PROGRAMMAZIONE II – A.A. 2020-21: Secondo Progetto Intermedio

[Consegna: 23 12 2019, 12 01 2020, 02 02 2020]

Descrizione: Progettazione e sviluppo di un interprete in OCaml

Il progetto prevede la progettazione e realizzazione di una semplice estensione del linguaggio didattico funzionale presentato a lezione che permetta di manipolare *insiemi*. Un insieme Set è una collezione di valori omogenei, non ordinati, che non contiene valori duplicati.

Un insieme può essere creato a partire dall'insieme vuoto:

let s = empty(t)
(* risultato è l'insieme vuoto di tipo t *)

dove t è il tipo degli elementi dell'insieme.

Alternativamente può essere creato attraverso un valore:

let s = singleton("hello", string)
(* risultato è un insieme di stringhe contenente la stringa "hello" *)

Gli insiemi sono caratterizzati da numerose operazioni di base (tra cui unione, intersezione e differenza). Ipotizziamo inoltre di avere altre operazioni che permettono:

1. inserire (risp. rimuovere) un elemento da un insieme,
2. controllare se un insieme è vuoto,
3. controllare se un elemento appartiene ad un insieme,
4. controllare se un insieme è sottoinsieme di un altro insieme,
5. determinare il minimo (risp. massimo) valore di un insieme.

Oltre a questo insieme di operazioni di base, il tipo insieme prevede un insieme di operatori di natura "funzionale".

- For_all(predicate, aSet) controlla se tutti gli elementi dell'insieme aSet soddisfano la proprietà definita dal parametro "predicate". Il paramentro "predicate" è una funzione che applicata ad un elemento dell'insieme restituisce un valore booleano.

- Exists(predicate,aSet) controlla se esiste almeno un elemento dell'insieme aSet che soddisfa la proprietà definita dal parametro "predicate".

- Filter(predicate, aSet) restituisce l'insieme degli elementi dell'insieme aSet che soddisfano la proprietà definita dal parametro "predicate".

- Map(function, aSet) restitusce l'insieme dei valori v tali che v = function(x) per x appartenente a aSet.

1. Definire le regole operazionali per l'introduzione del tipo di dato Set nel linguaggio didattico.
2. Definire le regole operazionali per le classi di operazione previste dal tipo di dato Set.
3. Estendere l'interprete OCaml del linguaggio funzionale assumendo scoping statico.
4. Definire il type checker dinamico del linguaggio risultante.
5. Si verifichi la correttezza dell'interprete progettando ed eseguendo una quantità di casi di test sufficiente a testare tutti gli operatori aggiuntivi.

Modalità di consegna

- Il progetto deve essere svolto e discusso col docente individualmente. Il confronto con colleghi mirante a valutare soluzioni alternative durante la fase di progetto è incoraggiato.
- Il progetto deve essere costituito da
    - i file sorgente contenenti il codice sviluppato e le corrispondenti batterie di test, ove tutto il codice deve essere adeguatamente commentato;
    - una relazione di massimo una pagina che descrive le principali scelte progettuali ed eventuali istruzioni per eseguire il codice.
- La consegna va fatta inviando per email tutti i file in un archivio entro le date previste. Per il corso A, inviare l'email al Prof. Ferrari con oggetto "[PR2A] Consegna progetto 2". Per il corso B, inviare l'email al Prof.ssa Levi con oggetto contenente la stringa "[PR2B] Consegna progetto 2".

Altre informazioni

- Per quanto riguarda il progetto, i docenti risponderanno solo a eventuali domande riguardanti l'interpretazione del testo, e non commenteranno soluzioni parziali prima della consegna.