GUIA I DESARROLLO DE SOFTWARE

Estudiante:

Johan Sebastian Alonso Gutiérrez

Universidad EAN

Profesor:

Dilsa Enith Triana Martínez

19/08/2024

**Introducción**

En presente trabajo, se aplicará el conocimiento adquirido en las diferentes tutorías brindadas por la profesora y se abarcarán temas del desarrollo de software tales como la construcción de diagramas de clase y programación orientada a objetos.

**Objetivos**

1. Aplicar el conocimiento adquirido en la construcción de diagramas con apoyo de la herramienta Star UML
2. Solucionar diferentes situaciones mediante la programación orientada a objetos
3. Solucionar la guía I establecida en la presente materia

**Guía I**

**Los ejercicios que se presentan a continuación se encuentran almacenados en el siguiente enlace:**

**https://github.com/jalonso37520/DESARROLLO-DE-SOFTWARE.git**

1. La clase Fraccionario: debemos poder almacenar el numerador y el denominador del  
   Fraccionario.

Se crea la clase “fraccionario” y se agregan los atributos propios de la clase. (Numerador, fraccionario). Junto con el tipo de dato que deben contener

A screenshot of a computer

Description automatically generated

Se genera el código en Java desde el editor

A screenshot of a computer

Description automatically generated

Se agregan los diferentes comportamientos que obtiene

A screenshot of a computer

Description automatically generated

Se ubica la carpeta que se va a usar para almacenar el código.

A screenshot of a computer

Description automatically generated

Se verifica el código, respectivo generado

A screenshot of a computer

Description automatically generated

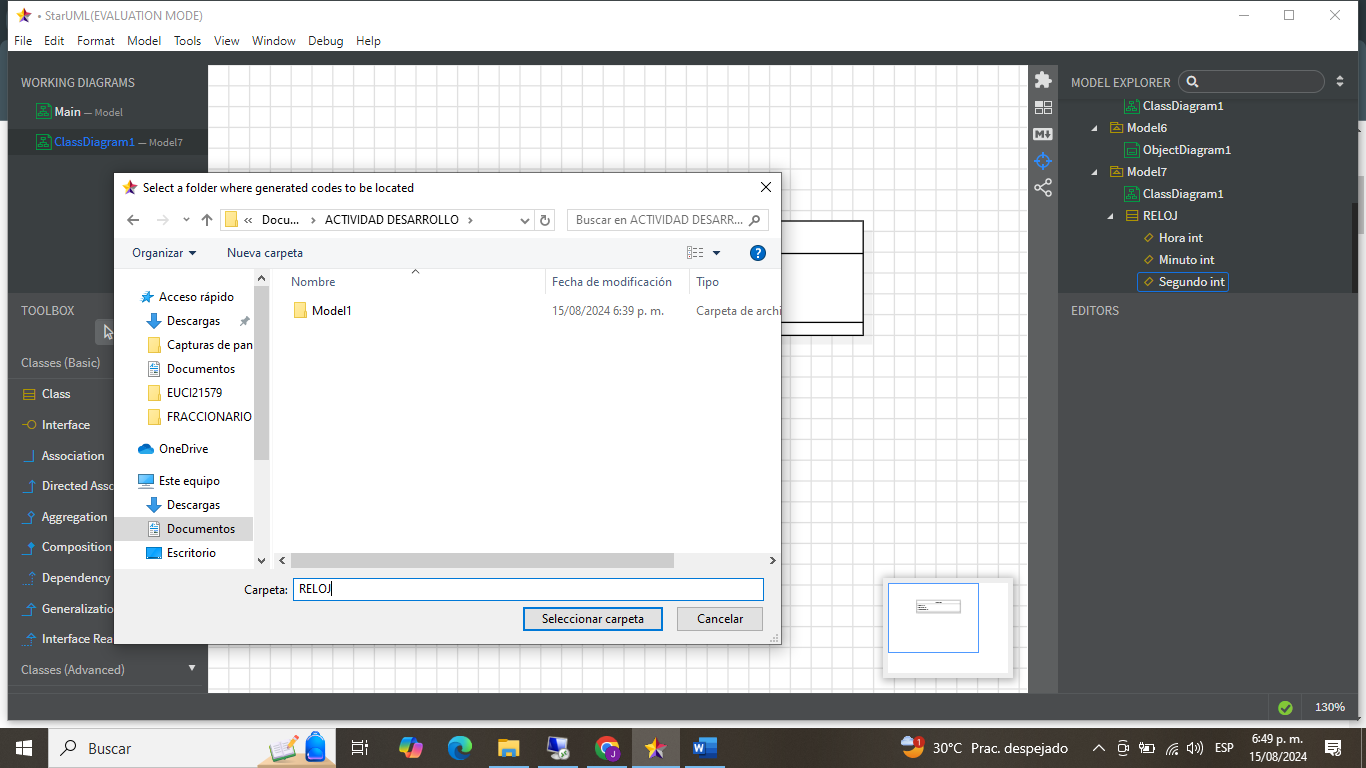
1. Un Reloj es una clase que almacena la hora, los minutos y los segundos.

