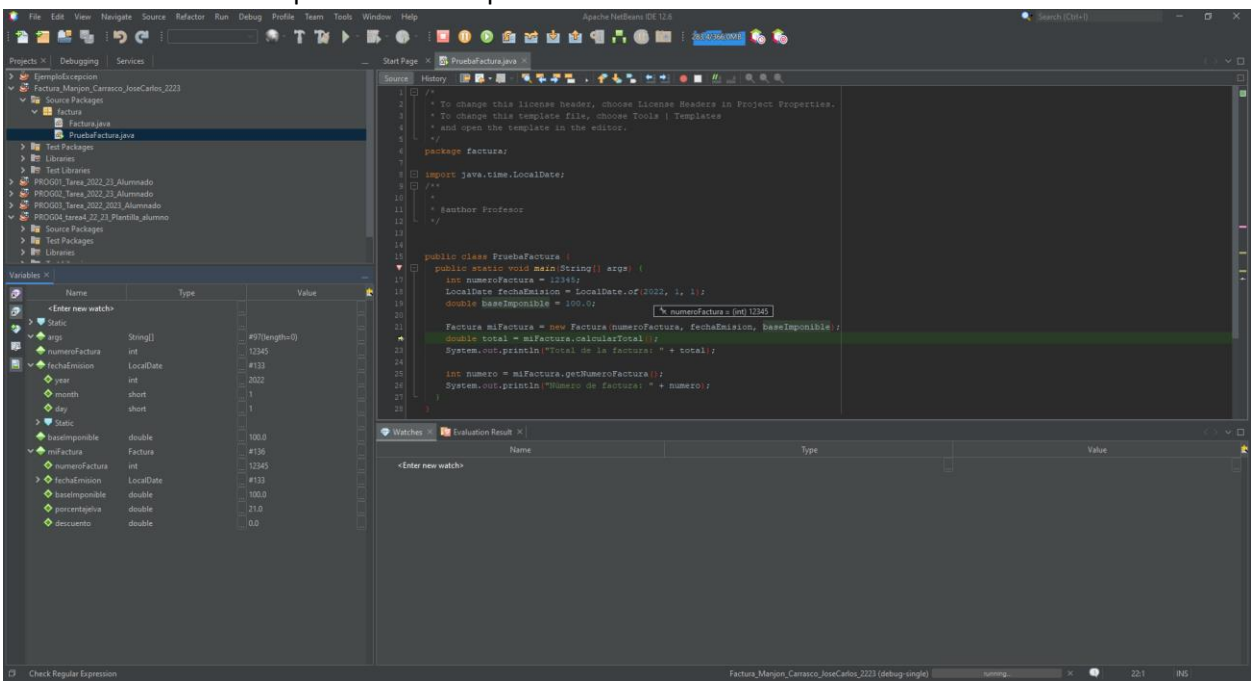
Continuamos una línea, y observamos que la variable fechaEmision ya tiene un valor



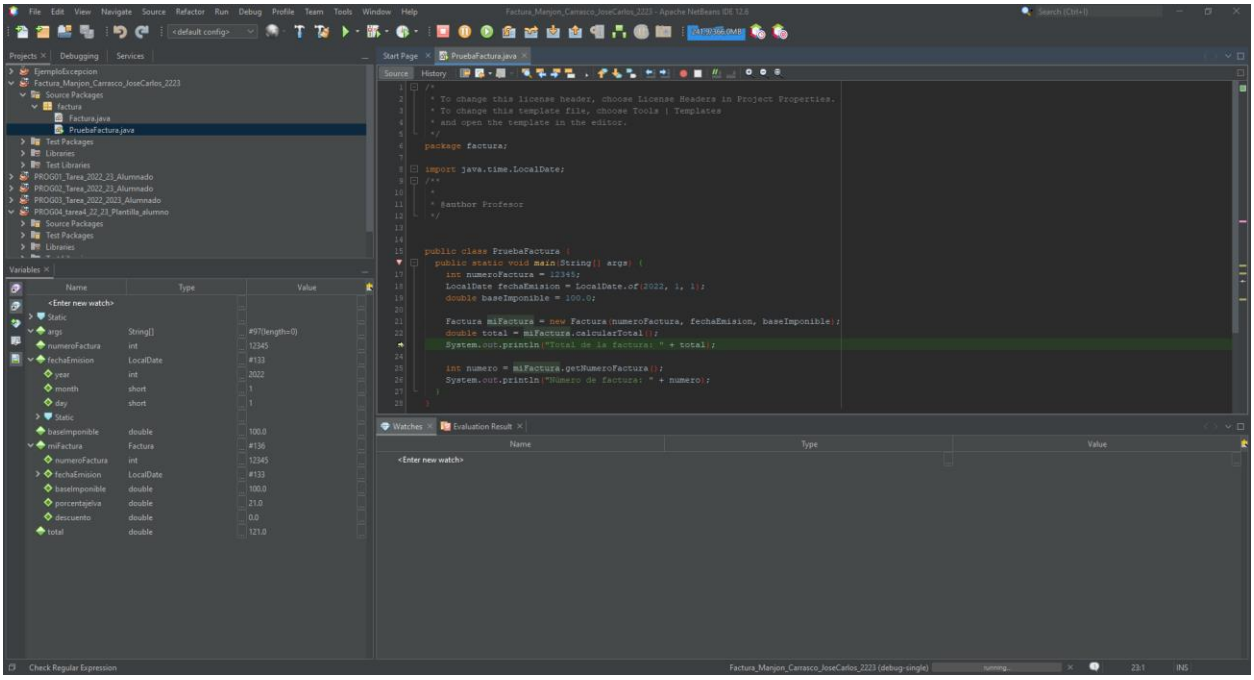
Continuamos una línea , y la variable baseImponible ya tiene un valor



