XML Tags

XML tags are created like HTML tags. There's a start tag and a closing tag. **<TAG>**content**</TAG>**
The closing tag uses a slash after the opening bracket, just like in HTML.
Syntax
The following rules are used for using XML tags:

- **<TRAVEL>** differs from the tags **<Travel>** and **<travel>.** (XML is case sensitive**)**
  <!-- Comments **-->**
- **<amount>**135**</amount>** is a valid tag for an element amount that has the content 135.

Empty tags
Besides a starting tag and a closing tag, you can use an empty tag. An empty tag does not have a content.
The syntax differs from HTML: **<TAG/>**
<tag></tag> is also an empty tag.
Examples
<car color = "green"></car>
<car color = "green"/>

XML elements must follow these naming rules:(Naming rules for tags)

- Names can contain letters, numbers, and other characters
- Names must not start with a number or punctuation character
- Names must not start with the letters xml (or XML, or Xml, etc). Elements beginning with "XML" whether capitalized or not are reserved.
- Names cannot contain spaces

Any name can be used, no words are reserved, but the idea is to make names descriptive. Names with an underscore separator are nice.
Examples: <first_name>, <last_name>.
Avoid "-" and "." in names. For example, if you name something "first-name," it could be a mess if your software tries to subtract name from first. Or if you name something "first.name," your software may think that "name" is a property of the object "first."
Element names can be as long as you like, but don't exaggerate. Names should be short and simple, like this: <book_title> not like this: <the_title_of_the_book>.
XML documents often have a corresponding database, in which fields exist corresponding to elements in the XML document. A good practice is to use the naming rules of your database for the elements in the XML documents.
The **characters that are excluded from element names in XML are `&, <, ", and` **>, which are used by XML to indicate markup.
The ":" should not be used *in element names because it is reserved to be used for something called namespaces.*

XML Attributes

Elements in XML can use attributes. The syntax is:
<element attribute-name = "attribute-value">....</element>
The value of an attribute needs to be quoted, even if it contains only numbers.
An example
<car color = "green">volvo</car>
The same information can also be defined without using attributes:
<car>
<brand>volvo</brand>
<color>green</color>

```
</car>
```

Use of Elements vs. Attributes

Data can be stored in child elements or in attributes.
Take a look at these examples:

```
<person sex="female">
  <firstname>Anna</firstname>
  <lastname>Smith</lastname>
</person>

<person>
  <sex>female</sex>
  <firstname>Anna</firstname>
  <lastname>Smith</lastname>
</person>
```

In the first example sex is an attribute. In the last, sex is a child element. Both examples provide the same information. There are no rules about when to use attributes, and when to use child elements.

Avoid attributes

When possible try to avoid attributes. Data structures are more easy described in XML tags. Software that checks XML-documents can do a better job with tags than with attributes.Processing value of attribute is more complex than processing value of elements.

From HTML you will remember this: <IMG SRC="computer.gif">. The SRC attribute provides additional information about the IMG element.
In HTML (and in XML) attributes provide additional information about elements:

```
<img src="computer.gif">
<a href="demo.asp">
```

Attributes often provide information that is not a part of the data. In the example below, the file type is irrelevant to the data, but important to the software that wants to manipulate the element:

```
<file type="gif">computer.gif</file>
```

Quote Styles, "female" or 'female'?
Attribute values must always be enclosed in quotes, but either single or double quotes can be used. For a person's sex, the person tag can be written like this:

```
<person sex="female">
```

or like this:

```
<person sex='female'>
```

Note: If the attribute value itself contains double quotes it is necessary to use single quotes, like in this example:

```
<gangster name='George "Shotgun" Ziegler'>
```

Note: If the attribute value itself contains single quotes it is necessary to use double quotes, like in this example:

```
<gangster name="George 'Shotgun' Ziegler">
```

Try to store data in child elements.

The following three XML documents contain exactly the same information:
A date attribute is used in the first example:

```
<note date="12/11/2002">
        <to>Tove</to>
        <from>Jani</from>
        <heading>Reminder</heading>
        <body>Don't forget me this weekend!</body>
</note>
```

A date element is used in the second example:

```
<note>
        <date>12/11/2002</date>
        <to>Tove</to>
        <from>Jani</from>
        <heading>Reminder</heading>
        <body>Don't forget me this weekend!</body>
</note>
```

An expanded date element is used in the third:

```
<note>
<date>
  <day>12</day>
  <month>11</month>
  <year>2002</year>
</date>
<to>Tove</to>
<from>Jani</from>
<heading>Reminder</heading>
<body>Don't forget me this weekend!</body>
</note>
```

Avoid using attributes?

Some of the problems with using attributes are:

- attributes cannot contain multiple values (child elements can)
- attributes are not easily expandable (for future changes)
- attributes cannot describe structures (child elements can)
- attributes are more difficult to manipulate by program code
- attribute values are not easy to test against a Document Type Definition (DTD) - which is used to define the legal elements of an XML document

If you use attributes as containers for data, you end up with documents that are difficult to read and maintain. Try to use **elements** to describe data. Use attributes only to provide information that is not relevant to the data.

Don't end up like this (this is not how XML should be used):

```
<note day="12" month="11" year="2002"
to="Tove" from="Jani" heading="Reminder"
body="Don't forget me this weekend!">
</note>
```

We can assign ID references to elements. These ID references can be used to access XML elements in much the same way as the NAME or ID attributes in HTML. This example demonstrates this:

```
<messages>
  <note id="p501">
    <to>Tove</to>
    <from>Jani</from>
    <heading>Reminder</heading>
    <body>Don't forget me this weekend!</body>
  </note>

  <note id="p502">
```

```
   <to>Jani</to>
   <from>Tove</from>
   <heading>Re: Reminder</heading>
   <body>I will not!</body>
  </note>
</messages>
```

The ID in these examples is just a counter, or a unique identifier, to identify the different notes in the XML file, and not a part of the note data.

What I am trying to say here is that metadata (data about data) should be stored as attributes, and that data itself should be stored as elements.

1) Write the naming rules for creating elements name or tags name.
2) Identify valid and invalid names from a given set of element's names
3) What is Tag? Write about starting tags and ending tags
4) What is attribute? Write the rules related to using attributes in an element
5) What is empty tag? Whether empty tag may have attributes