TCP3151
Integrative Programming and Technologies

Trimester Oct/Nov 2025

Project Report

Lab Session: 1A

Tutor Name: Dr Chong Lee Ying

Group Name:

Student ID	Student Name	Major
1211107989	Lincoln Law Jia Hern	ST
1221305518	Gareth Tan Yao Wei	ST
1211101093	Eddie Chin Jia Ze	ST
1211108635	Tan Joon Shen	ST

Table of Contents

Table of Contents 2
Task Distribution Form 3
Part A 4
Introduction 4
1.0 Well-format And Valid XML Document 5
1.1 Requirements Compliance 5
1.2 Special Features 7
1.3 XML Structure 8
2.0 Logical View Diagram 19
3.0 DTD Document 20
4.0 XSLT Transformation 23
4.1 XSL (HTML) Output 36
5.0 Modify XML Portion 38
Part B 51
Question 1 51
Question 2 56

Task Distribution Form

No.	Student Name and Student ID	Tasks Completed	Student Signature
1.	Lincoln Law Jia Hern (1211107989)	I have completed Part A and Part B with my teammates.	
2.	Gareth Tan Yao Wei (1221305518)	I have completed Part B question 2 and assist my teammates in other part.	
3.	Eddie Chin Jia Ze (1211101093)	I have completed Part A and Part B with other teammates.	
4.	Tan Joon Shen (1211108635)	I have completed Part B question 1 and assist teammates in the other part.	

Part A

Introduction

This report documents the completion of Part A of the TCP3151 project, which focuses on XML processing, DTD definition, XSLT transformation, and Java-based XML manipulation. The project implements a Student Information System that demonstrates fundamental concepts of data representation, validation, and transformation using industry standard XML technologies.

The system successfully processes student records and academic transcripts, transforming structured XML data into a formatted HTML report while maintaining data integrity through DTD validation and Java-based structural modifications.

Project Part A Objective:

I. Create a well-formed XML document with specific structural requirements

II. Generate a logical view diagram showing XML hierarchy

III. Design a Document Type Definition (DTD) to validate the XML structure

IV. Develop an XSLT stylesheet to transform XML into HTML

V. Write a Java program to programmatically modify XML structure

1.0 Well-format And Valid XML Document

The XML document follows a logical view diagram to represent the Student Information System. The root element, **StudentInformationSystem**, contains all student and academic information. This structure allows the system to manage multiple students and their academic records efficiently.

1.1 Requirements Compliance

(a) At least 8 child elements

The XML document exceeds the requirement by including 10 direct child elements under each <Student> element:

1. PersonalInfo - Contains personal details (Name, DateOfBirth, Gender, ContactEmail)
2. EnrollmentInfo - Contains enrollment details (Program, YearOfStudy, EnrollmentDate)
3. AcademicRecord - Empty element for future academic records
4. ContactInformation - Contains contact details (PhoneNumber, Address)
5. Nationality - Student's nationality
6. Religion - Student's religion
7. ParentInfo - Contains parent information (FatherName, MotherName, ParentContact)
8. Transportation - Student's mode of transportation
9. Accommodation - Student's accommodation type
10. Notes - Empty element for additional note

(b) At least 1 empty element

The XML document includes 3 empty elements:

1. <AcademicRecord/> (in xml lines 36, 73, 112, 151) : empty element representing academic records placeholder

2. <CompletedCourses/> (in xml lines 178, 184, 190, 196) : empty element representing completed courses placeholder

3. <Notes/> (in xml lines 56, 94, 133, 172): empty element representing additional notes or comments about the student

(c) At least 2 attributes

The XML document includes 6 attributes:

1. universityCode (line 20)

   - Type: CDATA
   - Value: “MMU001”
   - Purpose: Unique identifier for the university
   - Usage: <StudentInformationSystem universityCode=“MMU001”>

2. academicYear (line 20)

   - Type: CDATA
   - Value: “2025”
   - Purpose: Specifies the current academic year
   - Usage: <StudentInformationSystem academicYear=“2025”>

3. studentID (lines 22, 59, 98, 137)

- Type: ID (unique identifier)
- Values: “S1211108635”, “S1221305518”, “S1211107989”, “S1211101093”
- Purpose: Unique identifier for each student
- Usage: <Student studentID=“S1211108635”>

4. enrollmentStatus (lines 22, 59, 98, 137)

- Type: Enumeration (Active | Inactive | Suspended)
- Values: “Active” or “Inactive”
- Purpose: Indicates current enrollment status
- Usage: <Student enrollmentStatus=“Active”>

5. transcriptID (lines 176, 182, 188, 194)

- Type: ID (unique identifier)
- Values: “T001”, “T002”, “T003”, “T004”
- Purpose: Unique identifier for each transcript
- Usage: <Transcript transcriptID=“T001”>

6. studentRef (lines 176, 182, 188, 194)

- Type: IDREF (reference to studentID)
- Values: “S1211108635”, “S1221305518”, “S1211107989”, “S1211101093”
- Purpose: Links transcript to corresponding student
- Usage: <Transcript studentRef=“S1211108635”>