Se crea la clase “Reloj” y se agregan los atributos de horas, minutos y segundos con su respectivo dato.

A screenshot of a computer

Description automatically generated

Posterior, se genera el código en Java del ejercicio realizado

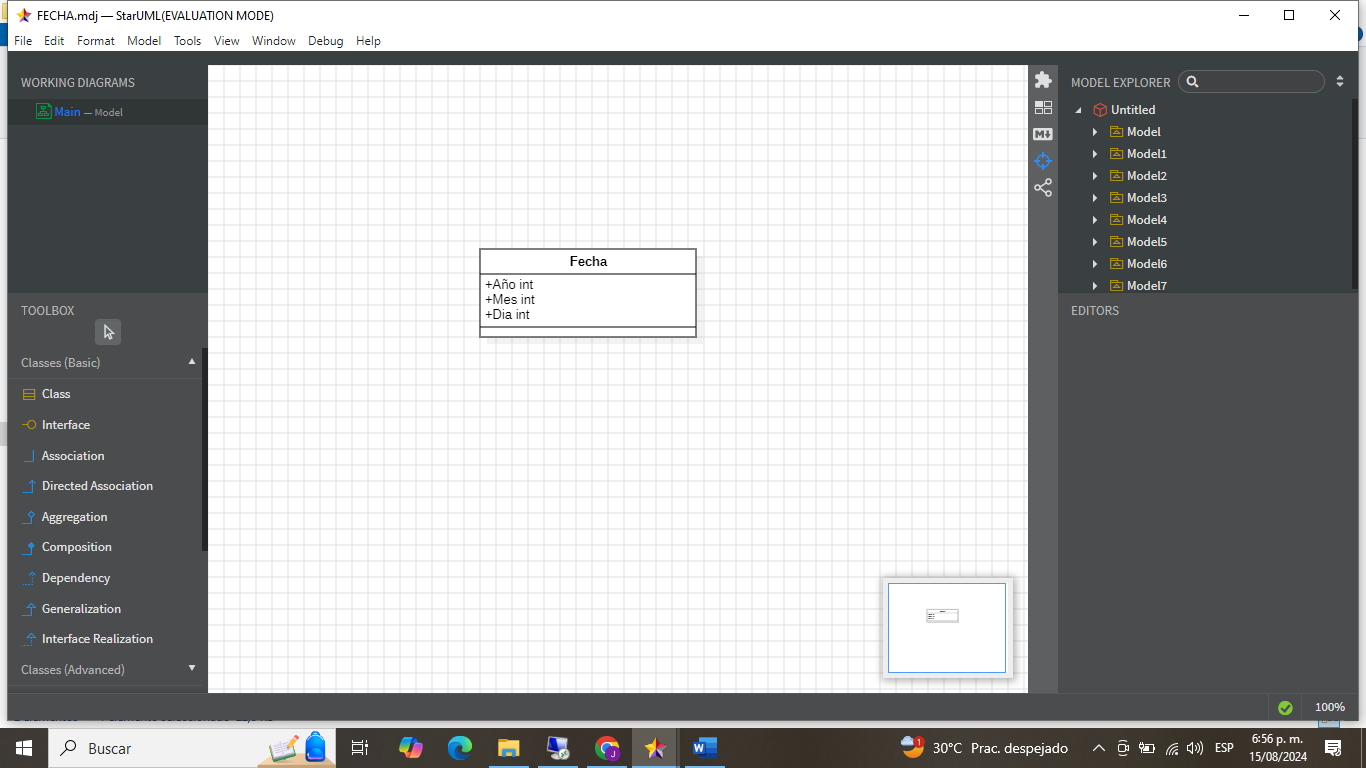


A screenshot of a computer

Description automatically generated

1. La clase Fecha permite almacenar año, mes y día

Se crea la clase fecha y se agregan los atributos de año, mes y día



1. Un proyectil es una clase que permite trabajar la velocidad a la que es disparado el  
   elemento, así como el ángulo con el que sale disparado.

