Relazione Roberto Lorenzetto (n° 531747) progetto 2

Type exp

Prendendo come riferimento l'interprete visto a lezione, in type exp ho aggiunto tutti i costruttori di tipo che mi servivano per implementare il dizionario.
Ho aggiunto:
il costruttore EmptyDict che rappresenta il dizionario vuoto, Insert di tipo string * exp * exp, Delete ed Has_Key di tipo string * exp, Iterate, Fold e Filter di tipo exp * exp.

Inoltre ho aggiunto tutti i costruttori per creare il tipo lista necessario per la funzione filter, in particolare:
EmptyKeyList che rappresenta la lista di chiavi vuota e AddKey string * exp per aggiungere una chiave alla lista

Tipi esprimibili

Ad evT ho dovuto aggiungere, in coerenza con la notazione dell'interprete: Dict of dict e KeyList of keylist.
In and con tutto il resto, perché si chiamano a vicenda, ho aggiunto dict (dictonary) che può essere Nil (vuoto) oppure Cons of string * evT * exp. Quindi l'ho definito in modo semplice ricorsivo.
Inoltre ci sono i tipi esprimibili per la lista che può essere KeyNil oppure KeyCons of string * keylist.

Funzioni primitive

Quasi tutte le funzioni primitive che ho aggiunto sono implementate con una funzione semplice che incapsula una funzione ricorsiva che svolge il lavoro vero e proprio. La funzione semplice esiste perché avevo bisogno di estrapolare dal tipo evT, il tipo effettivo sul quale poi potevo applicare la ricorsione. Oltre a ciò la funzione semplice fa anche un type check per assicurasi la correttezza della chiamata. In questo modo ho implementato il type checking dinamico.

Interprete

Per aggiungere le funzioni fold e iterate ho dovuto mettere la funzione funcall in and con la eval. Tutto questo perché avevo bisogno di chiamare la eval anche dentro funcall. Allo stesso modo ho messo in and con la eval anche la iterate e la fold per lo stesso motivo. Ovvero entrambe chiamano la funcall che a sua volta chiama la eval.

Test

La batteria di test è semplicissima ed è esattamente identica all'esempio mostrato nella consegna del progetto.