Ayudantía 6:
Tipos de Datos Abstractos y POO

Profesores: José Luis Martí Lara, Roberto Díaz Urra
Ayudantes:
Hugo Sepúlveda Arriaza
Gabriela Acuña Benito
Lucio Fondón Rebolledo
lucio.fondon@sansano.usm.cl

Universidad Técnica Federico Santa María
Departamento de Informática

Tipos de Datos Abstractos

Definición

Los tipos de datos abstractos (TDA) son estructuras de datos definidas por el usuario que son modificables mediantes métodos implementados para el propio funcionamiento del TDA. Cumplen específicamente con dos restricciones:

- **Ocultamiento de Información**: Todo el dato (*objeto*) se **debe** acceder mediante la *interfaz definida* para ellos, por lo que la implementación, la estructura y los componentes del dato están ocultos para el programa que lo usa.
- **Encapsulamiento**: El tipo, el objeto y sus métodos está contenido en una sola unidad.

Los TDA ofrecen: **Modularidad**, **Modificabilidad**, **Reusabilidad** y **Seguridad**.

Problemas que ataca la OO

- **Herencia (Inheritance)**: Permite extensión de datos u operaciones.

```
public class B extends A{...}
```

- **Redefinición (Overriding)**: A veces, queremos redefinir (y extender) operaciones de la clase para otros propósitos.
- **Abstracción**: Se requiere abstraer operaciones similares para diferentes componentes (Un auto y una moto tienen características similares pero funcionan de forma distinta.)
- **Polimorfismo:** Más de una forma en que el mismo objeto realiza diferentes operaciones según el requisito (parámetros).
  - Sobrecarga de Métodos (*overriding*)
  - Tipos Parametrizados

Modelo Básico

```
1  public class Stack { //clase Stack
2      public Stack(){
3          //Constructor
4      }
5
6      public void push(int num){
7          //Metodo
8      }
9
10     public void pop(){
11         //Metodo
12     }
13
14     public static void main(String[] args) {
15         Stack pila = new Stack(); //objeto
16         pila.push(3); //mensaje
17     }
18 }
```

- Programa
- Clases
- Objetos
- Métodos
- Mensaje

Conceptos Importantes

Clases y métodos abstractos

Un *método abstracto* es aquel método que solo se define, pero no se implementa. Una *clase abstracta* solo define una parte de su implementación. No se puede inicializar directamente.

```
abstract int area();
```

Interfaces

Un *interfaz* es una lista de acciones que puede llevar a cabo un determinado objeto. En una interfaz sólo existe el prototipo de una función, no su código.

```
public interface Nombre_Interfaz {
    void metodoUno();
    void metodoDos();
    /.../
}
```

Conceptos Importantes

Paquetes

Un *paquete* es un mecanismo que utiliza Java para facilitar la modularidad del código. Un paquete puede contener una o más definiciones de interfaces y clases, distribuyéndose habitualmente como un archivo.

```
package Nombre_Paquete;
import Nombre_Paquete;
```

Control de Acceso

Modificador	Clase	Paquete	Subclase	Global
`public`	✓	✓	✓	✓
`protected`	✓	✓	✓	✗
`default`	✓	✓	✗	✗
`private`	✓	✗	✗	✗

Ejercicios

Ejercicio 1

¿Cuál(es) de la(s) siguiente(s) afirmación(es) es(son) correcta(s)?

I. Si una clase es abstracta, entonces puede ser instanciada.
II. Si en una clase no define un constructor, se asume uno por defecto, sin argumentos.
III. En Java, es posible definir relaciones de herencia, tanto para clases como para interfaces.
IV. Un constructor, por tratarse de un método, debe retornar un valor, que corresponde al tipo de la referencia en cuestión.

a) Sólo II
b) I y II
c) I y IV
d) II y III

Ejercicio 1

¿Cuál(es) de la(s) siguiente(s) afirmación(es) es(son) correcta(s)?

