

TAD

...

Tipo Abstracto de Dato

Tipo Abstracto de Datos (TAD):

- Es un conjunto de valores y de operaciones definidos mediante una especificación independiente de cualquier representación.
- La manipulación de un TAD sólo depende de su especificación, nunca de su implementación.

QUÉ HACE

vs

CÓMO LO HACE

Introducción - Ejemplo Simple

- Abrir
- Cerrar



Puerta - lado usuario - Primitivas



```
type puerta struct;
```

```
abrir(puerta)
```



```
cerrar(puerta)
```



```
estaAbierta(puerta)
```



¿ Cómo está implementado adentro puerta ?



Usar el
algoritmo
conociendo la
implementacion



Ignorar
como esta
implementado

Lo importante de TDA

- Solamente se trabaja con las **primitivas** (funciones que expone) del TDA.
- Nunca se tiene que programar código que dependa de la implementación interna de un TDA.

Lo importante de TDA



Lo importante de un TDA



Puerta - lado implementación

```
type Puerta struct {  
    estado int;  
}  
  
func (puerta *Puerta) cerrar () {  
    puerta.estado = 0;  
}
```

```
type Puerta struct {  
    estado bool;  
}  
  
func (puerta *Puerta) cerrar() {  
    puerta.estado = false;  
}
```

Invariantes de las primitivas

- Sirven para especificar al TDA y son parte de la descripción de una primitiva.
- Componen las precondiciones y postcondiciones de cada primitiva.
- Ej: una invariante de la primitiva abrir()
“luego de aplicar abrir(), el estado de la puerta será abierta”

Presentando a nuestros avatares - Primitivas

ALAN-> implementa TAD



BARBARA-> utiliza TAD



FIN

...

Tipo Abstracto de Dato