Integrative Programming Technologies

Work Sheet

1. Data Transformation Scenario

Question:
You are working on a government portal that receives citizen application data in XML format. You need to display this data in a formatted HTML report. Explain how XSLT and XPath can be used in this scenario. Include how you would handle conditional data (e.g., applications with a missing middle name).

Expected Answer:
XSLT can be used to transform the XML application data into a readable HTML format using `<xsl:template>` and `<xsl:for-each>`. XPath expressions will select the data fields (e.g., `/citizens/citizen`). Conditional data can be managed using `<xsl:if>` or `<xsl:choose>` to check for missing values (e.g., `not(middle_name)`).

2. System Interoperability

Question:
Your company exchanges product inventory data with multiple vendors. Vendor A uses one XML schema, while Vendor B uses another. How would you use XSLT and XPath to ensure compatibility and generate a unified XML format?

Expected Answer:
XSLT can transform vendor-specific XML into a unified schema by defining templates for each vendor's format. XPath expressions identify and extract the necessary fields despite structural differences, enabling standardization across systems.

3. Conditional Formatting

Question:
In an e-commerce website, product prices above $100 should be highlighted in red in the final output. How can this be implemented using XSLT and XPath?

Expected Answer:
Use `<xsl:choose>` with XPath condition `price > 100` to set a CSS class or `bgcolor` attribute conditionally in the HTML output using `<td style='color:red'>`.

4. Dynamic Table Generation

Question:
Given a hierarchical XML dataset representing departments and employees, describe how XSLT and XPath can be used to generate a nested HTML table showing each department with its employees.

Expected Answer:
Use `<xsl:for-each select="organization/department">` to iterate over departments. Inside that loop, use another `<xsl:for-each select="employee">` to list employees. XPath ensures correct node traversal, and XSLT generates the nested table structure

5. Data Aggregation

Question:
You need to calculate and display the total sales from an XML sales report. Which XPath function would you use, and how would it integrate with XSLT?

Expected Answer:
Use the `sum()` function in XPath: `<xsl:value-of select="sum(/sales/report/item/amount)"/>`. XSLT embeds this function within a template to output the total sales.

Discussion Questions

1. Why is field mapping important in data integration?

Expected Concepts:

- Different sources use different field names (`id` vs. `product_id`)
- Ensures consistency in your application or database schema
- Prevents data loss or misinterpretation

2. Explain the difference between data transformation and data cleaning.

Expected Answer:

- **Data transformation** = Converting format or structure (e.g., XML → JSON, renaming fields)
- **Data cleaning** = Fixing bad or inconsistent data (e.g., removing duplicates, correcting typos)

3. What could go wrong if you save combined data without validation?

Expected Concepts:

- Malformed or incomplete records saved
- Incorrect data types break the frontend or analytics
- Security risks (e.g., script injection in text fields)
- Garbage-in-garbage-out issues in ML or reports

4. How would you handle a situation where one supplier omits the `category` field entirely?

Expected Answer:

- Provide a **default value** (e.g., "Uncategorized")
- Flag the entry for review
- Ensure your system handles missing fields gracefully

5. Why might converting all data to JSON be a good strategy in a real-world application?

Expected Concepts:

- JSON is widely supported (APIs, databases like MongoDB)
- Easy to parse in most programming languages
- Human-readable, nested structure for complex data

6. In what real-world systems might you use similar data merging logic? Name at least two.

Expected Answers:

- E-commerce platforms merging supplier catalogs
- Hospital systems integrating patient records from multiple clinics
- Financial systems consolidating transaction feeds
- IoT platforms collecting sensor data in different formats

7. Why is it important to preserve the original data format or a log of transformations?

Expected Concepts:

- For **auditing** and **traceability**
- To allow reprocessing with different logic if needed
- Helps debug issues with mapping or validation later

Multiple-Choice Questions

1. What problems can arise if data is not normalized before merging from multiple sources?

A) Inconsistent field names
B) Inconsistent data types (e.g., prices stored as strings instead of numbers)
C) Duplicate records
D) All of the above

Correct Answer: D) All of the above

2. How would you handle a situation where one supplier omits the `category` field entirely?

A) Ignore the missing field and proceed with the data.
B) Provide a default value such as "Uncategorized" and flag the entry for review.
C) Skip the data entirely and ask the supplier to resend it.
D) Automatically delete the product entry from the dataset.

Correct Answer: B) Provide a default value such as "Uncategorized" and flag the entry for review.

3. Why might converting all data to JSON be a good strategy in a real-world application?

A) JSON is faster to process than other data formats like XML and CSV.
B) JSON is widely supported, human-readable, and well-suited for complex data structures.
C) JSON can store images and files more efficiently than other formats.
D) JSON is the only format that can be used to represent hierarchical data.

Correct Answer: B) JSON is widely supported, human-readable, and well-suited for complex data structures.

4. In what real-world systems might you use similar data merging logic?

A) A recommendation engine for movies or products
B) A financial system consolidating transaction data from various sources
C) A weather system collecting data from multiple meteorological stations
D) All of the above

Correct Answer: D) All of the above