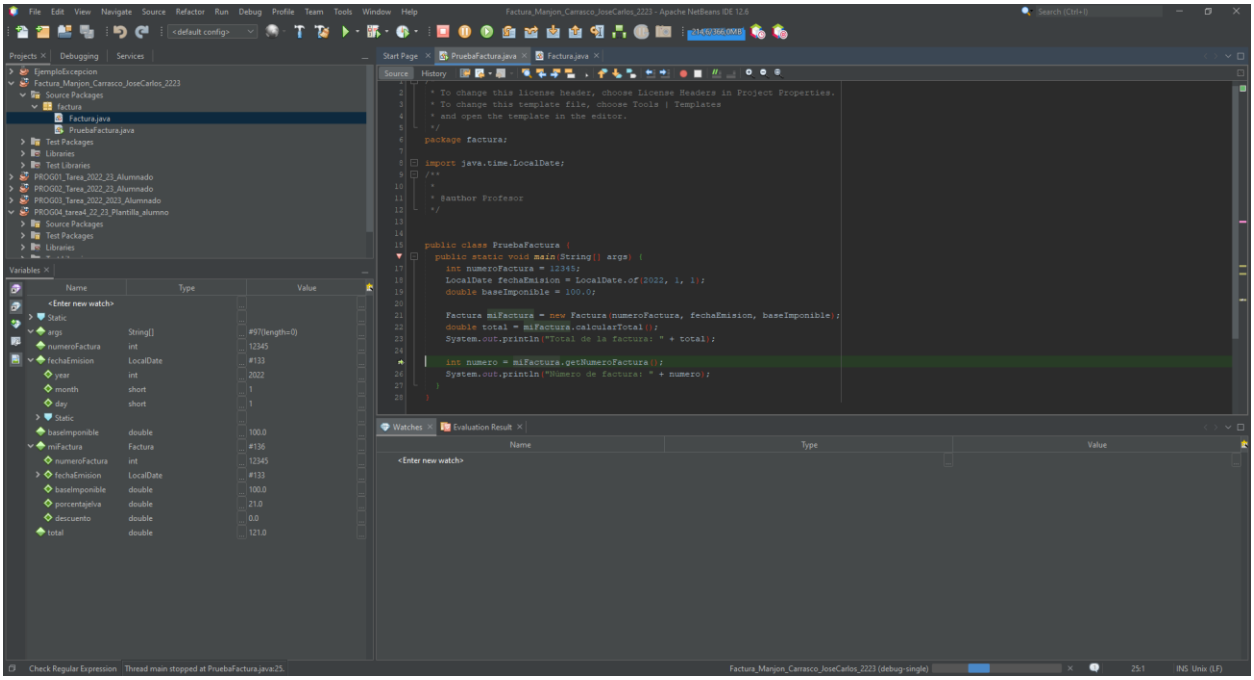
Continuamos una línea y comprobamos que la factura se ha instanciado correctamente, cogiendo los valores de las variables que hemos visto previamente



