Tipos Abstractos de Datos:

* Ignorar los detalles
* Posponer la consideración de detalles.

Programación:

* Los programas se expresan en lenguaje de alto nivel → Compilador
* Si observamos que se ejecuten se tienen que expresar en código de máquina

Abstraccion Procedural:

* Subprogramas(Procedimientos / Funcion)

Abstracción de Datos:

* TAD(Tipo Abstracto de Datos)

¿Qué es un TAD?

Es un tipo de dato que te permite usar el lenguaje(int, char, etc)

Tipos de datos:

* Elementales (Int, char, float, bool, etc)
* Estructuras ([ ], enum, estruct, dinámicas)

Coso se Introduce un TAD:

* Dándole nombre y asociarlo operaciones(Procedimientos / funciones)

Especificación (Definición) del TAD.h

* Operaciones -Constructores

(Cabezal y comentario con pre y post condiciones)

Implementación de TAD.c:

* Representación del TAD(Estructura)
* Operaciones (implementaciones)

racional crear(){

racional r;

r=new(nodo\_racional);

return r;

}

racional destruir(racional r){

delete r;

}

Conclusión:

* Permite especificar TADs Durante el proceso de Diseño de programas
* Provee un método adecuado de abstracción y refinamiento
* Los programas se mantienen independientes de la implementación de los Tads
* Las representaciones son OPACAS(Invisibles) lo que garantiza que:

- Los programas manipulan objetos del TAD SÓLO a través de las funciones

- Los programas no deben modificarse si la implementación del TAD cambia

Ventajas:

* Modulidad
* Separación entre especificaciones e implementaciones hace que los programas sean:

1. legibles
2. más fáciles de matar
3. más fáciles de verificar y probar
4. más fáciles de reusar
5. la independiza de distinto implementaciones