== Learning XQuery through Examples ==

= A simple FOR =

- Find the question and answer for every true/false question

```
        for $tfq in fn:doc("bank.xml")//TFQuestion
        return ($tfq/Text, $tfq/@answer)
```

- Output:

```
        <Text>The Prime Minister, Stephen Harper, is Canada&amp;apos;s Head of State.</Text>,
        attribute answer {"False"},
        <Text>CSC343 is really hard.</Text>,
        attribute answer {"False"},
        <Text>XQuery is fun.</Text>,
        attribute answer {"True"}
```

- Need the brackets.
  Otherwise, things the FLWR ends at the comma,
  and the part is a whole new expression (with a variable out of scope)

= Adding in a where =

- Keep only the questions whose answer is True

```
        for $tfq in fn:doc("bank.xml")//TFQuestion
        where $tfq/@answer="True"
        return ($tfq/Text, $tfq/@answer)
```

- Output:

```
        <Text>XQuery is fun.</Text>, attribute answer {"True"}
```

= We can return XML! =

- Put that into XML format

```
        for $tfq in fn:doc("bank.xml")//TFQuestion
        return
           <ITEM>
              <QU> {data($tfq/Text)} </QU>
              <ANS> {data($tfq/@answer)} </ANS>
           </ITEM>
```

- Output:

```
        <ITEM><QU>The Prime Minister, Stephen Harper, is Canada&amp;apos;s Head of State.</QU>
<ANS>False</ANS></ITEM>,
        <ITEM><QU>CSC343 is really hard.</QU><ANS>False</ANS></ITEM>,
        <ITEM><QU>XQuery is fun.</QU><ANS>True</ANS></ITEM>
```

- Need the {}
- data function grabs the contents of the Text element and the answer attribute
  omitting the rest of the node.

- Here's another version that uses attributes instead

```
        for $tfq in fn:doc("bank.xml")//TFQuestion
        return
           <ITEM
              question = '{data($tfq/Text)}'
              answer = '{data($tfq/@answer)}'
           />
```

- Output:

```
        <ITEM question="The Prime Minister, Stephen Harper, is Canada&amp;apos;s Head of State."
answer="False"/>,
        <ITEM question="CSC343 is really hard." answer="False"/>,
        <ITEM question="XQuery is fun." answer="True"/>


= FOR vs LET =

- Version 1:

        for $tfq in fn:doc("bank.xml")//TFQuestion
        return
           <ITEM>
              {data($tfq/Text)}
           </ITEM>

- Output:
        <ITEM>The Prime Minister, Stephen Harper, is Canada&amp;apos;s Head of State.</ITEM>,
        <ITEM>CSC343 is really hard.</ITEM>,
        <ITEM>XQuery is fun.</ITEM>

- Version 2, with a let instead:
        let $tfq := fn:doc("bank.xml")//TFQuestion
        return
           <ITEM>
              {data($tfq/Text)}
           </ITEM>

- Output:
        <ITEM>The Prime Minister, Stephen Harper, is Canada&amp;apos;s Head of State. CSC343 is really
hard. XQuery is fun.</ITEM>


== Order-by ==

- Order according to the Text subelement:

        for $tfq in fn:doc("bank.xml")//TFQuestion
        order by $tfq/Text
        return
           <ITEM>
              {data($tfq/Text)}
           </ITEM>

- Output:
        <ITEM>CSC343 is really hard.</ITEM>,
        <ITEM>The Prime Minister, Stephen Harper, is Canada&amp;apos;s Head of State.</ITEM>,
        <ITEM>XQuery is fun.</ITEM>

- Can even order by something that's not included in the result:
        for $tfq in fn:doc("bank.xml")//TFQuestion
        order by $tfq/Hint
        return
           <ITEM>
              {data($tfq/Text)}
           </ITEM>

- Output:
        <ITEM>XQuery is fun.</ITEM>,
        <ITEM>CSC343 is really hard.</ITEM>,
        <ITEM>The Prime Minister, Stephen Harper, is Canada&amp;apos;s Head of State.</ITEM>

- Let's see the questions and their hints to confirm that this is the correct order:
        for $tfq in fn:doc("bank.xml")//TFQuestion
        order by $tfq/Hint
```

```
        return
           <ITEM>
              <QU>{data($tfq/Text)}</QU>
              <HINT>{data($tfq/Hint)}</HINT>
           </ITEM>
```

- Output:

```
        <ITEM><QU>XQuery is fun.</QU><HINT>Obscure syntax, practical value -- what&apos;s not to
like?</HINT></ITEM>,
        <ITEM><QU>CSC343 is really hard.</QU><HINT>Seriously?</HINT></ITEM>,
        <ITEM><QU>The Prime Minister, Stephen Harper, is Canada&apos;s Head of State.</QU><HINT>Think
royalty.</HINT></ITEM>
```