Se crea la clase proyectil, con los atributos correspondientes

A screenshot of a computer

Description automatically generated

Posterior, se genera el código en java

1. Una tienda (de la que guardamos el nombre, la dirección y el teléfono) ofrece 4 productos. De cada uno de estos productos queremos almacenar un código numérico, el nombre, el tipo del producto (que puede ser LACTEO, CÁRNICO, FRUTA o ENLATADO), la fecha de expiración, el nombre del fabricante, la cantidad en inventario y el precio unitario.

Se realiza la creación de las clases tienda, productos y fabricante con sus respectivos atributos.

A screenshot of a computer

Description automatically generated

1. Una Universidad (de la que nos interesa su nombre, nombre del rector y nombre de la  
   ciudad donde funciona) guarda información de sus facultades. Una facultad tiene un  
   nombre y un código numérico. Un profesor es una persona de la que tenemos que guardar también la profesión, la nacionalidad, y el sueldo. Un profesor pertenece a una sola facultad, pero una facultad tiene muchos profesores. Hay también carreras, y de una carrera necesitamos guardar el nombre, el número de créditos totales, el número de semestres que dura la carrera y el nivel de la carrera (PREGRADO o POSTGRADO). Una carrera pertenece a una sola facultad, pero una facultad tiene cero o más carreras. Los estudiantes son personas de las que tenemos que almacenar la siguiente información adicional: el colegio del cual se graduó del bachillerato y la fecha en que ingresó a la universidad. Un estudiante pertenece a 1 o más carreras, y claro, una carrera posee muchos estudiantes. Hay que guardar información de los cursos, y de cada curso hay que almacenar el código (un  
   número), el nombre del curso, el número de créditos, el número del salón donde se imparte y el nombre del edificio donde se dicta. Un curso tiene 1 o más profesores que lo dictan, y un profesor puede dictar muchos cursos también. En un curso pueden estar inscritos muchos estudiantes y un estudiante puede estar inscrito en muchos cursos. Los cursos pertenecen a una sola carrera, y la carrera tiene muchos cursos. Una persona tiene cédula, nombre, fecha de nacimiento y lugar de nacimiento (ciudad, departamento, país).

Para este ejercicio, se ingresa la clase universidad, con los atributos de nombre universidad, nombre de rector y nombre ciudad. A su vez, se ingresa la información de las operaciones que se pueden realizar (Insertar, Modificar, Eliminar, Consultar)

A screenshot of a computer

Description automatically generated

De igual manera, se ingresa la clase, facultad, con los atributos y las operaciones que puede realizar

A computer screen shot of a computer screen

Description automatically generated

Teniendo en cuenta ele enunciado, se crea la clase persona. Esta servirá para heredar los atributos a otras clases (profesor, estudiante.)

A screenshot of a computer

Description automatically generated

Se crea la clase, profesor y estudiante, la cual tienen herencia de la clase persona