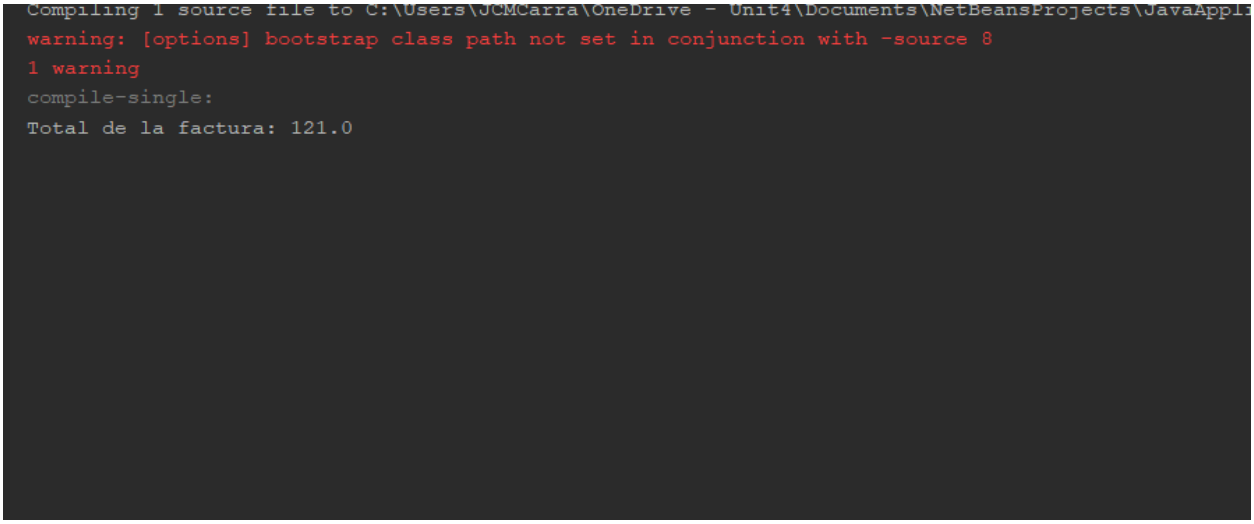
Continuamos una línea , y se calcula el computo total de la factura, donde el importe del Iva se coje de una variable presente en Factura



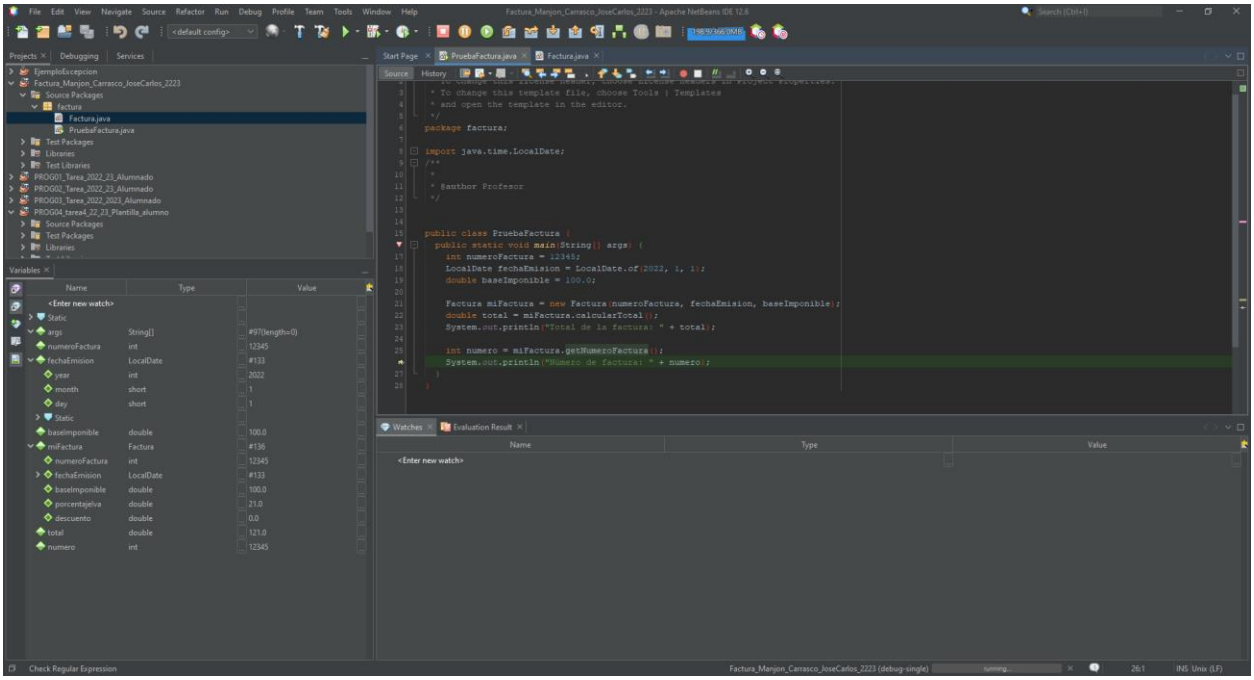
En la siguiente línea, definimos como se calcula el número de factura