1.2 Special Features

The XML uses two entity references defined in the DTD:

1. &university; (lines 40, 77, 116, 155)

- Expands to: “Multimedia University”

- Usage: <Address>Room 100, &university;</Address>
- Benefit: Centralized definition allows easy updates across the document

2. &academicTerm; (lines 179, 185, 191, 197)

- Expands to: “Trimester Oct/Nov 2025”
- Usage: <AcademicTerm>&academicTerm;</AcademicTerm>
- Benefit: Ensures consistency in term naming

1.3 XML Structure

Document name (StudentSystem.xml)

```
<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE StudentInformationSystem SYSTEM "StudentSystem.dtd">

<?xml-stylesheet type="text/xsl" href="StudentSystem.xsl"?>



<!-- ========== HIGHLIGHTED REQUIREMENTS ========== -->

<!-- (a) At least 10 DIRECT child elements under Student: -->

<!--    1. PersonalInfo -->

<!--    2. EnrollmentInfo -->

<!--    3. AcademicRecord (EMPTY) -->

<!--    4. ContactInformation -->

<!--    5. Nationality (NEW) -->

<!--    6. Religion (NEW) -->

<!--    7. ParentInfo (NEW) -->

<!--    8. Transportation (NEW) -->
```

```
<!--     9. Accommodation (NEW) -->

<!--     10. Notes (EMPTY) -->

<!-- (b) At least 1 empty element: AcademicRecord, CompletedCourses, Notes -->

<!-- (c) At least 2 attributes: studentID, enrollmentStatus, transcriptID, studentRef, universityCode, academicYear -->

   <!- - c) 1st Attribute -->                                    <!- -c) 2nd Attribute - ->

<StudentInformationSystem universityCode="MMU001" academicYear="2025">

   <!- - c) 1st Attribute -->              <!- -c) 2nd Attribute - ->

<Student studentID="S1211108635" enrollmentStatus="Inactive">

   <!- - a) 1st Child Element - ->

   <PersonalInfo>

      <Name>Tan Joon Shen</Name>

      <DateOfBirth>2004-07-19</DateOfBirth>

      <Gender>Male</Gender>

      <ContactEmail>1211108635@student.mmu.edu.my</ContactEmail>

   </PersonalInfo>

   <!- - a) 2nd Child Element - ->

   <EnrollmentInfo>

      <Program>Computer Science</Program>

      <YearOfStudy>3</YearOfStudy>

      <EnrollmentDate>2026-01-01</EnrollmentDate>

   </EnrollmentInfo>
```

```
    <!- - a) 3rd Child Element  b) 1st Empty Element - ->
<AcademicRecord/>
    <!- - a) 4th Child Element - ->
    <ContactInformation>
        <PhoneNumber>123-1000</PhoneNumber>
       <Address>Room 100, &university;</Address>
    </ContactInformation>
    <!- - a) 5th Child Element - ->
    <Nationality>Malaysian</Nationality>
    <!- - a) 6th Child Element - ->
    <Religion>Islam</Religion>
    <!- - a) 7th Child Element - ->
    <ParentInfo>
       <FatherName>Tan Joon Kok</FatherName>
       <MotherName>Mary Wong</MotherName>
       <ParentContact>1234-2000</ParentContact>
    </ParentInfo>
    <!- - a) 8th Child Element - ->
    <Transportation>University Bus</Transportation>
    <!- - a) 9th Child Element - ->
    <Accommodation>On-Campus Hostel</Accommodation>
```

```
    <!- - a) 10th Child Element  b) 2nd Empty Element - ->
<Notes/>
    </Student>
