29 giugno 2004

# Linguaggi di Programmazione

| | |
|---|---|
| *Nome e Cognome* | |
| *Matricola* | |
| *Anno di corso* | |
| *Telefono* | |

1. Specificare la grammatica BNF di un linguaggio in cui ogni frase corrisponde alla specifica di un package che incapsula la definizione di tipi e variabili, come nel seguente esempio:

```
package P is
  type
    T1, T2: record(a: int, b: real);
    T3: array [1..10] of string;
    T4, T5, T6: T1;
    T7: array [2..20] of array [3..6] of T3;
  var
    i, j: int;
    x, y, z: T2;
    v, w : record(s: string, v: array [1..100] of T7);
 end P
```

Il package ha un nome e due sezioni, una per i tipi e l'altra per le variabili. Le sezioni non possono essere vuote. I tipi semplici primitivi sono **`int`**, **`real`** e **`string`**. Esistono due costruttori (ortogonali) di tipo: il **`record`** e l'**`array`**. Per quest'ultimo, è necessario specificare il range dell'indice mediante due costanti intere. La sezione dei tipi permette di associare uno o più nomi ad una certa struttura dati. Tali nomi possono quindi comparire in dichiarazioni di tipi e/o variabili.

2. Assumendo la seguente tabella di operatori (in cui ogni operatore binario valuta prima l'operando di sinistra e poi quello di destra, e la valutazione delle espressioni logiche è in corto circuito),

| *Operatori* | *Associatività* |
|---|---|
| *, / | destra |
| +, - | sinistra |
| <, > | nonassoc |
| **and**, **or** | sinistra |

specificare la semantica operazionale del seguente assegnamento (mediante espressione condizionale) della variabile logica `x`, in cui l'espressione condizionale è valutata da sinistra a destra:

```
x := (a * b / c + 3 * d > e + 1 / f or y ? z : p() and q())
```

NB: Il linguaggio di specifica operazionale contempla le seguenti limitazioni:

- Non contiene gli operatori logici;
- Contiene gli operatori +, -, *, /, <, >, che però possono essere applicati solo a costanti o variabili.

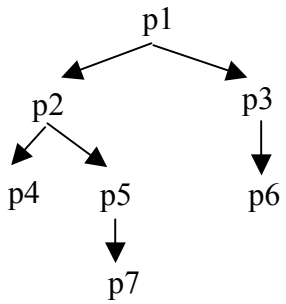
3. Definire nel linguaggio *Haskell* le seguenti due strutture tabellari

```
Studente(Matricola, Anno, Corso)
Docente(Nome, Corso)
```

Quindi, codificare la funzione `docenti_dello_studente` che, ricevendo in ingresso le tabelle `stud` e `doc` e la matricola `mat` di uno studente, computa la lista dei nomi dei docenti relativi ai corsi di tale studente.

4. Data una base di fatti *Prolog* relativa alla descrizione di chiamate di procedure in un linguaggio imperativo, come nel seguente esempio,

```
calls(p1, p2).
calls(p1, p3).
calls(p2, p4).
calls(p2, p5).
calls(p3, p6).
calls(p5, p7).
```




definire il predicato `activates(P, Q)`, in cui la procedura `Q` è (direttamente o indirettamente) chiamata dalla procedura `P`.

5. Discutere il concetto di ortogonalità nei linguaggi di programmazione, mettendo in evidenza vantaggi e svantaggi.

6. Enunciare ed illustrare i cinque modelli linguistici di passaggio dei parametri nei sottoprogrammi.