Laboratorio

Sommario: 30 Marzo, 2017

- Sviluppo del Progetto.
- Commenti ed Esercizi.

Dichiarazioni di SmallC in Ocaml

```
(* The Token, i.e. Lexical Categories *)

type ide = string;;
type num = int;;


(* The Syntactic Categories *)

type dcl =
   Const of ide * num
   | Var of ide * num
   | Array of ide * num
   | SeqDcl of dcl * dcl
   | EmptyDcl
   ;;
```

Commenti

- **Costrutto: Tipi Alegbrici o Concreti**.
  - Nella slide precedente abbiamo iniziato ad implementare in Ocaml un interprete per SmallC.
  - L'implementazione come prima cosa provvede a definire una *rappresentazione* in Ocaml per gli alberi di Sintassi Astratta dei programmi di SmallC.
  - La rappresentazione in Ocaml usa un costrutto (quello dei Tipi Algebrici) che è "sorprendentemente" vicino al formalismo utilizzato per esprimere la sintassi astratta.
- **La rappresentazione**
  - Ad ogni categoria lessicale o sintattica di SmallC facciamo corrispondere un nuovo tipo e una distinta classe di valori costruiti da specifici *costruttori di valori*.
  - Al momento abbiamo introdotto i tipi `ide`, `num`, `dcl` per le categorie `Ide`, `Num`, `Dcl`, rispettivamente.
  - Prossimamente provvederemo alle rimanenti fino alla categoria `Prog`

Commenti ed Esercizi

Esercizio (Rappresentazione dei Token)

Il token degli identificatori, `Ide` *è stato definito con:*

```
type ide = string;;
```

(a) Avremmo potuto definirlo con la definizione seguente?

```
type ide = Ide string;;
```

(b) Che differenza troviamo tra le due definizioni e quali differenti conseguenze?

Esercizio (Una dichiarazione)

Utilizzando le definizione date in Ocaml, si mostri un codice Ocaml per esprimere l'albero astratto relativo alla seguente dichiarazione. Attenzione: La dichiarazione che diamo sotto è in Sintassi Concreta di SmallC.

```
var x = 7; max = 100; var y;
```

Esercizio (Rappresentazione in C)

Supponiamo di dover usare il linguaggio C in luogo di Ocaml.

(a) Forniamo una rappresentazione in C per `Ide, Num, Dcl`

(b) Cosa possiamo dire confrontando i differenti codici Ocaml e C?