    <!- - c) 1st Attribute -->                <!- -c) 2nd Attribute - ->
<Student studentID="S1221305518" enrollmentStatus="Active">
    <!- - a) 1st Child Element - ->
    <PersonalInfo>
        <Name>Tan Yao Wei</Name>
        <DateOfBirth>2000-03-22</DateOfBirth>
        <Gender>Male</Gender>
        <ContactEmail>1221305518@student.mmu.edu.my</ContactEmail>
    </PersonalInfo>
    <!- - a) 2nd Child Element - ->
    <EnrollmentInfo>
        <Program>Business Administration</Program>
        <YearOfStudy>2</YearOfStudy>
        <EnrollmentDate>2023-09-01</EnrollmentDate>
    </EnrollmentInfo>
    <!- - a) 3rd Child Element  b) 1st Empty Element - ->
    <AcademicRecord/>
```

```
<!- - a) 4th Child Element - ->
<ContactInformation>
    <PhoneNumber>1234-1121</PhoneNumber>
    <Address>Room 101, &university;</Address>
</ContactInformation>
<!- - a) 5th Child Element - ->
<Nationality>Chinese</Nationality>
<!- - a) 6th Child Element - ->
<Religion>Buddhist</Religion>
<!- - a) 7th Child Element - ->
<ParentInfo>
    <FatherName>David Tan</FatherName>
    <MotherName>Linda Wong</MotherName>
    <ParentContact>2345-2431</ParentContact>
</ParentInfo>
<!- - a) 8th Child Element - ->
<Transportation>Private Car</Transportation>
<!- - a) 9th Child Element - ->
<Accommodation>Off-Campus Apartment</Accommodation>
 <!- - a) 10th Child Element  c) 2nd Empty Element- ->
<Notes/>
```

```
    </Student>

    <!- - c) 1st Attribute -->                <!- -c) 2nd Attribute - ->

<Student studentID="S1211107989" enrollmentStatus="Active">

    <!- - a) 1st Child Element - ->

    <PersonalInfo>

        <Name>Lincoln Law</Name>

        <DateOfBirth>2003-11-08</DateOfBirth>

        <Gender>Male</Gender>

        <ContactEmail>1211107989@student.mmu.edu.my</ContactEmail>

    </PersonalInfo>

    <!- - a) 2nd Child Element - ->

    <EnrollmentInfo>

        <Program>Engineering</Program>

        <YearOfStudy>4</YearOfStudy>

        <EnrollmentDate>2023-09-01</EnrollmentDate>

    </EnrollmentInfo>

    <!- - a) 3rd Child Element  b) 1st Empty Element - ->

    <AcademicRecord/>

    <!- - a) 4th Child Element - ->

    <ContactInformation>

        <PhoneNumber>456-7890</PhoneNumber>

        <Address>Room 102, &university;</Address>
```

```
    </ContactInformation>

    <!- - a) 5th Child Element - ->

    <Nationality>Malaysian</Nationality>

    <!- - a) 6th Child Element - ->

    <Religion>Christian</Religion>

    <!- - a) 7th Child Element - ->

    <ParentInfo>

        <FatherName>Li Chen</FatherName>

        <MotherName>Wei Chen</MotherName>

        <ParentContact>345-0987</ParentContact>

    </ParentInfo>

    <!- - a) 8th Child Element - ->

    <Transportation>Motorcycle</Transportation>

    <!- - a) 9th Child Element - ->

    <Accommodation>On-Campus Hostel</Accommodation>

     <!- - a) 10th Child Element  c) 2nd Empty Element- ->

    <Notes/>



    </Student>

    <!- - c) 1st Attribute -->                <!- -c) 2nd Attribute - ->

<Student studentID="S1211101093" enrollmentStatus="Active">
```

