2 - Algebra

Astrazione funzionale

L'astrazione funzionale è la tecnica che permette di potenziare il linguaggio disponibile introducendo nuovi operatori, definiti attraverso sotto-programmi in cui si fa uso degli operatori di base già disponibili

I linguaggi di programmazione ad alto livello permettono l'uso di astrazione funzionale, cioè consentono di:

- creare delle unità di programma dando un nome ad un gruppo di istruzioni e stabiliscono le modalità di comunicazione tra l'unità di programma creata ed il resto del programma in cui essa si inserisce
- al momento della attivazione (su chiamata) della unità di programma viene sospesa l'esecuzione del programma (o dell'unità) chiamante, passando il controllo all'unità attivata, terminando poi e tornando al programma chiamante

I costrutti linguistici per realizzare l'astrazione funzionale consentono di definire una **specifica** e una **realizzazione**:

- La **specifica** definisce "cosa" ci si aspetta dalla funzione quindi la relazione che caratterizza il legame tra dati di ingresso e risultati
- La **realizzazione** definisce "come" il risultato viene ottenuto, ad una stessa specifica ci possono essere più realizzazioni

Astrazione dati

L'**astrazione di dati** permette di ampliare i tipi di dati disponibili attraverso l'introduzione sia di nuovi tipi di dati che di nuovi **operatori**.

Un'astrazione di dati consente un'estensione dell'algebra dei dati disponibile in un linguaggio di programmazione.
Un'algebra (in informatica) è formata da:

- un **insieme di valori** (dominio),
- un **insieme di operazioni** su quei valori.

Un **tipo astratto di dato (ADT)** è esattamente questo: un insieme di valori e le operazioni che ci possiamo fare, senza preoccuparci di come sono implementati.

Per estendere l'algebra dei dati disponibile in un linguaggio di programmazione, molti linguaggi forniscono i mezzi per definire e creare nuovi tipi di dati, ma non tutti consentono di fare astrazione dati o creare tipi di dati astratti.

Un tipo di dati si dice astratto se le operazioni applicabili sugli oggetti che li rappresentano sono isolate dai dettagli usati per realizzare il **tipo**.

Requisiti dell'astrazione dati

Disponendo di un linguaggio con tipi, possiamo utilizzare dati che definiamo e dichiariamo direttamente come complessi, prescindendo dalla effettiva realizzazione, e fare in modo che alle procedure (operatori) costruite per i nostri dati abbiano accesso esclusivamente quei dati. Queste caratteristiche necessarie a una buona astrazione di dati sono note come i requisiti della astrazione di dati:

1. **Requisito di astrazione**: i programmi che usano i nuovi tipi devono dipendere solo dalla loro **definizione**, non da come sono realizzati.
   In sostanza è verificato quando i programmi che usano un'astrazione possono essere scritti in modo da non dipendere dalle scelte di realizzazione
2. **Requisito di protezione**: i nuovi operatori devono poter essere usati **solo** sui nuovi tipi, e non su dati che “assomigliano” ma non sono dello stesso tipo.
   La mancanza del requisito di protezione si manifesta con la possibilità di lavorare con gli operatori definiti per i nuovi dati anche su dati con rappresentazioni simili, ma non omogenei per tipo, creando quindi errori

Specifica di realizzazione

Un tipo astratto si definisce in due fasi:

1. **Specifica**: cosa rappresenta e quali operatori ha (livello “matematico”), si suddivide in:
   - **Sintattica**: fornisce l'elenco dei nomi dei tipi di dato utilizzati per definire la struttura, delle operazioni specifiche della struttura e delle costanti, nonché i domini di partenza e di arrivo, cioè i tipi degli operandi e del risultato per ogni nome di operatore.
   - **Semantica**: definisce il significato dei nomi introdotti con la specifica sintattica. Associa un insieme ad ogni nome di tipo introdotto nella specifica sintattica, un valore ad ogni costante, e una funzione ad ogni nome di operatore esplicitando le seguenti condizioni sui domini di partenza e di arrivo:
     - La **pre-condizione** che definisce quando l'operatore è applicabile
     - La **post-condizione** che stabilisce come il risultato sia vincolato agli argomenti dell'operatore.
2. **Realizzazione**: come implementare praticamente la specifica nel linguaggio scelto. Mentre la definizione di un formalismo per le specifiche sintattiche non pone particolari problemi, l'individuazione di formalismi per le specifiche semantiche è assai più difficile, e in genere le specifiche semantiche sono date con frasi estratte dal linguaggio naturale o da quello matematico.

Struttura di dati

Tra i tipi di dato possiamo individuarne particolari, chiamate **strutture di dati**, che tendono a rappresentare una collezione di dati che la compongono dal tipo e dall'organizzazione degli elementi componenti.
Nei linguaggi si trovano solo tabelle monodimensionali (vettori o array)

In generale si distinguono in:

- **Lineari**: i suoi elementi sono in una sequenza, con un certo ordine
- **Non lineari**: non esiste una sequenza
- **A dimensione fissa**: il numero di elementi non varia nel tempo
- **A dimensione variabile**: il cui numero di elementi varia nel tempo
- **Omogenee**
- **Non omogenee**

Riprendendo l'esempio degli array (o dei vettori), è una struttura a dimensione fissa, con una tabella monodimensionale di elementi omogenei su cui si possono effettuare:

- Lettura (o selezione)
- Scrittura (o sostituzione)
  La sua organizzazione di memoria consente l'accesso diretto al singolo componente tramite l'indice (che viene calcolato con l'indirizzo di memoria, tornare indietro o avanti significa moltiplicare quella cella di memoria n volte per quanto vogliamo andare avanti o indietro)

Per esempi guardare il pacchetto di slide n.2