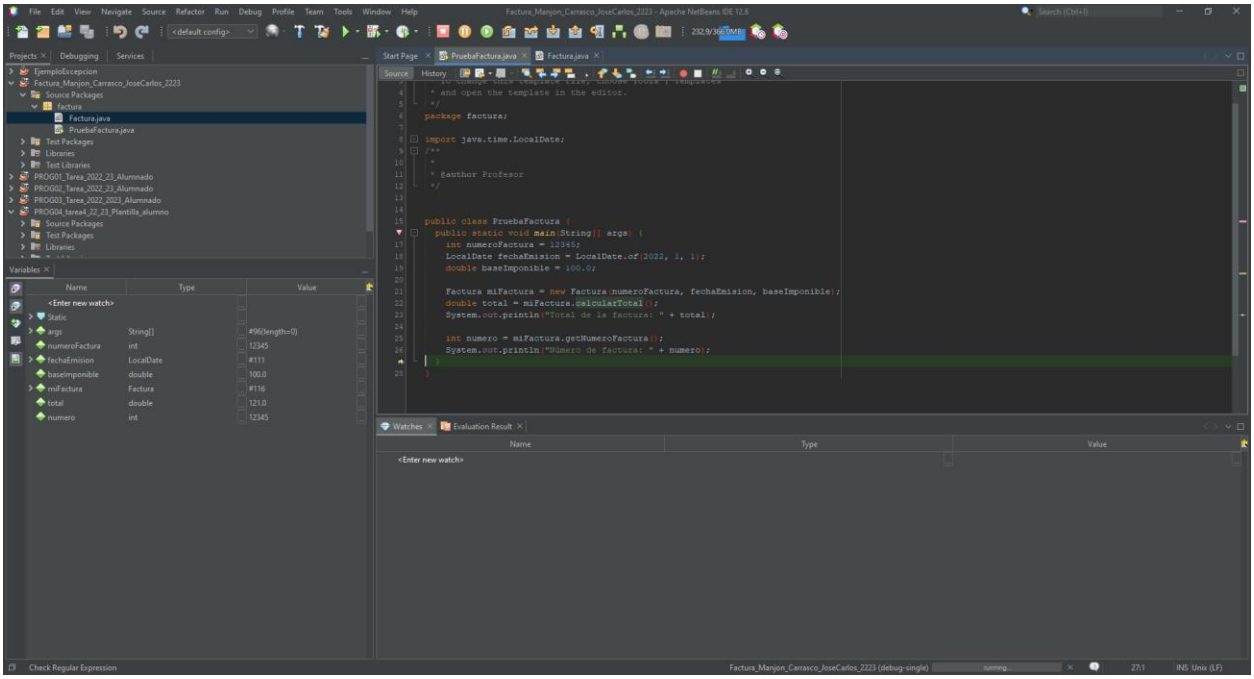
En la siguiente línea tras haber recorrido la línea 23 se imprime



En la siguiente línea obtenemos el número de factura, que podemos verlo en las variables



En el siguiente paso ,tras haber recorrido la línea 26, se imprime por pantalla en la línea 27



```
warning: [options] bootstrap class path not set in conjunction with -source 8
1 warning
compile-single:
Total de la factura: 121.0
Número de factura: 12345
```

En el siguiente paso, termina la ejecución, con la siguiente información

```
warning: [options] bootstrap class path not set in conjunction with -source 8
1 warning
compile-single:
Total de la factura: 121.0
Número de factura: 12345
debug-single:
BUILD SUCCESSFUL (total time: 1 minute 43 seconds)
```

3.- Realiza todos los casos de prueba posibles para el método calcularTotal() de la clase Factura.

Para detectar los posibles casos de prueba de calcularTotal(), debemos identificar la fórmula, nos dirigimos al archivo Factura.java, calcularTotal () obtiene a partir de baseImponible, el IVA correspondiente a esa base imponible y un posible descuento, al ser todos los parámetros de carácter numérico, contemplaremos valores menores, iguales o mayores que cero.

Parámetro	Valor inválido	Valor válido
Base imponible	<=0 (1)	>0 (2)
Iva	<=0(3) a >21 (4)	>0 a <= 21(5)
Descuento	<0 (6) y >=baseImponible+Iva (7)	>=0 a <baseImponible+Iva (8)

Gráficos, siendo lo rojo valores inválidos y el verde valores válidos

Base imponible

Iva

Descuento

El cálculo total cuando es de signo positivo, lo entiendo como que el vendedor de dicho artículo recibe dinero por parte del comprador, por lo que, si es de signo negativo, el vendedor le entrega dinero al comprador por la adquisición del artículo.