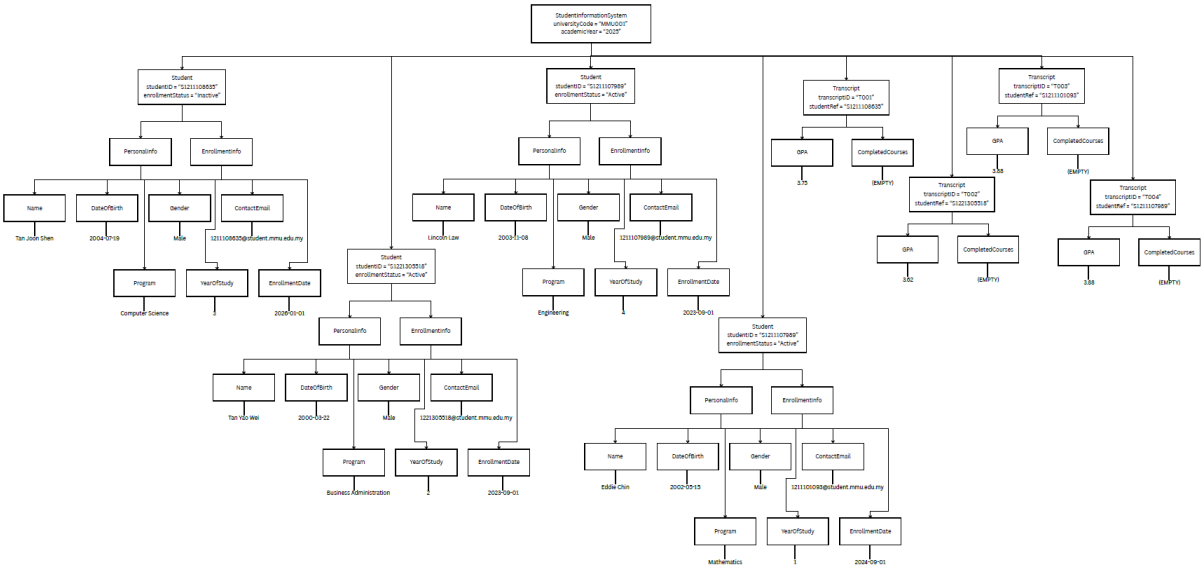
```
<!- - a) 1st Child Element - ->
<PersonalInfo>
    <Name>Eddie Chin</Name>
    <DateOfBirth>2002-05-15</DateOfBirth>
    <Gender>Male</Gender>
    <ContactEmail>1211101093@student.mmu.edu.my</ContactEmail>
</PersonalInfo>
<!- - a) 2nd Child Element - ->
<EnrollmentInfo>
    <Program>Mathematics</Program>
    <YearOfStudy>1</YearOfStudy>
    <EnrollmentDate>2024-09-01</EnrollmentDate>
</EnrollmentInfo>
<!- - a) 3rd Child Element  b) 1st Empty Element - ->
<AcademicRecord/>
<!- - a) 4th Child Element - ->
<ContactInformation>
    <PhoneNumber>034-7456</PhoneNumber>
    <Address>Room 103, &university;</Address>
</ContactInformation>
<!- - a) 5th Child Element - ->
<Nationality>Malaysian</Nationality>
```

```
<!- - a) 6th Child Element - ->

<Religion>Buddhism</Religion>

<!- - a) 7th Child Element - ->

<ParentInfo>

    <FatherName>Chin Long</FatherName>

    <MotherName>Mei Ren</MotherName>

    <ParentContact>555-2003</ParentContact>

</ParentInfo>

<!- - a) 8th Child Element - ->

<Transportation>Public Transport</Transportation>

<!- - a) 9th Child Element - ->

<Accommodation>Off-Campus Rental</Accommodation>

<!- - a) 10th Child Element  c) 2nd Empty Element- ->

<Notes/>



</Student>

<!- - c) 1st Attribute -->          <!- -c) 2nd Attribute - ->

<Transcript transcriptID="T001" studentRef="S1211108635">

<!- - a) 1st Child Element - ->

   <GPA>3.75</GPA>

<!- - a) 2nd Child Element  b) 1st Empty Element - ->

   <CompletedCourses/>
```

```
<!- - a) 3rd Child Element  - ->

    <AcademicTerm>&academicTerm;</AcademicTerm>

</Transcript>

<!- - c) 1st Attribute -->          <!- -c) 2nd Attribute - ->

<Transcript transcriptID="T002" studentRef="S1221305518">

<!- - a) 1st Child Element - ->

    <GPA>3.62</GPA>

<!- - a) 2nd Child Element  b) 1st Empty Element - ->

    <CompletedCourses/>

<!- - a) 3rd Child Element  - ->

    <AcademicTerm>&academicTerm;</AcademicTerm>

</Transcript>

<!- - c) 1st Attribute -->      <!- -c) 2nd Attribute - ->

<Transcript transcriptID="T003" studentRef="S1211101093">

<!- - a) 1st Child Element - ->

    <GPA>3.88</GPA>

<!- - a) 2nd Child Element  b) 1st Empty Element - ->

    <CompletedCourses/>

<!- - a) 3rd Child Element  - ->

    <AcademicTerm>&academicTerm;</AcademicTerm>

</Transcript>
```

```
<!- - c) 1st Attribute and 2nd Attribute - ->

   <Transcript transcriptID="T004" studentRef="S1211107989">

   <!- - a) 1st Child Element - ->

      <GPA>3.88</GPA>

   <!- - a) 2nd Child Element  b) 1st Empty Element - ->

      <CompletedCourses/>

<!- - a) 3rd Child Element  - ->

      <AcademicTerm>&academicTerm;</AcademicTerm>

   </Transcript>



</StudentInformationSystem>
```