A screenshot of a computer

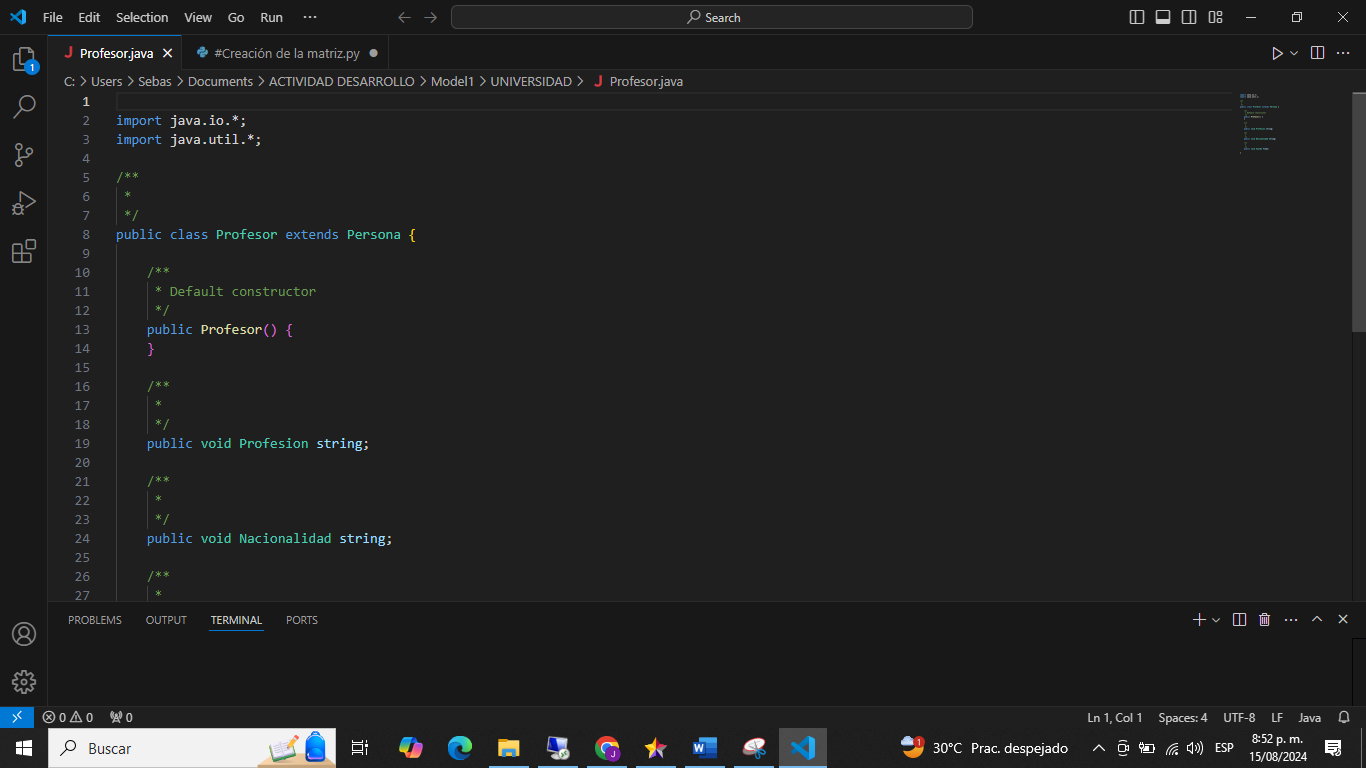
Description automatically generated

Dejando el siguiente diagrama de clases.

A screenshot of a computer

Description automatically generated

AL generar el código correspondiente y verificar los diferentes archivos se logra identificar las herencias. En la imagen adjunta se la clase profesor, se logra identificar la herencia como extends. Esto hacer referencia a la herencia que se definió para la clase profesor, la cual hereda los atributos de la clase persona. Esta herencia se genera dado que se configuró en el diagrama.



En conclusión, el diagrama UML mostrará las clases con sus atributos y cómo se relacionan entre sí. Las relaciones estarán marcadas con etiquetas que indican cuántos elementos pueden estar conectados (por ejemplo, 1 a muchos, muchos a muchos). Este diagrama, incluye, sus atributos y las líneas que indican las relaciones, con las etiquetas de multiplicidad y herencia cuando sea necesario

