Linguaggi per il Web – appello del 8/2/2013

COGNOME: _____

NOME: _____

MATRICOLA: _____

Autorizzo la pubblicazione del mio voto di questo esame sul sito web `http://www.dis.uniroma1.it/~rosati/lw`, secondo quanto prevede il decreto legislativo 196/2003 (codice in materia di protezione dei dati personali) che dichiaro di conoscere. In fede,

_____

Esercizio 1 (a) Data la seguente grammatica G:

$$S \to T \mid U$$
$$T \to abT \mid bUV \mid cUV$$
$$U \to baU \mid cTV \mid aTV$$
$$V \to cV \mid b$$

è possibile stabilire se G è una grammatica LL(1) senza costruire esplicitamente gli insiemi FIRST e FOLLOW e la tabella di parsing di G? motivare la risposta;

(b) Data la seguente grammatica G:

$$S \to Sa \mid a \mid SAb \mid SABc$$
$$A \to abA \mid aBC \mid abcC$$
$$B \to A \mid C \mid AB \mid CB$$
$$C \to CBA \mid CBB \mid ab$$

scrivere una grammatica G' tale che G' non presenti né ricorsione sinistra diretta né prefissi comuni e tale che $\mathcal{L}(G') = \mathcal{L}(G)$.

Esercizio 2 Si consideri il frammento del linguaggio Java costituito dalle stringhe che corrispondono alla definizione di una sequenza di istruzioni Java. Una sequenza di istruzioni è delimitata da parentesi graffe e contiene zero o più istruzioni. Ogni istruzione può essere: (i) una dichiarazione di una o pi ù variabili (separate da virgole) di tipo `int` o `String`; (ii) una assegnazione, il cui lato destro può essere una costante intera, una costante di tipo stringa, un identificatore di variabile, o una somma di sottoespressioni di questo tipo; (iii) una istruzione di tipo `if`, con ramo `else` opzionale, il cui ramo `then` e l'eventuale ramo `else` contengono una istruzione, e la cui condizione è un confronto di uguaglianza o disuguaglianza tra due espressioni, ognuna delle quali può essere una costante intera, una costante di tipo stringa, un identificatore di variabile, o una somma di sottoespressioni di questo tipo; (iv) una istruzione `while` il cui corpo contiene una istruzione e la cui condizione è un confronto di uguaglianza o disuguaglianza tra due espressioni, ognuna delle quali può essere una costante intera, una costante di tipo stringa, un identificatore di variabile, o una somma di sottoespressioni di questo tipo; (v) una sequenza di istruzioni delimitata da parentesi graffe.

Esempi di stringhe appartenenti a questo linguaggio sono i seguenti:

```
{
  int x,y,z;
  z=1;
  y=54+z+100;
  if (z==y+1000) x=12+y;
}
```

```
{
  paperino="ciao";
  z=met4+1+x+"abcd";
  while (x+y!=15+z+3) a=2;
  if (x==0) {
    int w,p1;
    w=1+z;
  }
  else b=c;
  { z=1; x=2; int w,f; }
  { z=1; x=2; { } y=3+c; }
}
```

```
{
  String paperino, a, b;
  int x,y,z;
  paperino="ciao"+"abcd";
  while (x+y!=15+z+3) {
    a=2;
    while (c==d+"ciao") { b=c; }
    if (x==0) {
      if (x!=y+3+z) w=1+z;
      else { p="abcd"+1; x=0; }
    }
  }
  else b=c;
}
```

Scrivere una grammatica non contestuale per tale linguaggio, dividendo la specifica del lessico del linguaggio (che va definita mediante espressioni regolari) dalla specifica della sintassi vera e propria. Scrivere preferibilmente tale specifica come specifica JavaCC.