I. Si una clase es abstracta, entonces puede ser instanciada.
II. Si en una clase no define un constructor, se asume uno por defecto, sin argumentos.
III. En Java, es posible definir relaciones de herencia, tanto para clases como para interfaces.
IV. Un constructor, por tratarse de un método, debe retornar un valor, que corresponde al tipo de la referencia en cuestión.

a) Sólo II
b) I y II
c) I y IV
d) II y III

R: **Alternativa d**

Ejercicio 2 (C2 2018-1)

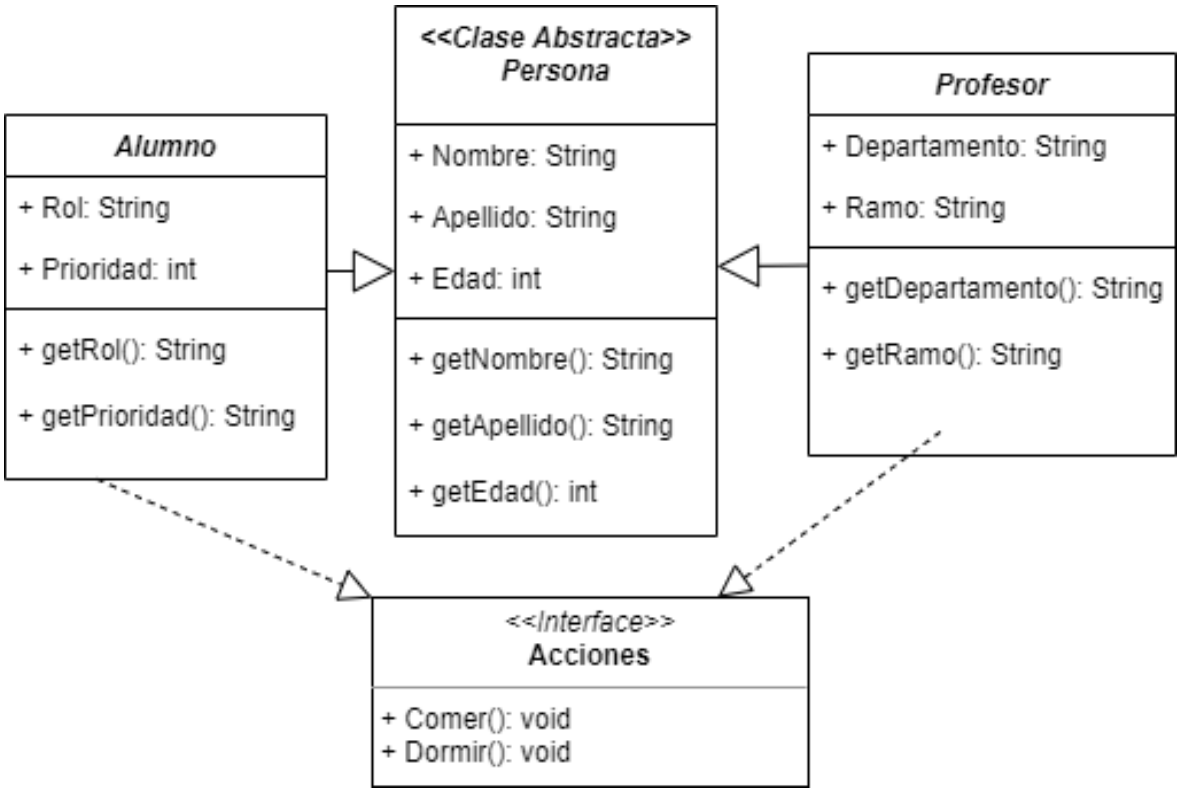
El elemento geométrico que queda definido por dos puntos es la línea, pues existe sólo una que conecte dichos dos puntos. Operaciones posibles de definir para manipular cualquier objeto de tipo línea son:

- Crear una línea, a partir de las coordenadas de los dos puntos que la definen.
- Trasladar una línea hacia arriba o abajo, una distancia dada.
- Trasladar una línea hacia la izquierda o derecha, una distancia dada.
- Cambiar cualquiera de las coordenadas de los dos puntos asociados.

Usando Java, construir una clase Linea, que se ajuste a la especificación anterior.

Ejercicio 3

Dado el siguiente modelo de clases:



construir las clases e interfaces correspondientes en Java.