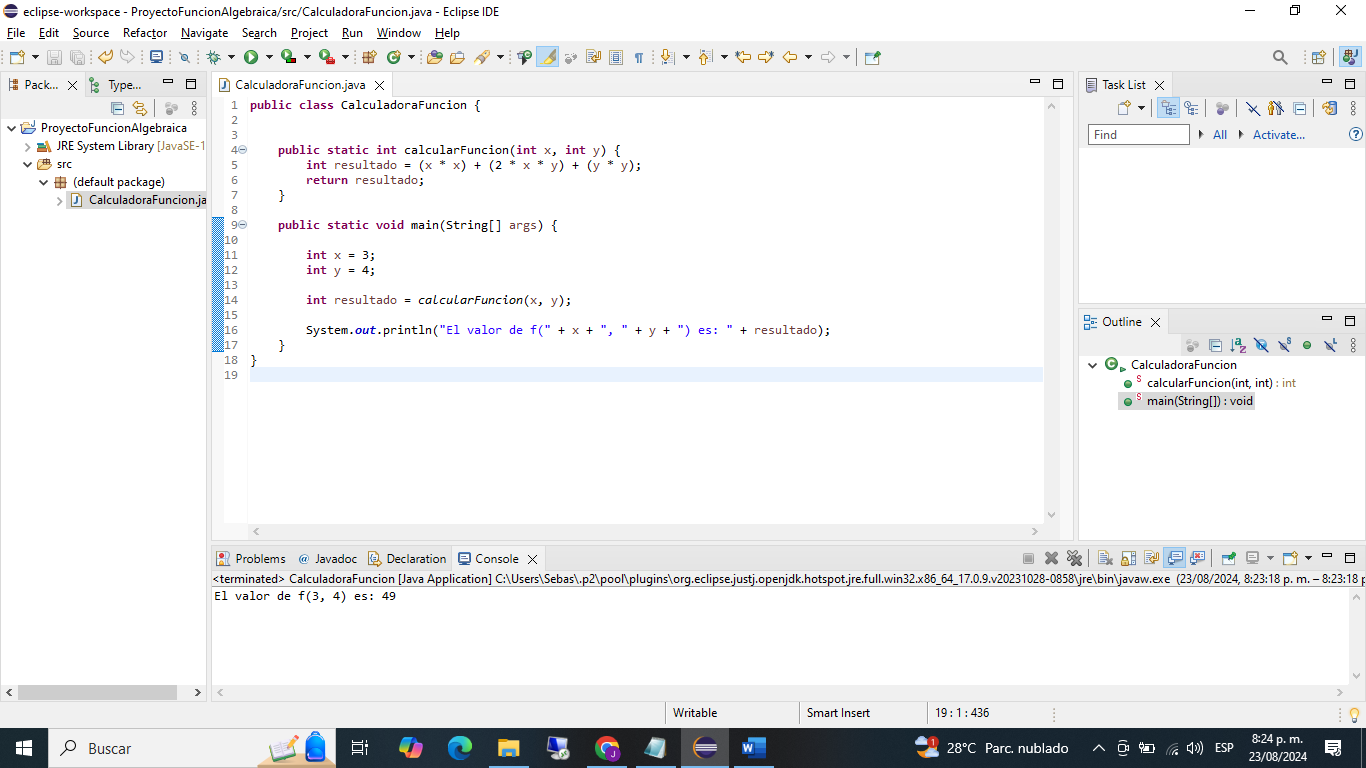
**Guía 2**

1. Escriba una función o método estático y luego un programa que use esa función, que encuentre el valor de la siguiente función algebraica:

En eclipse, se crea el nuevo proyecto denominado “ProyectoFucnionAlgebraica”.

Se crea la clase “CalculadoraFuncion” el cual es el método que recibe los parámetros para solucionar la ecuación.

El método main, se crea para llamar el método “calcular función” y por último el resultado se imprime en consola como se muestra en la imagen adjunta.



1. **El paseo: ya que finalizó la pandemia, vamos a organizar un paseo. Al paseo irán estudiantes gordos y flacos. Un estudiante gordo ocupa dos sillas de un bus y un estudiante flaco ocupa solo una silla. Si sabemos cuántas sillas tiene un bus, cuántos estudiantes gordos van al paseo y cuántos estudiantes flacos van al paseo, escriba una función y un programa en Java que encuentre el número de buses que se necesitan alquilar para llevar todos esos estudiantes al paseo.**

Para solucionar este problema se crea inicialmente el proyecto en Eclipse. Denominado “ProyectoPaseo”

Se crea la clase “CalculoBuses” la cual sirve para solucionar el problema planteado. En método “CalculoBuses” se ingresan 3 variables que permiten solucionar el problema planteado. Sillas por bus que permiten identificar la cantidad de sillas de cada bus. A su vez identificar los estudiantes flacos y gordos para identificar la cantidad de buses. Lo anterior se describe en el siguiente código

A screenshot of a computer

Description automatically generated

1. **Escriba una función que calcule la longitud de la escalera (𝑦) teniendo en cuenta que conocemos la altura que tiene la escalera (𝑥) cuando está inclinada sobre la pared con un ángulo α. Tenga en cuenta la siguiente figura:**

Se realiza la creación del proyecto en java “Escalera”. Se la clase “calculo escalera” el cual contendrá los métodos necesarios para solucionar y obtener la longitud de la escalera.