2.0 Logical View Diagram



3.0 DTD Document

A DTD file named **StudentSystem.dtd** was created to validate the XML document.

(a) Element declaration

- All elements used in the XML document, such as Student, PersonalInfo, EnrollmentInfo, Nationality, and Transcript, are declared in the DTD.

(b) Attribute declaration using ID and IDREF

- studentID is declared as type ID in the <Student> element
- studentRef is declared as type IDREF in the <Transcript> element
- This establishes a valid reference between student records and transcript records.

(c) Entity declaration

- An internal entity is declared in the DTD to represent university-related information, improving readability and reusability.

The required portions (a–c) are highlighted/underlined in the DTD document:

```
<!ELEMENT StudentInformationSystem (Student+, Transcript*)>
<!ATTLIST StudentInformationSystem universityCode CDATA #REQUIRED>
<!ATTLIST StudentInformationSystem academicYear CDATA #REQUIRED>

<!-- Student Element - HAS 10 CHILD ELEMENTS -->
<!ELEMENT Student (PersonalInfo, EnrollmentInfo, AcademicRecord, ContactInformation, Nationality, Religion, ParentInfo, Transportation, Accommodation, Notes)>
<!ATTLIST Student studentID ID #REQUIRED>
<!ATTLIST Student enrollmentStatus (Active | Inactive | Suspended) #REQUIRED>
```

```
<!-- PersonalInfo Elements -->
<!ELEMENT PersonalInfo (Name, DateOfBirth, Gender, ContactEmail)>
<!ELEMENT Name (#PCDATA)>
<!ELEMENT DateOfBirth (#PCDATA)>
<!ELEMENT Gender (#PCDATA)>
<!ELEMENT ContactEmail (#PCDATA)>

<!-- EnrollmentInfo Elements -->
<!ELEMENT EnrollmentInfo (Program, YearOfStudy, EnrollmentDate)>
<!ELEMENT Program (#PCDATA)>
<!ELEMENT YearOfStudy (#PCDATA)>
<!ELEMENT EnrollmentDate (#PCDATA)>

<!-- AcademicRecord Element (empty) -->
<!ELEMENT AcademicRecord EMPTY>

<!-- ContactInformation Elements -->
<!ELEMENT ContactInformation (PhoneNumber, Address)>
<!ELEMENT PhoneNumber (#PCDATA)>
<!ELEMENT Address (#PCDATA)>

<!-- NEW SIMPLE ELEMENTS -->
<!ELEMENT Nationality (#PCDATA)>
<!ELEMENT Religion (#PCDATA)>
<!ELEMENT Transportation (#PCDATA)>
```

```
<!ELEMENT Accommodation (#PCDATA)>
<!ELEMENT Notes EMPTY>

<!-- ParentInfo Elements -->
<!ELEMENT ParentInfo (FatherName, MotherName, ParentContact)>
<!ELEMENT FatherName (#PCDATA)>
<!ELEMENT MotherName (#PCDATA)>
<!ELEMENT ParentContact (#PCDATA)>

<!-- Transcript Element -->
<!ELEMENT Transcript (GPA, CompletedCourses, AcademicTerm?)>
<!ATTLIST Transcript transcriptID ID #REQUIRED>
<!ATTLIST Transcript studentRef IDREF #REQUIRED>

<!ELEMENT GPA (#PCDATA)>
<!ELEMENT CompletedCourses EMPTY>
<!ELEMENT AcademicTerm (#PCDATA)>

<!-- Entity Declarations -->
<!ENTITY university "Multimedia University">
<!ENTITY academicTerm "Trimester Oct/Nov 2025">
```

4.0 XSLT Transformation

An XSL file named **StudentSystem.xsl** was written to transform the XML document into an HTML output. The XSL file contains the following required components:

i. xsl:apply-templates – used to process Student and Transcript nodes

ii. xsl:call-template – used to modularize repeated formatting logic

iii. xsl:sort – used to sort student records alphabetically by name

iv. xsl:choose – used to conditionally display student status (Active / Inactive)

v. CSS styling – applied to enhance the visual presentation of the HTML output

The required portions (i–v) are highlighted/underlined in the XSL file:

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
    xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

    <xsl:output method="html" encoding="UTF-8" indent="yes"/>
    <xsl:strip-space elements="*"/>

    <!-- ========== HIGHLIGHTED REQUIREMENTS ========== -->
    <!-- (ii) xsl:call-template - used below in cssStyle template -->
    <xsl:template name="cssStyle">
      <style>
        body {
          font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;
          margin: 40px;
          background-color: #f5f5f5;
```

```
        }
        h1 {
            color: #2c3e50;
            border-bottom: 3px solid #3498db;
            padding-bottom: 10px;
        }
        h2 {
            color: #34495e;
            margin-top: 30px;
        }
        table {
            border-collapse: collapse;
            width: 100%;
            margin-bottom: 30px;
            background-color: white;
            box-shadow: 0 2px 4px rgba(0,0,0,0.1);
        }
        th, td {
            border: 1px solid #ddd;
            padding: 12px;
            text-align: left;
        }
        th {
            background-color: #3498db;
            color: white;
            font-weight: bold;
```

```
        }
        tr:nth-child(even) {
            background-color: #f9f9f9;
        }
        tr:hover {
            background-color: #f1f1f1;
        }
        .badge {
            padding: 4px 12px;
            border-radius: 12px;
            font-size: 0.9em;
            font-weight: bold;
        }
        .badge-active {
            background-color: #2ecc71;
            color: white;
        }
        .badge-inactive {
            background-color: #e74c3c;
            color: white;
        }
        .info-box {
            background-color: #ecf0f1;
            padding: 15px;
            border-radius: 5px;
            margin-bottom: 20px;
```

```
}
ul {
    background-color: white;
    padding: 20px;
    border-radius: 5px;
    box-shadow: 0 2px 4px rgba(0,0,0,0.1);
}
li {
    padding: 5px 0;
}
.student-details {
    background-color: white;
    padding: 20px;
    margin-bottom: 20px;
    border-radius: 5px;
    box-shadow: 0 2px 4px rgba(0,0,0,0.1);
}
.student-details h3 {
    color: #3498db;
    margin-top: 0;
}
.detail-row {
    display: grid;
    grid-template-columns: 180px 1fr;
    margin: 8px 0;
}
```

```
            .detail-label {
                font-weight: bold;
                color: #555;
            }
            .section-title {
                background-color: #3498db;
                color: white;
                padding: 8px;
                margin-top: 15px;
                margin-bottom: 10px;
                border-radius: 3px;
                font-weight: bold;
            }
        </style>
    </xsl:template>

