**PROGRAMMAZIONE II – A.A. 2016-17 – Secondo Progetto v.1**

Telleschi Marco

504382

Corso B

Come da richiesta si realizza un interprete Ocaml di un estensione del linguaggio funzionale presentato a lezione, che manipoli tuple di espressioni, e che permetta di combinare opportunamente funzioni.

Durante la fase di progetto è stato deciso di modificare leggermente la sintassi astratta suggerita come segue:

*type exp = …*

*| Etup of tuple*

*| Pipe of tuple \* exp*

*| ManyTimes of int \* exp \* exp*

*and tuple = Nil*

*| Seq of exp \* tuple*

Valutando l’exp Etup of tuple l’interprete restituisce un’altra etup nella quale le exp presenti all’interno della Etup iniziale sono state valutate ricorsivamente dall’interprete.

Pipe è considerata come una tupla comprendente funzioni. La tupla contiene le funzioni e la exp rappresenta il paramentro da passare alla prima funzione. L’interprete valutando Pipe of tuple \* exp restituisce quindi il risultato derivante dall’applicazione in sequenza delle suddette funzioni.

ManyTimes è stata considerata come l’applicazione in sequenza della stessa funzione un numero fissato di volte. Int rappresenta il numero di volte che la funzione deve essere applicata, la prima exp la funzione e la seconda il parametro da passare alla prima applicazione della funzione.

E’ stato aggiunto un type checker dinamico che viene utilizzato durante l’applicazione delle operazioni di base per controllare il tipo dei valori sui quali si applica l’operazione.

Per verificare la correttezza dell’interprete sono presenti opportuni test che valutano tutti gli operatori aggiuntivi.

Si fornisce il file .ml contenente l’interprete e i corrispondenti test.

Il codice è stato sviluppato è stato scritto utilizzando l’editor di testo Sublime Text ed è stato compilato ed eseguito sia da terminale che utilizzando Try Ocaml (<https://try.ocamlpro.com/>) su una macchina OS Ubuntu a 64 bit.