En el código se declaran los parámetros de entrada para realizar los cálculos respectivos. Adicionalmente, es necesario convertir el Angulo de la escalera de grados a radianes lo cual se realiza a través del método Math.toRadians. Ya con los datos ingresados se realiza el cálculo de la longitud de la escalera el cual se almacena en la variable “longitud escalera”. La explicación de lo mencionado anteriormente se detalla en el código de la imagen adjunta

A screenshot of a computer

Description automatically generated

1. Vamos a resolver el siguiente problema: Pedro acabó de recibir el sueldo mensual y se gasta el 40% de ese sueldo en el arriendo y el 15% del sueldo en comida, determine cuánto dinero se gastó en cada uno de esos dos ítems (en arriendo y en comida) y cuánto dinero le queda al final. Para ello: a. Escriba una función que reciba el sueldo de Pedro y obtenga y retorna el gasto en arriendo b. Escriba una función que reciba el sueldo de Pedro y obtenga y retorna el gasto en comida c. Escriba un programa que lea el sueldo de Pedro, y, usando las funciones anteriores muestre el gasto en arriendo, el gasto en comida y cuánto dinero le quedó al final a Pedro después del arriendo y la comida.

Se realiza la creación del nuevo proyecto.

Se realiza la creación de la clase para identificar los gastos de pedro. Al no saber cual es el salario de pedro se habilita la opción para poderlos capturar por consola. De esta manera poder hacer el cálculo e identificar los gastos individuales de pedro. Para esto se crea la calse GastosPedro en dónde se fijan los valores de arriendo y comida para realizar el cálculo automático como se observa en la siguiente imagen

