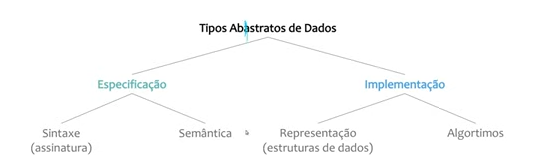
# O que são?

* Implementações e funcionalidades -> você sabe o que faz e não como faz;
* Ou até, ter especificações de o quais são a funcionalidades que precisam ser implementadas e, então, cada um faz elas como quer;

De forma geral, o TAD visa dividir o código em especificações (o que ele faz) e a implementação (como faz);

Visa desvincular o **tipo de dado** (**estrutura de dados** e  **operações** que as manipulam) de sua **implementação**;

* Ideia parecida com Encapsulamento em Orientação a Objetos; -> Escondemos os dados e detalhes do usuário, fornecendo apenas uma **interface pública** para manipulá-los.



*Exemplo*

*sintaxe:* int Soma (int a, int b);

*semântica:* Documentação:: Comentários ou lugares que explicam como as funções são

usadas, etc;

*Algoritmos:*

int Soma(int a, int b){

return a+b;

}

*Representação:*

* **Arquivo de cabeçalho (.h) -> especificação**

seu\_tad.h

* **Arquivo fonte (.c) -> implementação**

seu\_tas,c

Os programas ou outras TADs que utiliuzam seuTAD devem incluir sua especificação:

**#include “seu\_tad.h”** -> Ou seja, inclui somente as especificações

### Vantagens

* reutilização:
* facilidade de manutenção: mudanças na implementação do TAD não afetam o código fonte dos programas que o utilizam
* corretude: códigos testados em diferentes contextos

# Implementação:

