Tecnologie Web (12 CFU)

C.d.L. in Informatica

Compito del 1 settembre 2011

Nome:

Cognome:

Matricola:

Team:

□ Non è la prima volta che provo questo esame

Ricapitolo: fare qui sotto una croce sul numero di ciascun esercizio risolto. Se lo si consegna su un foglio protocollo, scrivere in maniera chiara il numero dell'esercizio presso la sua soluzione.

Esercizio		**Punti**	**Voto**
1	Domande di base	12	
2	HTML	6	
3	XSLT	4	
4	Linguaggi di validazione	4	
5	Semantic Web	6	
Totale		**32**	

Domande di base (12 punti)

Rispondere correttamente ad almeno tre delle seguenti domande:

Considerate la sequente frase: "XML e HTML sono due meta-linguaggi di markup". Spiegare se questa affermazione è vera o falsa e perché.

Dato il seguente frammento XML, scrivere un'espressione XPath che restituisca tutti i compiti che sono stati corretti.

```
<compiti>
        <compito data="2011-09-01" corretto="no" />
        <compito data="2011-02-14" corretto="sì" />
        <compito data="2012-01-15" corretto="no" />
        <compito data="2010-04-24" corretto="sì" />
</compiti>
```

Quali sono e come vengono visualizzati gli elementi HTML indicati dal seguente frammento di CSS

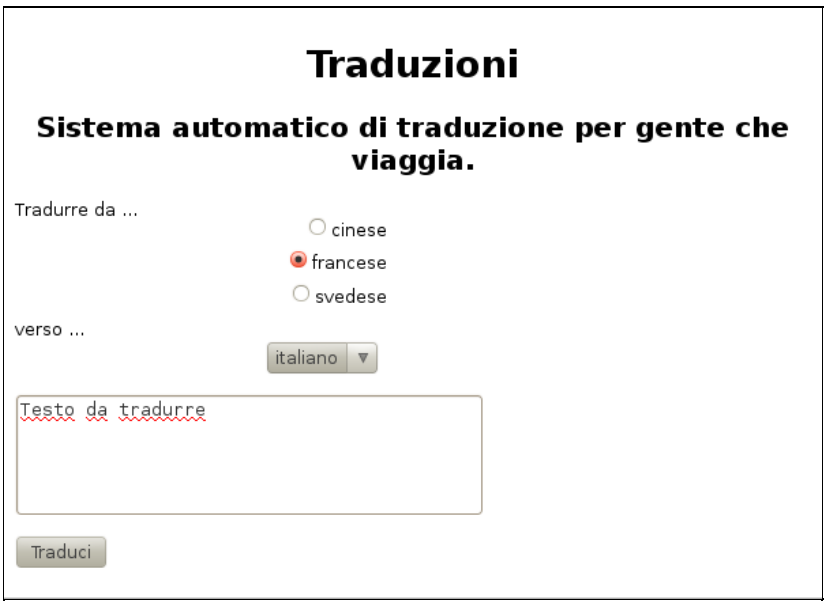
```
body {
    font-size: 12pt;
    text-align: justify;
}

.code {
    font-family:courier, fixed, monospace;
}
```

Che cos'è una tassonomia?

HTML (6 punti)

Scrivere il codice XHTML 1.0 Strict (e CSS) di un documento che rappresenti il contenuto di questa immagine:



Nello scrivere il codice si tengano in considerazione questi vincoli:

- le possibili lingue di destinazione sono "italiano", "tedesco" e "arabo";
- la lingua da tradurre è inviata nella variabile "da", la lingua verso la quale si desidera tradurre nella variabile "verso";
- i valori associati alle varie scelte di lingua possono essere scelti liberamente

Inoltre,

- nessun elemento deve contenere l'attributo `class` (di conseguenza non si possono usare usare selettori classe nel codice CSS);
- nessun elemento deve contenere l'attributo `style`;

Si consiglia di scrivere tutto il codice CSS in un "file" separato, non in elementi `<style>`

Nota: il DocType di XHTML 1.0 Strict è `<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">` e il namespace è `http://www.w3.org/1999/xhtml`.

XSLT (4 punti)

Scrivere un foglio di stile XSLT (indipendente dagli identificatori e dai nomi usati per le destinazioni) che trasforma il documento A nel documento B:

Documento A

```
<aereoporto>
    <volo id="AZ12234" ora="10:00">
        <destinazione>Roma</destinazione>
        <prezzo>230</prezzo>
        <posti_totali>100</posti_totali>
        <posti_liberi>10</posti_liberi>
    </volo>
    <volo id="AZ12444" ora="11:00">
        <destinazione>Milano</destinazione>
        <prezzo>200</prezzo>
        <posti_totali>100</posti_totali>
        <posti_liberi>5</posti_liberi>
    </volo>
    <volo id="AZ223" ora="09:00">
        <destinazione>Parigi</destinazione>
        <prezzo>500</prezzo>
        <posti_totali>180</posti_totali>
        <posti_liberi>20</posti_liberi>
    </volo>
    <volo id="AZ12444" ora="09:30">
        <destinazione>Roma</destinazione>
        <prezzo>190</prezzo>
        <posti_totali>100</posti_totali>
        <posti_liberi>2</posti_liberi>
    </volo>
    <volo id="AZ777" ora="17:30">
        <destinazione>Parigi</destinazione>
        <prezzo>340</prezzo>
        <posti_totali>140</posti_totali>
        <posti_liberi>23</posti_liberi>
    </volo>
    <volo id="AZ7655" ora="11:00">
        <destinazione>Roma</destinazione>
        <prezzo>50</prezzo>
        <posti_totali>100</posti_totali>
        <posti_liberi>32</posti_liberi>
    </volo>
</aereoporto>
```