Esercizio 3 Data la seguente DTD:

```
<!DOCTYPE x [
  <!ELEMENT x (a1,a2,a3)>
  <!ELEMENT a1 ((b1|b2)*,(b3,b4)+)>
  <!ELEMENT a2 ((b3|b4)*,(b1,b2)+)>
  <!ELEMENT a3 (b1?,b2*,b3?)>
  <!ELEMENT b1 (#PCDATA|a1)>
  <!ELEMENT b2 (#PCDATA)>
  <!ELEMENT b3 (#PCDATA|x)*>
  <!ELEMENT b4 EMPTY>
  <!ATTLIST x attrx CDATA #REQUIRED>
  <!ATTLIST a1 attra CDATA #IMPLIED>
  <!ATTLIST b1 attrb CDATA #REQUIRED>
]>
```

1) dire se la DTD è corretta ed in caso negativo evidenziare gli errori presenti e correggerli; 2) scrivere un documento XML che sia valido rispetto alla DTD (eventualmente corretta).

Esercizio 4 Data la seguente DTD:

```
<!DOCTYPE x [
  <!ELEMENT x (y1,y2)+>
  <!ELEMENT y1 ((z1|z2)*,(z3,z4)+)>
  <!ELEMENT y2 (z1?,z2*,z3?)>
  <!ELEMENT z1 (#PCDATA)>
  <!ELEMENT z2 (#PCDATA)>
  <!ELEMENT z3 (#PCDATA)>
  <!ELEMENT z4 EMPTY>
  <!ATTLIST x attrx CDATA #REQUIRED>
  <!ATTLIST y1 attry CDATA #IMPLIED>
]>
```

scrivere un XML Schema corrispondente a tale DTD.

Esercizio 5 Scrivere un foglio di stile XSL che, dato un documento XML, restituisce il documento tale che: 1) l'elemento radice di input viene copiato e il suo contenuto viene ricorsivamente trasformato; 2) ogni elemento figlio dell'elemento radice di input viene trasformato in un elemento `new`, aggiungendo un attributo di nome `name` e valore uguale al nome dell'elemento di input corrente. Inoltre il contenuto dell'elemento viene ricorsivamente trasformato; 3) ogni elemento che è figlio di un figlio dell'elemento radice viene copiato in output come figlio dell'elemento radice di output, aggiungendo come parte testuale dell'elemento la stringa `level 3 element`. Inoltre, il contenuto dell'elemento di input viene eliminato; 4) nessuna parte testuale del documento di input viene copiata in output.

Ad esempio, se il documento XML di input è il seguente:

```
<z>
  <d>testo 0
     <rad>testo 1</rad>
  </d>
  <b>
    <p>
      <w>
        <z>testo 2</z>
      </w>
    </p>
  </b>
  <c>
    <p/>
    <y>testo 3
      <d>testo 4</d>
    </y>
    <p>testo 5
      <d>testo 6</d>
    </p>
    <p/>
  </c>
  <z/>
</z>
```

il foglio di stile applicato al documento deve restituire il documento seguente:

```
<z>
  <new name="d">
  <rad>level 3 element</rad>
  </new>
  <new name="b">
  <p>level 3 element</p>
  </new>
  <new name="c">
  <p>level 3 element</p>
  <y>level 3 element</y>
  <p>level 3 element</p>
  <p>level 3 element</p>
  </new>
  <new name="z"/>
</z>
```

Esercizio 6 Siano `b`, `c`, `d`, `r`, `s` e `t` delle URI di un namespace con prefisso `p`. Scrivere un modello RDF che rappresenta le seguenti informazioni: "Le URI `b`, `c`, `d`, `r`, `s` e `t` rappresentano persone i cui nomi sono, rispettivamente, Beatrice, Carlo, Dario, Renata, Simona, Tommaso. Inoltre: Tommaso conosce una persona; Carlo conosce una persona che conosce Dario e una persona che conosce Beatrice; infine, Simona conosce una persona che crede che Dario conosca Carlo. Usare la URI `foaf:name` per esprimere il predicato "ha nome" e la URI `foaf:knows` per esprimere il predicato "conosce".