11 dicembre 2007

Linguaggi di Programmazione

Nome e Cognome	
Corso di laurea	
Telefono	
Email	

1. Specificare la grammatica BNF di un linguaggio di markup (tipo *HTML*) per la specifica di tabelle. Ogni frase del linguaggio definisce una o più tabelle. Ogni tabella ha almeno una colonna. Inoltre, ogni tabella ha necessariamente una prima riga, che specifica l'intestazione, e una serie (eventualmente vuota) di righe successive che specificano il contenuto. In ogni cella relativa a ciascuna riga è inserito un dato. Un dato può essere un identificatore, un numero o un'altra tabella. Nel caso di intestazione, il dato di ogni cella è necessariamente un identificatore. Tuttavia, un dato può essere omesso (in tal caso, la cella risulta vuota). Le celle relative all'intestazione, invece, non possono essere vuote. Ecco un esempio di tabella e relativa specifica:

Studente	Matricola	Anno	Esami
carlo	46124		
anna	42567	2	corso / voto: algebra 26; geometria 25

Ogni tabella è racchiusa dai tag `<table>` e `</table>`. Ogni riga è racchiusa dai tag `<tr>` e `</tr>` (*table row*). Ogni cella della prima riga è racchiusa dai tag `<th>` e `</th>` (*table heading*), mentre le celle delle righe successive sono racchiuse dai tag `<td>` e `</td>` (*table data*).

```
<table>
    <tr>
        <th> Studente </th>
        <th> Matricola </th>
        <th> Anno </th>
        <th> Esami </th>
    </tr>
    <tr>
        <td> carlo </td>
        <td >46124 </td>
        <td> </td>
        <td> </td>
    </tr>
    <tr>
        <td> anna </td>
        <td> 42567 </td>
        <td> 2 </td>
        <td>
          <table>
              <tr>
                  <th> corso </th>
                  <th> voto </th>
              </tr>
              <tr>
                  <td> algebra </td>
                  <td> 26 </td>
              </tr>
              <tr>
                  <td> geometria </td>
                  <td> 25 </td>
              </tr>
          </table>
        </td>
    </tr>
</table>
```

2. E' dato il seguente frammento di grammatica BNF relativo alla specifica del ciclo a condizione iniziale in un linguaggio imperativo:

while-stat → **while** *expr* **do** *stat*
expr → **true** | **false** | **id** | *expr* **and** *expr* | **(** *expr* **)**
stat → *assign-stat* | *while-stat*

Si chiede di specificare la semantica denotazionale del corrispondente frammento di linguaggio assumendo che:

- **id** rappresenti il nome di una variabile logica;
- la valutazione dell'operatore **and** sia in corto circuito, con valutazione degli operandi da sinistra a destra;
- sia disponibile una funzione M_{id}(**id**, s) che restituisce il valore della variabile **id** nello stato s;
- M_{id}(**id**, s) = ERRORE, qualora il valore della variabile **id** non sia definito;
- sia disponibile una funzione M_{assign}(*assign-stat*, s) che restituisce il nuovo stato (eventualmente ERRORE);
- il linguaggio di specifica denotazionale non disponga di operatori logici.

In particolare, si richiede la specifica della seguenti funzioni semantiche:

- M_{while}(*while-stat*, s)
- M_{expr}(*expr*, s)
- M_{stat}(*stat*, s)

3. Nel linguaggio *Scheme*, si assume di avere a disposizione la funzione booleana binaria **`prefix?`** (di cui non è richiesta la specifica), che stabilisce se una lista (primo argomento) è un prefisso di un'altra lista (secondo argomento). Si chiede di definire la funzione booleana binaria **`sublist?`** che, ricevendo le liste **`S`** ed **`L`**, stabilisce se **`S`** è una sottolista di **`L`**. Ecco alcuni esempi:

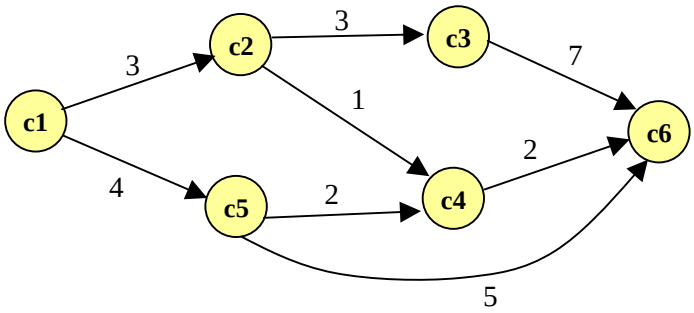
`S`	**`L`**	**`(sublist? S L)`**
`()`	`(1 2)`	`#t`
`(1 2)`	`()`	`#f`
`(1 2)`	`(1 3 2)`	`#f`
`(1 2)`	`(1 2)`	`#t`
`(1 2)`	`(2 3 1 2 3 4 5)`	`#t`
`((a b) c)`	`(() (a b) c () d)`	`#t`
`(() a)`	`(1 b () a c)`	`#t`
`(())`	`()`	`#f`

4. E' data una struttura tabellare che rappresenta un insieme di famiglie, come nel seguente esempio:

Cognome	Madre	Padre	Figli
rossi	anna	andrea	alessio angela
bianchi	rosa	rino	rino renata rachele
neri	monica	mino	mina mauro
gialli	serena	sergio	sonia sofia serena

Si chiede di specificare nel linguaggio *Haskell* la funzione **`omonimie`** (protocollo e corpo) che, ricevendo in ingresso una lista di famiglie (come nell'esempio), computa la lista dei cognomi delle famiglie in cui almeno uno dei figli ha lo stesso nome di uno dei genitori (nell'esempio, **`[bianchi, gialli]`**).

5. E’ data una base di fatti *Prolog* relativa alla specifica di una rete stradale modellata da un grafo (aciclico e diretto), in cui ogni nodo rappresenta una città, ed ogni arco rappresenta un collegamento tra due città, marcato dalla relativa distanza, come nel seguente esempio:




```
link(c1,3,c2).
link(c1,4,c5).
link(c2,3,c3).
link(c2,1,c4).
link(c3,7,c6).
link(c5,2,c4).
link(c5,5,c6).
link(c4,2,c6).
```

Si chiede di specificare il predicato **`percorso(C1,D12,C2)`**, che risulta vero qualora esista un cammino che collega la città **`C1`** alla città **`C2`**, di lunghezza minore o uguale a **`D12`**.

6. Illustrare le varie scelte progettuali relative al passaggio di parametri che sono sottoprogrammi.