Documento B

```
<destinazioni>
    <destinazione nome="Roma">
        <volo>
            <numero>AZ12234</numero>
            <ora>10:00</ora>
            <posti_venduti>90</posti_venduti>
        </volo>
        <volo>
            <numero>AZ12444</numero>
            <ora>09:30</ora>
            <posti_venduti>98</posti_venduti>
        </volo>
        <volo>
            <numero>AZ7655</numero>
            <ora>11:00</ora>
            <posti_venduti>68</posti_venduti>
        </volo>
    </destinazione>
    <destinazione nome="Milano">
        <volo>
            <numero>AZ12444</numero>
            <ora>11:00</ora>
            <posti_venduti>95</posti_venduti>
        </volo>
    </destinazione>
    <destinazione nome="Parigi">
        <volo>
            <numero>AZ223</numero>
            <ora>09:00</ora>
            <posti_venduti>160</posti_venduti>
        </volo>
        <volo>
            <numero>AZ777</numero>
            <ora>17:30</ora>
            <posti_venduti>117</posti_venduti>
        </volo>
    </destinazione>
</destinazioni>
```

Linguaggi di validazione (4 punti)

Si consideri il seguente documento XML:

```
<da-portare>
        <addosso>
                <tasche>
                        <oggetto tipo="cellulare">Nukia bianco</oggetto>
                </tasche>
                <zaino>
                        <capienza>20 litri</capienza>
                        <oggetto tipo="passaporto" trascurabile="false">
                                <rimpiazzo>
                                        <oggetto tipo="patente"/>
                                <rimpiazzo>
                        </oggetto>
                        <oggetto tipo="accessorio">caricabatterie Nukia</oggetto>
                        <oggetto tipo="accessorio"
                                 trascurabile="true">adattatore elettricità</oggetto>
                        <oggetto tipo="libro">the grapes of wrath</oggetto>
                </zaino>
        </addosso>
        <stiva>
                <valigia>
                        <capienza>60 litri</capienza>
                        <oggetto tipo="indumento">maglietta rossa</oggetto>
                        <oggetto tipo="indumento">maglietta a righe</oggetto>
                        <oggetto tipo="indumento">pantaloni neri</oggetto>
                        <oggetto tipo="indumento">scarpe blu</oggetto>
                </valigia>
                <valigia codice-rfid="11028837">
                        <oggetto tipo="accessorio">nécessaire con
                                <oggetto tipo="accessorio">rasoio</oggetto>,
                                <oggetto tipo="igiene">spazzolino</oggetto>,
                                <oggetto tipo="igiene">dentifricio</oggetto> e
                                <oggetto tipo="igiene">sapone</oggetto>
                        </oggetto>
                        <oggetto tipo="indumento">biancheria</oggetto>
                </valigia>
        </stiva>
</da-portare>
```

Usando questo ducumento come modello, si scrivano due schemi che rendano questo documento valido.

1. Il primo schema deve essere scritto usando XML Schema nello stile *fette di salame*.
2. Il secondo schema deve essere scritto usano Relax NG nello stile *bambole matrioška*.

Nota: massima libertà nella scelta dei dettagli sotto-specificati (ad esempio, ci deve essere per forza almeno un `oggetto` in `tasche`? l'ordine in cui compaiono gli oggetti è importante?)

Semantic Web (6 punti)

Considerate le seguenti asserzioni RDF:

```
:intertextual-semantics a ontology:ResearchPaper
        ; ontology:title “Intertextual semantics: A semantics for information design”
        ; ontology:hasPortrayal :10.1002/asi.21134/full
        ; ontology:keywords “semantics of markup” , “semiotic application” , “xml”
        ; ontology:hasSubjectTerm :markup-languages , :semantics .

:markup-languages a ontology:SubjectTerm
        ; ontology:prefLabel “Markup languages”
        ; ontology:inScheme <http://www.acm.org/class/1998>
        ; ontology:broader :document-preparation .

:semantics a ontology:SubjectTerm
        ; ontology:prefLabel “Semantics”
        ; ontology:inScheme <http://www.acm.org/class/1998>
        ; ontology:broader :formal-definitions-and-theory .

<http://www.acm.org/class/1998> a ontology:TermDictionary
        ; ontology:prefLabel “The 1998 ACM Computing Classification System”
        ; ontology:hasDiscipline :Computer_science .
```

Rispondere ai seguenti quesiti:

1. Scrivere una query SPARQL in modo che vengano restituiti i *term dictionary* a cui appartengono i vari *subject term* relativi ai soli articoli di ricerca (*research paper*)

2. In un linguaggio a scelta tra RDF/XML, Manchester Syntax e Turtle, definire un'ontologia contenente tutte le entità, presenti nelle varie assezioni RDF date, aventi prefisso "ontology:". L'ontologia deve essere il più possibile coerente con le assezioni RDF.

3. In un linguaggio a scelta tra RDF/XML, Manchester Syntax e Turtle, aggiungere all'ontologia la proprietà *ontology:paperHasDiscipline*. Definirla in modo che un ragionatore riesca a inferire automaticamente quali sono le discipline di un *research paper* P a partire da quelle dei vari *term dictionary* a cui fanno riferimento i *subject term* associati a P.

4. Considerando l'ipotesi del mondo aperto propria di OWL, so dire con precisione quante sono le discipline del dizionario "http://www.acm.org/class/1998"? Se sì, elencarle; se no, spiegarne la ragione.