Exercise

Produce an example XML document corresponding to a directed graph.
EXTENSIONS

A \rightarrow B

A \rightarrow B \rightarrow C

A \rightarrow B \rightarrow C \rightarrow D

\text{"AB"} = \text{dir} \downarrow \text{wt}

\text{dir} = \text{"BA"}

A' = \text{name}

A' = \text{src}

\text{dest} \rightarrow B
Compare with Lisp

List processing language
- S-expressions
- Cons pairs: car and cdr
- Lists as nil-terminated s-expressions
- Arbitrary structures built from few primitives
- Untyped
- Easy parsing
- Regularity of structure encourages recursion
Well-Formedness and Parsing

- An XML document maps to a parse tree (if well-formed; otherwise not XML)
  - Each element must end (exactly once): obvious nesting structure (one root)
  - An attribute can have at most one occurrence within an element; an attribute’s value must be a quoted string
- Well-formed XML documents can be parsed

```
X <A>
X <A>A</A> ... <A>
X <vertex name='A' name='B'/>
X <B>1</B>2</B>3</B>
```
XML InfoSet

A standardization of the low-level aspects of XML

- What an element looks like
- What an attribute looks like
- What comments and namespace references look like
- Ordering of attributes is irrelevant
- Representations of strings and characters

Primarily directed at tool vendors
Elements Versus Attributes: 1

■ Elements are essential for XML: structure and expressiveness
  ■ Have subelements and attributes
  ■ Can be repeated
  ■ Loosely might correspond to independently existing entities (*or association*)
  ■ Can capture all there is to attributes
Elements Versus Attributes: 2

- Attributes are not essential
  - End of the road: no subelements or attributes
  - Like text; restricted to string values
  - Guaranteed unique for each element
  - Capture adjunct information about an element
  - Great as references to elements

Good idea to use in such cases to improve readability

```html
<size="11pt" font="Times">
  <type>
  <type>
</size="11pt" font="Times">
Elements Versus Attributes: 3

```xml
<invoice>
  <price currency='USD'>
    19.95
  </price>
</invoice>

Or

<invoice amount='19.95' currency='USD'/>

Or even

<invoice amount='USD 19.95'/>
Meaning in XML

- Relational DBMSs work for highly structured information, but rely on column names for meaning
- Same problem in XML (reliance on names for meaning) but better connections to richer meaning representations
XML Namespaces: 2

```xml
<!-- xml* is reserved -->
<?xml version="1.0"?>
<arb*top xmlns="a URI" <!-- default namespace -->
    xmlns:arb*="http://wherever.it.might.be/arbit-ns"
    xmlns:random="http://another.one/random-ns">
  <arb*:aElem attr1="v1" attr2="v2">
    Optional text (PCDATA)
    <arb*:bElem attr1="v1" attr2="v2"/>
  </arb*:aElem>
</arb*:top>
```
XML Namespaces:

- Because XML supports custom vocabularies and interoperation, there is a high risk of name collision.
- A namespace is a collection of names.
- Namespaces must be identical or disjoint.
  - Crucial to support independent development of vocabularies.
  - MAC addresses.
  - Postal and telephone codes.
  - Vehicle identification numbers.
  - Domains as for the Internet.
- On the Web, use URIs for uniqueness.