    <!-- Root Template -->
    <xsl:template match="/StudentInformationSystem">
        <html>
            <head>
                <title>Student Information System</title>
                <meta charset="UTF-8"/>
                <!-- (ii) xsl:call-template - calling the cssStyle template -->
                <xsl:call-template name="cssStyle"/>
            </head>
```

```
        <body>
            <h1>Student Information System</h1>
            <div class="info-box">
                <p>
                    <strong>University Code:</strong>
                    <xsl:value-of select="@universityCode"/><br/>
                    <strong>Academic Year:</strong>
                    <xsl:value-of select="@academicYear"/>
                </p>
            </div>

            <!-- (i) xsl:apply-templates & (iii) xsl:sort - Loop #1 -->
            <h2>Students Summary Table</h2>
            <table>
                <tr>
                    <th>ID</th>
                    <th>Name</th>
                    <th>Date of Birth</th>
                    <th>Gender</th>
                    <th>Program</th>
                    <th>Year</th>
                    <th>Status</th>
                </tr>

                <!-- (i) xsl:apply-templates with (iii) xsl:sort -->
                <xsl:apply-templates select="Student">
```

```
            <xsl:sort select="PersonalInfo/Name"/>
        </xsl:apply-templates>
    </table>

    <!-- Detailed Student Information -->
    <h2>Detailed Student Information</h2>
    <xsl:for-each select="Student">
        <xsl:sort select="@studentID"/>
        <div class="student-details">
            <h3>
                <xsl:value-of select="@studentID"/> -
                <xsl:value-of select="PersonalInfo/Name"/>
                <!-- (iv) xsl:if - conditional display -->
                <xsl:if test="@enrollmentStatus='Active'">
                    <span class="badge badge-active">Active</span>
                </xsl:if>
                <xsl:if test="@enrollmentStatus!='Active'">
                    <span class="badge badge-inactive">
                        <xsl:value-of select="@enrollmentStatus"/>
                    </span>
                </xsl:if>
            </h3>

            <div class="section-title">Personal Information</div>
            <div class="detail-row">
                <span class="detail-label">Full Name:</span>
```

```
            <span><xsl:value-of select="PersonalInfo/Name"/></span>
        </div>
        <div class="detail-row">
            <span class="detail-label">Date of Birth:</span>
            <span><xsl:value-of select="PersonalInfo/DateOfBirth"/></span>
        </div>
        <div class="detail-row">
            <span class="detail-label">Gender:</span>
            <span><xsl:value-of select="PersonalInfo/Gender"/></span>
        </div>
        <div class="detail-row">
            <span class="detail-label">Contact Email:</span>
          <span><xsl:value-of select="PersonalInfo/ContactEmail"/></span>
        </div>
        <div class="detail-row">
            <span class="detail-label">Nationality:</span>
            <span><xsl:value-of select="Nationality"/></span>
        </div>
        <div class="detail-row">
            <span class="detail-label">Religion:</span>
            <span><xsl:value-of select="Religion"/></span>
        </div>

        <div class="section-title">Contact Information</div>
        <div class="detail-row">
            <span class="detail-label">Phone Number:</span>
```

```
<span><xsl:value-of select="ContactInformation/PhoneNumber"/></span>
                </div>
                <div class="detail-row">
                    <span class="detail-label">Address:</span>

<span><xsl:value-of select="ContactInformation/Address"/></span>
                </div>

                <div class="section-title">Parent Information</div>
                <div class="detail-row">
                    <span class="detail-label">Father's Name:</span>
                    <span><xsl:value-of select="ParentInfo/FatherName"/></span>
                </div>
                <div class="detail-row">
                    <span class="detail-label">Mother's Name:</span>
                    <span><xsl:value-of select="ParentInfo/MotherName"/></span>
                </div>
                <div class="detail-row">
                    <span class="detail-label">Parent Contact:</span>
                    <span><xsl:value-of select="ParentInfo/ParentContact"/></span>
                </div>

