LEONARDO STOPPANI - 580486
PR2 - Corso A

RELAZIONE SECONDO PROGETTO
DIZIONARI

UNIVERSITÀ DI PISA

Introduzione

Il progetto consiste nel realizzare un estensione del linguaggio didattico funzionale, definito a lezione, che permetta di manipolare Dizionari. Questa relazione ha come obiettivo quello di fare chiarezza sulle scelte di implementazione prese dallo studente nello svolgimento del progetto.

Scelte di Implementazione

Nel mio progetto ho deciso di caratterizzare un Dizionario con alcune scelte di implementazione :

- Un Dizionario deve essere omogeneo, ovvero non sono ammessi valori con tipi diversi in uno stesso dizionario
- Un Dizionario può contenere valori di 2 tipi : Interi e Booleani
- In generale è stata adottata una nozione di programmazione difensiva, ovvero è stato preferito limitare le operazioni eseguibili sul dizionario (condizionate dai tipi ammissibili) per evitare di generare errori non controllabili e confusionari.

Type Checker Dinamico

Il type checker dinamico che ho implementato controlla che un dizionario sia omogeneo e abbia valori di tipo ammissibile ogni volta che viene creato con l'operatore Dict() o passato come argomento alle operazioni primitive. In caso contrario viene restituito un messaggio di errore tramite failwith.

Comportamento Atteso

In base al type checking dinamico le operazioni Dict() e Insert() restituiscono il messaggio di errore "Type error : dict must be homogeneous!" se si sta provando, nel primo caso a creare un dizionario con valori di tipi diversi, nel secondo a inserire un elemento con valore di tipo diverso da quello del dizionario; inoltre generano il messaggio di errore "Invalid dict value" se si prova a creare un dizionario con elementi di valori di tipo non ammissibili (diversi da int e bool) o se prova a inserire un elemento di valore di tipo non ammissibile. Tutte le nuove operazioni di creazione e manipolazione di dizionari restituiscono il messaggio di errore "This operation must be applied to a dict type" se vengono applicate ad un argomento diverso da un dizionario; non e' necessario controllare che il dizionario passato alle operazioni sia omogeneo in quanto lo stiamo gia' garantendo al momento della creazione. La Delete() nel caso in cui non esista nel dizionario l'elemento da rimuovere semplicemente non fa nulla, quello che si aspetta chi usa questa operazione e' che dopo la chiamata nel dizionario non esista piu' l'elemento, non mi sembra necessario lanciare un errore. Per Implementare la Fold() ho prima esteso il linguaggio con le funzioni a due argomenti che in questo caso vengono usate come funzioni con accumulatore. Le operazioni FunAcc() e FunAccCall() non permettono la definizione e applicazione di funzioni ricorsive per evitare di complicarle troppo dato che nella fold non sarebbero state utilizzate. Il concetto di accumulatore viene gestito dalla Fold() che controlla prima se la funzione passatagli come argomento sia una funzione con accumulatore altrimenti genera il messaggio di errore "Not a fun with acc"; poi controlla se la funzione e' ammissibile per il tipo del dizionario, ovvero somma, differenza e prodotto per gli interi, and e or per i booleani, altrimenti genera il messaggio di errore "Unsupported function with fold". Il cliente sa che dovrebbe usare la fold con una funzione di questo tipo, quello che non deve sapere e' come viene gestito l'accumulatore il quale infatti viene inizializzato automaticamente con l'elemento neutro dell'operazione.