Bookmark this page

In these exercises, you will be working with a small XML data set drawn from the Stanford course catalog. There are multiple departments, each with a department chair, some courses, and professors and/or lecturers who teach courses. The XML data is here.

Instructions: Each problem asks you to write a query in XPath or XQuery. When you click "Check Answer" our back-end runs your query against the sample database using Saxon. It displays the result and compares your answer against the correct one. When you're satisfied with your solution for a given problem, click the "Submit" button to check your answer.

You may perform these exercises as many times as you like, so we strongly encourage you to keep working with them until you complete the exercises with full credit.

Q1

1.0/1.0 point (graded)
Return all Title elements (of both departments and courses).

Note: Your solution will need to reference *doc("courses.xml")* to access the data.

```
1 doc("courses.xml")//*/Title
```

Press ESC then TAB or click outside of the code editor to exit

Correct

Correct

Your Query Result:

```
<Title>Computer Science</Title>
<Title>Programming Methodology</Title>
<Title>Programming Abstractions</Title>
<Title>Computer Organization and Systems</Title>
<Title>Introduction to Probability for Computer Scientists</Title>
<Title>From Languages to Information</Title>
<Title>Compilers</Title>
<Title>Introduction to Databases</Title>
<Title>Artificial Intelligence: Principles and Techniques</Title>
<Title>Structured Probabilistic Models: Principles and Techniques</Title>
<Title>Machine Learning</Title>
<Title>Electrical Engineering</Title>
<Title>Digital Systems I</Title>
<Title>Digital Systems II</Title>
<Title>Linguistics</Title>
<Title>From Languages to Information</Title>
```

Expected Query Result:

```
<Title>Computer Science</Title>
<Title>Programming Methodology</Title>
<Title>Programming Abstractions</Title>
<Title>Computer Organization and Systems</Title>
<Title>Introduction to Probability for Computer Scientists</Title>
<Title>From Languages to Information</Title>
<Title>Compilers</Title>
<Title>Introduction to Databases</Title>
<Title>Artificial Intelligence: Principles and Techniques</Title>
<Title>Structured Probabilistic Models: Principles and Techniques</Title>
<Title>Machine Learning</Title>
<Title>Electrical Engineering</Title>
<Title>Digital Systems I</Title>
<Title>Digital Systems II</Title>
<Title>Linguistics</Title>
<Title>From Languages to Information</Title>
```

Submit

Q2

1.0/1.0 point (graded)

Return last names of all department chairs.

Note: Your solution will need to reference *doc("courses.xml")* to access the data.

```
1 doc("courses.xml")/Course_Catalog/Department/Chair/Professor/Last_Name
```

Press ESC then TAB or click outside of the code editor to exit

Correct

Correct

Your Query Result:

```
<Last_Name>Widom</Last_Name>
<Last_Name>Horowitz</Last_Name>
<Last_Name>Levin</Last_Name>
```

Expected Query Result:

```
<Last_Name>Widom</Last_Name>
<Last_Name>Horowitz</Last_Name>
<Last_Name>Levin</Last_Name>
```

Submit

Q3

1.0/1.0 point (graded)

Return titles of courses with enrollment greater than 500.

Note: Your solution will need to reference *doc("courses.xml")* to access the data.

```
1 doc("courses.xml")/Course_Catalog/Department/Course[@Enrollment > 500]/Title
```

Press ESC then TAB or click outside of the code editor to exit

Correct

Correct

Your Query Result:

```
<Title>Programming Methodology</Title>
<Title>Programming Abstractions</Title>
```

Expected Query Result:

```
<Title>Programming Methodology</Title>
<Title>Programming Abstractions</Title>
```

Submit

Q4

1.0/1.0 point (graded)

Return titles of departments that have some course that takes "CS106B" as a prerequisite.

Note: Your solution will need to reference *doc("courses.xml")* to access the data.

```
for $d in doc("courses.xml")/Course_Catalog/Department
where $d/Course/Prerequisites/Prereq = "CS106B"
return $d/Title
```

Press ESC then TAB or click outside of the code editor to exit

Correct

Correct

Your Query Result:

```
<Title>Computer Science</Title>
<Title>Electrical Engineering</Title>
```

Expected Query Result:

```
<Title>Computer Science</Title>
<Title>Electrical Engineering</Title>
```

Submit

Q5

1.0/1.0 point (graded)

Return last names of all professors or lecturers who use a middle initial. Don't worry about eliminating duplicates.

Note: Your solution will need to reference *doc("courses.xml")* to access the data.

```
doc("courses.xml")/Course_Catalog/Department//*/(Professor|Lecturer)[Middle_Initial]/Last_Name
```

Press ESC then TAB or click outside of the code editor to exit

Correct

Correct

Your Query Result:

```
<Last_Name>Cain</Last_Name>
<Last_Name>Cain</Last_Name>
<Last_Name>Aiken</Last_Name>
<Last_Name>Horowitz</Last_Name>
<Last_Name>Dally</Last_Name>
```

Expected Query Result:

```
<Last_Name>Cain</Last_Name>
<Last_Name>Cain</Last_Name>
<Last_Name>Aiken</Last_Name>
<Last_Name>Horowitz</Last_Name>
<Last_Name>Dally</Last_Name>
```

Submit

Q6

1.0/1.0 point (graded)

Return the count of courses that have a cross-listed course (i.e., that have "Cross-listed" in their description).

Note: Your solution will need to reference *doc("courses.xml")* to access the data.

```
1 count(doc("courses.xml")/Course_Catalog/Department/Course[contains(Description, "Cross-listed")])
```

Press ESC then TAB or click outside of the code editor to exit

Correct

Correct

Your Query Result:

```
2
```

Expected Query Result:

```
2
```

Submit

Q7

1.0/1.0 point (graded)

Return the average enrollment of all courses in the CS department.

Note: Your solution will need to reference *doc("courses.xml")* to access the data.

```
let $a := avg(doc("courses.xml")/Course_Catalog/Department[@Code="CS"]/Course/@Enrollment)
    return $a
```

Press ESC then TAB or click outside of the code editor to exit

Correct

Correct

Your Query Result:

```
336
```

Expected Query Result:

```
336
```

Submit

Q8

1.0/1.0 point (graded)

Return last names of instructors teaching at least one course that has "system" in its description and enrollment greater than 100.

Note: Your solution will need to reference *doc("courses.xml")* to access the data.

```
doc("courses.xml")/Course_Catalog/Department/Course[contains(Description, "system") and @Enrollment > 100]/Instructors/*/Last_Name
```

Press ESC then TAB or click outside of the code editor to exit

Correct

Correct

Your Query Result:

```
<Last_Name>Zelenski</Last_Name>
<Last_Name>Widom</Last_Name>
```

Expected Query Result:

```
<Last_Name>Zelenski</Last_Name>
<Last_Name>Widom</Last_Name>
```

Submit

Q9

1.0/1.0 point (graded)

Return the title of the course with the largest enrollment.

Note: Your solution will need to reference *doc("courses.xml")* to access the data.
(This problem is quite challenging; congratulations if you get it right.)

```
let $clist := doc("courses.xml")/Course_Catalog/Department/Course
for $c in $clist
   where $c/@Enrollment = max($clist/@Enrollment)
return $c/Title
```

Press ESC then TAB or click outside of the code editor to exit

Correct

Correct

Your Query Result:

```
<Title>Programming Methodology</Title>
```

Expected Query Result:

```
<Title>Programming Methodology</Title>
```

Submit