                <div class="section-title">Enrollment Information</div>
                <div class="detail-row">
                    <span class="detail-label">Program:</span>
```

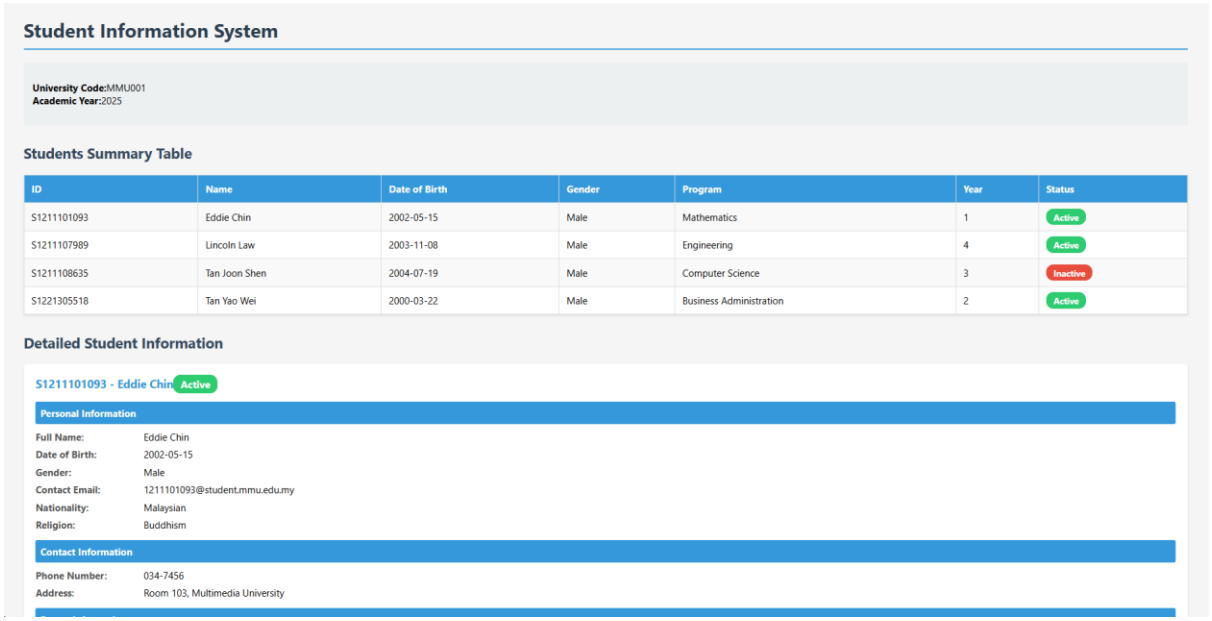
```
                        <span><xsl:value-of select="EnrollmentInfo/Program"/></span>
                    </div>
                    <div class="detail-row">
                        <span class="detail-label">Year of Study:</span>

<span><xsl:value-of select="EnrollmentInfo/YearOfStudy"/></span>
                    </div>
                    <div class="detail-row">
                        <span class="detail-label">Enrollment Date:</span>

<span><xsl:value-of select="EnrollmentInfo/EnrollmentDate"/></span>
                    </div>

                    <div class="section-title">Other Information</div>
                    <div class="detail-row">
                        <span class="detail-label">Transportation:</span>
                        <span><xsl:value-of select="Transportation"/></span>
                    </div>
                    <div class="detail-row">
                        <span class="detail-label">Accommodation:</span>
                        <span><xsl:value-of select="Accommodation"/></span>
                    </div>
                </div>
            </xsl:for-each>

            <!-- Loop #2 with xsl:for-each and xsl:sort -->
```

```
<h2>Academic Transcripts</h2>
<table>
  <tr>
    <th>Transcript ID</th>
    <th>Student Ref</th>
    <th>Student Name</th>
    <th>GPA</th>
    <th>Academic Term</th>
  </tr>

  <!-- (iii) xsl:sort within xsl:for-each -->
  <xsl:for-each select="Transcript">
    <xsl:sort select="GPA" order="descending" data-type="number"/>
    <xsl:variable name="studRef" select="@studentRef"/>
    <tr>
      <td><xsl:value-of select="@transcriptID"/></td>
      <td><xsl:value-of select="@studentRef"/></td>
      <td>

<xsl:value-of select="//Student[@studentID=$studRef]/PersonalInfo/Name"/>
      </td>
      <td><xsl:value-of select="GPA"/></td>
      <td><xsl:value-of select="AcademicTerm"/></td>
    </tr>
  </xsl:for-each>
</table>
```

```
            <!-- Loop #3 with conditional display -->
            <h2>Student Contact Directory</h2>
            <ul>
               <xsl:for-each select="Student">
                  <xsl:sort select="PersonalInfo/Name"/>
                  <li>
                     <strong><xsl:value-of select="@studentID"/></strong> -
                     <xsl:value-of select="PersonalInfo/Name"/>
                     (<xsl:value-of select="PersonalInfo/ContactEmail"/>)
                     <!-- (iv) xsl:if - conditional display -->
                     <xsl:if test="EnrollmentInfo/YearOfStudy &gt; 3">
                        <em> - Senior Student</em>
                     </xsl:if>
                  </li>
               </xsl:for-each>
            </ul>
         </body>
      </html>
   </xsl:template>

   <!-- (i) Student template for xsl:apply-templates -->
   <xsl:template match="Student">
      <tr id="{@studentID}">
         <td><xsl:value-of select="@studentID"/></td>
         <td><xsl:value-of select="PersonalInfo/Name"/></td>
```

```
            <td><xsl:value-of select="PersonalInfo/DateOfBirth"/></td>
            <td><xsl:value-of select="PersonalInfo/Gender"/></td>
            <td><xsl:value-of select="EnrollmentInfo/Program"/></td>
            <td><xsl:value-of select="EnrollmentInfo/YearOfStudy"/></td>
            <td>
              <!-- (iv) xsl:choose - conditional formatting -->
              <xsl:choose>
                <xsl:when test="@enrollmentStatus='Active'">
                  <span class="badge badge-active">Active</span>
                </xsl:when>
                <xsl:otherwise>
                  <span class="badge badge-inactive">
                    <xsl:value-of select="@enrollmentStatus"/>
                  </span>
                </xsl:otherwise>
              </xsl:choose>
            </td>
          </tr>
        </xsl:template>