A screenshot of a computer

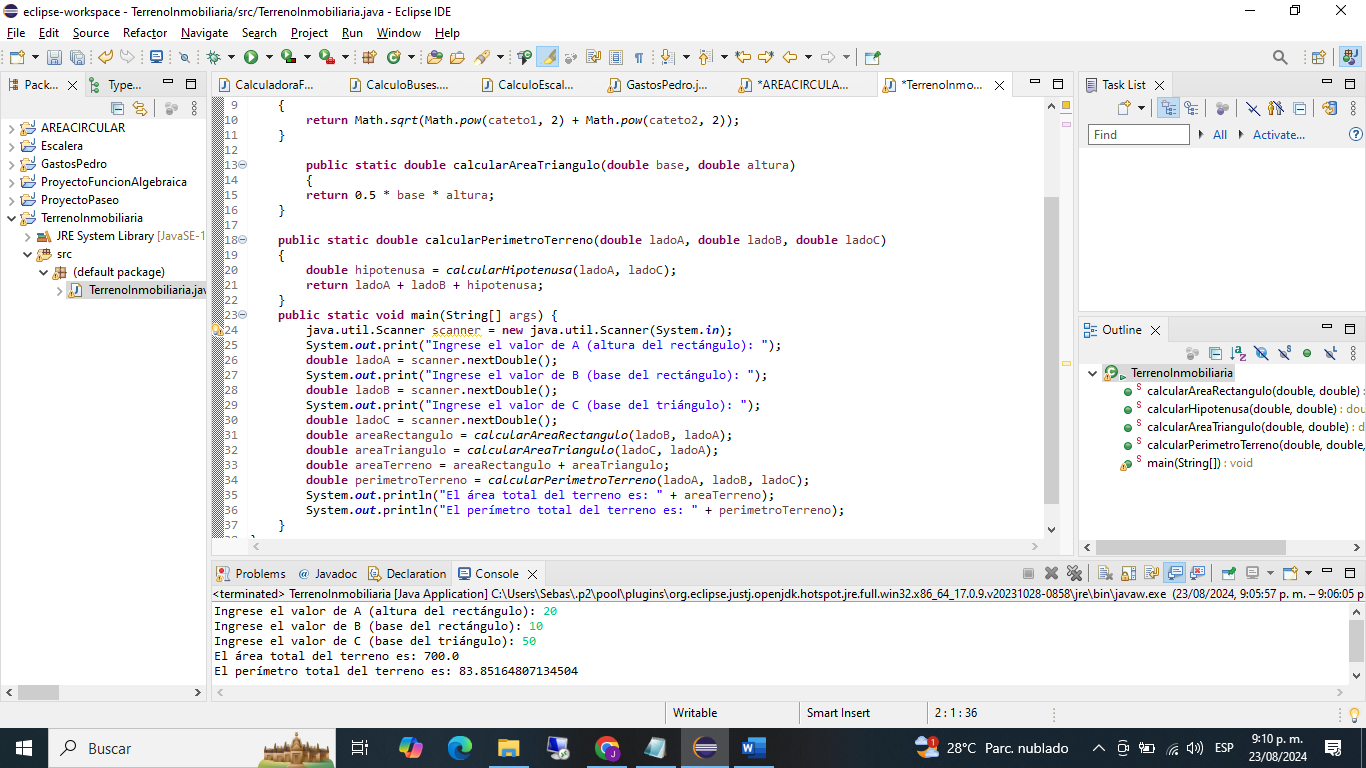
Description automatically generated

1. **Escriba una función que halle el área de un círculo de acuerdo con la siguiente fórmula: 𝐴 = 𝜋𝑟 2 . Ahora escriba otra función que halle el área de la corona circular (región en verde en la siguiente figura) a partir de la resta de las áreas del círculo grande (con radio R) y el círculo pequeño (con radio r). En esta segunda función debe usar la primera función. Además, escriba un programa que lea el valor de los dos radios (el radio pequeño r y el radio grande R) y, usando la función anterior, calcule y halle el área de la corona circular.**

A screenshot of a computer

Description automatically generated

1. Una empresa inmobiliaria vende terrenos con la forma a continuación:



1. **En el nuevo edificio de la universidad, cada piso tendrá el mismo número de salones y todos estos salones tendrán la misma capacidad (cantidad de estudiantes que caben en el salón). Escriba las siguientes funciones: a. La primera función que calcule el número de salones del nuevo edificio a partir del número de estudiantes que tiene la universidad y la capacidad de un salón. b. La segunda función obtendrá el número de pisos que tendrá el nuevo edificio a partir del número de salones que vamos a necesitar y el número de salones que tendrá un piso del edificio. c. Escriba finalmente el programa que lea desde el teclado el número de estudiantes que tiene la universidad, la capacidad de un salón y el número de salones que tendrá un piso, y qué usando las dos funciones anteriores, encuentre e imprima el número de salones y el número de pisos del nuevo edificio.**

A screenshot of a computer

Description automatically generated

1. **Escribir una función que reciba dos números enteros y retorne el mayor número de los dos. Escriba un programa, que lea dos números enteros, y usando la función anterior, imprima el mayor de los dos. No puede usar el método MAX de Java, use la instrucción condicional IF.**

A screenshot of a computer

Description automatically generated

**Conclusiones**

1. El uso de herramientas como Star UML permite una representación clara y organizada de los conceptos y relaciones dentro de un sistema. Al aplicar el conocimiento adquirido en la construcción de diagramas UML, los estudiantes son capaces de visualizar la estructura de un sistema, facilitando así su análisis, diseño y posterior implementación en código. Esta habilidad es esencial para abordar problemas complejos de manera estructurada. Solucionar diferentes situaciones mediante la programación orientada a objetos
2. La programación orientada a objetos (POO) es una metodología que permite descomponer un problema en componentes más manejables y reutilizables, conocidos como objetos. Al resolver situaciones mediante POO, los estudiantes desarrollan la capacidad de pensar en términos de clases, objetos, herencia y polimorfismo, lo cual es fundamental para diseñar sistemas robustos y escalables. Esta práctica refuerza la comprensión de cómo modelar situaciones del mundo real en código de manera eficiente.
3. La resolución de guías y ejercicios prácticos, como la guía I de la materia, permite a los estudiantes aplicar de manera directa los conceptos teóricos aprendidos. Este proceso de "aprender haciendo" es crucial para consolidar el conocimiento y desarrollar la habilidad de abordar problemas reales con soluciones basadas en principios de la programación orientada a objetos y modelado UML. Al completar estas guías, los estudiantes fortalecen su capacidad de integrar conocimientos teóricos con la práctica, mejorando su competencia en la materia.