Linguaggi per il Web – appello del 10/2/2014

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:

$$S \to aT \mid bV$$
$$T \to Sc \mid Vd$$
$$V \to c \mid a$$

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:

$$S \to A \mid AB \mid AC \mid AS$$
$$A \to aA \mid aAC \mid aBC \mid aBA \mid AAAa$$
$$B \to BBB \mid BBC \mid b \mid bbc \mid BABB$$
$$C \to c \mid d \mid CS \mid CScc$$

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 istruzione Java. Una istruzione può essere: (i) una dichiarazione di una o più variabili (separate da virgole) di tipo `int` o `float` o `String`; (ii) una assegnazione, il cui lato destro può essere una costante intera, una costante di tipo `float`, una costante di tipo `String`, un identificatore di variabile, o una invocazione di metodo; (iii) una invocazione di metodo, in cui ogni argomento (parametro attuale) può essere una costante intera, una costante di tipo `float`, una costante di tipo `String`, o un identificatore di variabile, o a sua volta una invocazione di metodo; (iv) 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 `float`, una costante di tipo `String`, un identificatore di variabile, o una invocazione di metodo; (v) 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 `float`, una costante di tipo `String`, un identificatore di variabile, o una invocazione di metodo; (vi) una sequenza (anche vuota) di istruzioni delimitata da parentesi graffe; (vii) una istruzione `return` il cui argomento può essere una costante intera, una costante di tipo `float`, una costante di tipo `String`, un identificatore di variabile, o una invocazione di metodo.

Esempi di stringhe appartenenti a questo linguaggio sono i seguenti:

```
{
  int z,w;
  x=metodo3(123);
  metodo4();
  { y = p(x, 3.21); }
  while (a==-3.42)
    x23=-0.23E-12;
}
```

```
{
  String a, b12;
  { z=metodo5();
    {  y=met3(x,-1,3.0);
       z=metodo6(z,1); }
    if (p!=pluto(y)) a=z;
    else if (x==0) { return .25; } }
}
```

```
{
  float a12;
  paperino=-0.32e-3;
  { x=met1(x,abc12,0,a,y); y=0; }
  if (z==3.14) return x;
  else {
    pluto(25,x); if (p4!=0) x=1.15; }
}
```

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 (a) Scrivere un documento HTML contenente una form che presenta i seguenti campi:

- cognome e nome (casella di testo editabile lunga 50 caratteri)
- codice fiscale (casella di testo editabile lunga 16 caratteri)
- sesso (selezionabile tramite due bottoni radio)
- nazionalità (da scegliere da un menu che riporta alcune nazioni europee)
- indirizzo (casella di testo editabile lunga 50 caratteri)
- telefono (casella di testo editabile lunga 12 caratteri)
- bottone di invio
- bottone di reset

(b) Aggiungere al documento HTML una funzione JavaScript che esegue i seguenti controlli: (i) verifica che il campo cognome e nome non sia vuoto; (ii) verifica che il campo telefono sia un numero; (iii) verifica che l'indirizzo contenga almeno 6 caratteri; (iv) verifica che la nazionalità sia stata selezionata. Inoltre, fare in modo che, nel documento HTML, tale funzione JavaScript venga eseguita quando l'utente invia la form.

Esercizio 4 Data la seguente DTD:

```
<!DOCTYPE r [
  <!ELEMENT r (a*)>
  <!ELEMENT a ANY>
  <!ELEMENT b (#PCDATA|c|d)*>
  <!ELEMENT c ((a|b|c)*,e)>
  <!ELEMENT d (a*,b*,(c|d)+,e?)>
  <!ELEMENT e (#PCDATA)>
  <!ATTLIST b attrb CDATA #REQUIRED>
  <!ATTLIST c attrc 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) e che contenga il minor numero possibile di elementi; 3) scrivere un documento XML che sia valido rispetto alla DTD (eventualmente corretta) e che contenga almeno una occorrenza di tutti gli elementi dichiarati nella DTD.

Esercizio 5 Data la seguente DTD:

```
<!DOCTYPE x [
  <!ELEMENT x ((a|b|c)*,d?)>
  <!ELEMENT a (#PCDATA)>
  <!ELEMENT b EMPTY>
  <!ELEMENT c (#PCDATA)>
  <!ELEMENT d ANY>
  <!ATTLIST a attra CDATA #REQUIRED>
  <!ATTLIST b attrc CDATA #IMPLIED>
]>
```

scrivere un XML Schema corrispondente a tale DTD.

Esercizio 6 Scrivere un foglio di stile XSL che, dato un documento XML, restituisce il documento tale che: 1) l'elemento radice di input viene trasformato in un elemento `root`, e viene creato, come sottoelemento di tale elemento `root`, un elemento vuoto che ha per nome il nome dell'elemento radice di input. Inoltre, il contenuto dell'elemento radice di input viene ricorsivamente trasformato; 2) per ogni elemento che è figlio dell'elemento radice viene creato in output, come figlio dell'elemento radice, un elemento `child`, e viene creato per tale elemento un attributo di nome `name` e valore uguale al nome dell'elemento di input corrente. Inoltre, il contenuto dell'elemento di input viene ricorsivamente trasformato; 3) per ogni elemento che è figlio di un figlio dell'elemento radice viene creato in output, come figlio dell'elemento radice, un elemento `child`, e viene creato per tale elemento un attributo di nome `name` e valore uguale al nome dell'elemento di input corrente. Inoltre, il contenuto dell'elemento di input viene ricorsivamente trasformato; 4) tutti gli altri elementi non vengono copiati e il loro contenuto viene ricorsivamente trasformato: 5) nessuna parte testuale del documento di input viene copiata in output.

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

```
<a>
  <x>testo 0
  </x>
  <b>
    <p>
      <w>testo 2</w>
    </p>
  </b>
  <y>
    <p/>
    <y>testo 3
      <d>testo 4</d>
    </y>
  </y>
  <f/>
</a>
```

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

```
<root>
  <a/>
  <child name="x"/>
  <child name="b"/>
  <child name="p"/>
  <child name="y"/>
  <child name="p"/>
  <child name="y"/>
  <child name="f"/>
</root>
```