## 4. Examinar las expresiones

Dado el siguiente código:

String a = "abc";

String s = a;

String t;

Indique qué valores retornan las siguientes expresiones o, si dan error, por qué se producen:

● s.length(); -> Retorna 3

● t.length(); -> Error, t indefinido

● 1 + a; -> Retorna 1abc

● a.toUpperCase(); -> Retorna ABC

● "Libertad".indexOf("r"); -> Retorna 4

● "Universidad".lastIndexOf('i'); -> Retorna 7

● "Quilmes".substring(2,4); -> Retorna “il”

● (a.length() + a).startsWith("a"); -> Retorna false

● s == a; -> Retorna true

● a.substring(1,3).equals("bc") -> Retorna true

## 5. Tipos primitivos

Teniendo en cuenta la documentación oficial de Java:

1. ¿Qué son los tipos de datos primitivos?

Son los tipos de datos más básicos que provee el lenguaje. Estos tipos no son objetos y se almacenan directamente en memoria, lo que los hace más eficientes para aplicaciones en donde la rápida performance es crítica. Según la documentación oficial existen 8 tipos de datos primitivos: byte, short, int, long, float, double, char y boolean.

2. ¿Cuál es la diferencia entre un int y un Integer?

int: Es un tipo de dato primitivo. Almacena un entero con signo de 32 bits.

Es más eficiente en memoria y rendimiento porque no es un objeto.

No puede ser null. Su valor predeterminado es 0.

Integer: Es una clase wrapper para el tipo primitivo int, y forma parte del paquete java.lang.

Permite usar valores “int” como objetos. Puede ser null.

Proporciona métodos útiles (por ejemplo, Integer.parseInt(), Integer.valueOf()).

Soporta características como autoboxing y unboxing (conversión automática entre int e Integer).

3. ¿Si se define una variable de instancia de tipo int cuál sería su valor predeterminado? ¿Y si se define una de tipo Integer? Haga la prueba en Eclipse.

Valor por default int = 0;

Valor por default Integer = null;

4. Responder la pregunta del punto anterior (3), pero ahora en lugar de definir una variable de instancia se define una variable de método.

Valor por default int = 0;

Valor por default Integer = null;

## 7. Jerarquía de paquetes

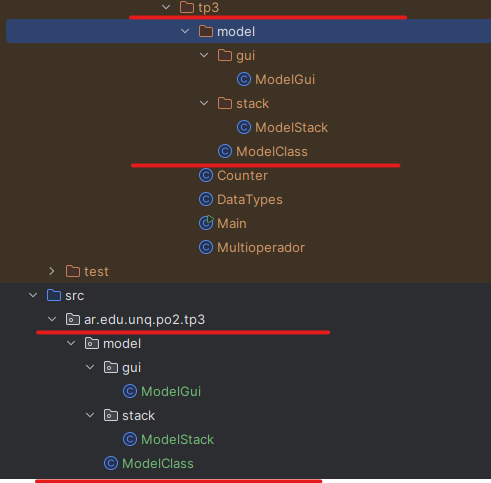
Cree una clase que se encuentre en el paquete **model**, otra clase que se encuentre en el paquete **model.gui** y otra que se encuentre en el paquete **model.stack** . Compílelas utilizando la forma que crea conveniente.

1. ¿Cómo están organizadas en el sistema de archivos?

Gui y Stack se crean como sub paquetes dentro de model

2. ¿Encuentra alguna relación entre el nombre del paquete y la ubicación de los archivos fuentes de las clases (.java) y los archivos compilados (.class)?

Se organizan de la misma forma en los archivos fuente que luego de compilarse.



## 10. Encapsulamiento

1. Responder: Si un objeto cualquiera que le pide la edad a una Persona: ¿Conoce cómo ésta calcula u obtiene tal valor? ¿Cómo se llama el mecanismo de abstracción que permite esto? No, el objeto que pide la edad no sabe cómo está implementado el método que calcula la edad, solo sabe que puede invocar el método público getAge() para obtener la edad y obtiene el resultado. El mecanismo de abstracción que permite esto es el encapsulamiento, este principio oculta los detalles internos de la implementación.