

Types Abstraits de Données et Collections

Algo – Chapitre 4

I. Définition d'un TAD

Un Type Abstrait de Données (TAD) est la **définition mathématique d'un type de données**. C'est un ensemble d'éléments et d'opérations associées (lci, lce, ...) Il est défini par les éléments suivants :

Nom :	<i>Nom du TAD</i>
Paramètre :	<i>Pour les TAD collections, le nom générique du type des éléments stockés</i>
Utilise :	<i>Types utilisés</i>
Opérations :	$\text{nomOp}_1 : \text{Type}_{e1} \times \text{Type}_{e2} \times \dots \rightarrow \text{Type}_{s1} \times \text{Type}_{s2} \times \dots$ ⋮ <i>Contient des opérations d'accès à l'information (constantes), sans entrée.</i>
Axiomes/Sémantique :	<i>Axiome : décrit logiquement une composition d'opérations</i> <i>Sémantique : explique ce que fait chaque opération</i>
Préconditions :	$\text{nomOp}_1 : \text{précondition de l'opération}$ ⋮

II. Etapes de création d'un TAD

- **Analyse** : Identifier et définir les TAD du programme
 - Complétude (pas d'op° manquantes)
 - Consistance (non contradictoire)
 - Identifiants significatifs
 - Préconditions
- **Conception préliminaire** : Déclarer les fonctions et procédures des opérations du TAD

Conception détaillée : Choisir une manière de représenter les entités du TAD, écrire algos

Exemple :

Nom :	Réel
Utilise :	Booléen
Opérations :	$[0-9]^+.[0-9]^+ :$ → Réel
+	Réel × Réel → Réel
-	Réel × Réel → Réel
*	Réel × Réel → Réel
/	Réel × Réel → Réel
<	Réel × Réel → Booléen
	⋮
Axiomes :	...
Préconditions :	$a / b : b \neq 0$