(1)→ Carece de sentido que el cálculo de un precio sea negativo o cero, lo que indicaría que el establecimiento le abona al cliente por adquirir un artículo, que se pudiera ver incrementado por un posible descuento e Iva en el caso de valor menor que 0 , o si es 0 la baseImponible, el Iva ya sería 0, por lo que el parámetro descuento si es 0, el artículo es gratuito, si es mayor que 0 el establecimiento le abona al cliente por adquirir el artículo o si es menor que 0, un descuento negativo carece de sentido.

(2) → Apoyándome en la motivación de (1), el parámetro crítico es el importe pues es la base del cálculo, está implícito o explícitamente presente en 2 de 3 parámetros.

(3)→Carece de sentido un Iva menor que 0, pues en lugar de gravar un artículo, se restaría el importe correspondiente, se ha puesto como valor límite superior 21, por ser el actual máximo porcentaje de Iva en España.

(4) → Si el rango de valores inválidos es menor a 0 y mayor que 21, todo el abanico de valores entre ambos serán valores válidos como 0.01 y 21.0.

(5)→Descuento menor a 0, tal como está la formula indicaría un recargo al artículo en lugar de descontar parte del importe.

(6)→ El valor máximo se ha determinado como la suma de la baseImponible y el correspondiente valor válido de Iva, es decir, si la baseImponible es 100 y se le aplica un Iva del 10%, la suma sería 110.0, si el descuento fuera de 110.1, indicaría que el establecimiento le abonaría al cliente esa diferencia de 0.1, que carece de sentido.

(7)→ El valor base, lo he establecido en valor mínimo de 0 es decir no habría descuento, y un numero positivo sería aceptado siempre que no superase la suma de (8), lo que indicaría que el descuento es superior a la baseImponible más el correspondiente Iva.

(8)→ Basándome en (7), el límite superior estaría en cualquier importe que sea menor que la baseImponible sumando el correspondiente Iva, lo que indicaría que por muy grande que fuera el descuento el artículo no sería gratis o que el cliente recibiese dinero por adquirir el artículo.

4.- A continuación, diseñamos los casos de prueba necesarios para comprobar si se obtiene la salida deseada cuando se introducen valores no válidos (se debe diseñar un caso de prueba para cada tipo de valor no válido posible).

Caso	Clase de equivalencia	Resultado esperado
BaseImponible =100 Iva=21 Descuento->no aplicado	(2),(5)	121
BaseImponible =100 Iva=21 Descuento= 0	(2),(5)(8)	121
BaseImponible =0 Iva=20 Descuento=0	(1),(5),(8)	Mensaje de error (no puede haber una base imponible que sea 0)
BaseImponible =-0.1 Iva=20 Descuento= 0	(1),(5),(8)	Mensaje de error (no puede haber una base negativa)
BaseImponible =50 Iva=-0.1 Descuento= 0.1	(2),(3),(8)	Mensaje de error (no puede haber un iva negativo)
BaseImponible =50 Iva=0 Descuento= 0.1	(2),(3),(8)	Mensaje de error (no puede haber un iva que sea 0)
BaseImponible =40 Iva=21.1 Descuento=5	(2),(4),(8)	Mensaje de error (no puede haber un iva mayor de 21)
BaseImponible =50 Iva=21 Descuento= 0.55	(2),(5),(8)	59.95
BaseImponible =100 Iva=21 Descuento= -0.1	(2),(5),(6)	Mensaje de error (no puede haber un descuento negativo)
BaseImponible =100 Iva=21 Descuento= 121.1	(2),(5),(7)	Mensaje de error (no puede haber un descuento mayor que la suma de la base imponible mas el iva)

He detallado los casos que me han parecido más destacables, puesto que se podrían haber realizado infinidad de test de jUnit, los cuales en mi opinión serían redundantes, pues se podrían resumir en los que he detallado, puesto que apoyándome en el apartado 3, si el valor válido de base imponible es >0 da igual si probamos con 0.00001 o con 100000000, está en el mismo rango de valores, en >0.

5.- Instalamos JUnit en Netbeans y realizamos la implementación de las pruebas de los métodos ***calcularTotal()*** y ***setPorcentajeIva()*** de la clase ***Factura*** con JUnit. Generamos todas las clases Test y TestSuite correspondientes en nuestro proyecto.