    </xsl:stylesheet>
```

4.1 XSL (HTML) Output



Student Information System

University Code:MMU001
Academic Year:2025

Students Summary Table

ID	Name	Date of Birth	Gender	Program	Year	Status
S1211101093	Eddie Chin	2002-05-15	Male	Mathematics	1	Active
S1211107989	Lincoln Law	2003-11-08	Male	Engineering	4	Active
S1211108635	Tan Joon Shen	2004-07-19	Male	Computer Science	3	Inactive
S1221305518	Tan Yao Wei	2000-03-22	Male	Business Administration	2	Active

Detailed Student Information

S1211101093 - Eddie Chin Active

Personal Information

Full Name: Eddie Chin
Date of Birth: 2002-05-15
Gender: Male
Contact Email: 1211101093@student.mmu.edu.my
Nationality: Malaysian
Religion: Buddhism

Contact Information

Phone Number: 034-7456
Address: Room 103, Multimedia University

Detailed Student Information

S1211101093 - Eddie Chin Active

Personal Information

Full Name: Eddie Chin
Date of Birth: 2002-05-15
Gender: Male
Contact Email: 1211101093@student.mmu.edu.my
Nationality: Malaysian
Religion: Buddhism

Contact Information

Phone Number: 034-7456
Address: Room 103, Multimedia University

Parent Information

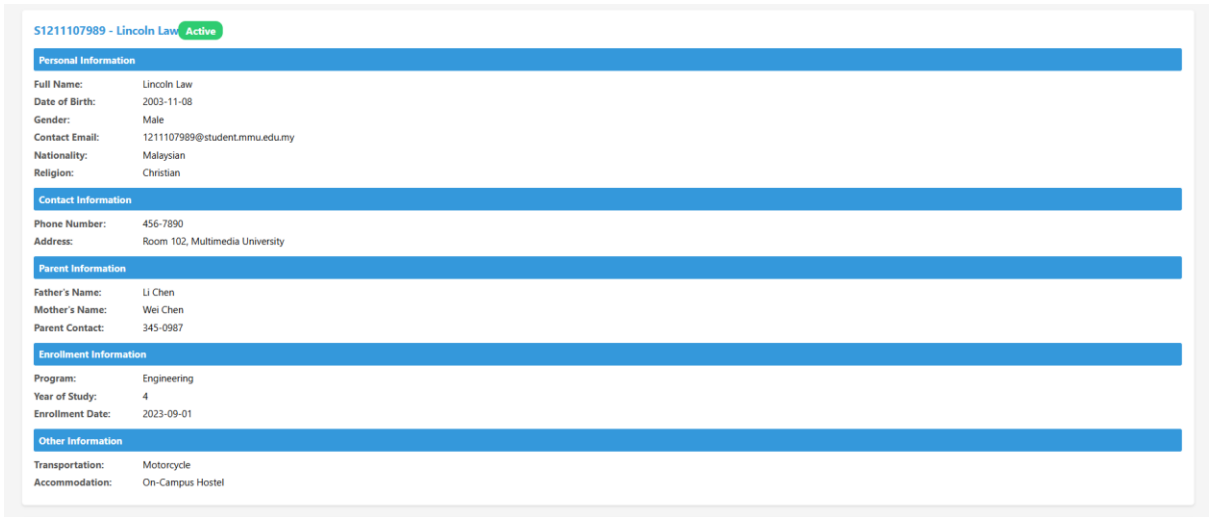
Father's Name: Chin Long
Mother's Name: Mei Ren
Parent Contact: 555-2003

Enrollment Information

Program: Mathematics
Year of Study: 1
Enrollment Date: 2024-09-01

Other Information

Transportation: Public Transport
Accommodation: Off-Campus Rental



S1211107989 - Lincoln Law Active

Personal Information

Full Name: Lincoln Law
Date of Birth: 2003-11-08
Gender: Male
Contact Email: 1211107989@student.mmu.edu.my
Nationality: Malaysian
Religion: Christian

Contact Information

Phone Number: 456-7890
Address: Room 102, Multimedia University

Parent Information

Father's Name: Li Chen
Mother's Name: Wei Chen
Parent Contact: 345-0987

Enrollment Information

Program: Engineering
Year of Study: 4
Enrollment Date: 2023-09-01

Other Information

Transportation: Motorcycle
Accommodation: On-Campus Hostel

S1211108635 - Tan Joon Shen Inactive

Personal Information

Full Name:	Tan Joon Shen
Date of Birth:	2004-07-19
Gender:	Male
Contact Email:	1211108635@student.mmu.edu.my
Nationality:	Malaysian
Religion:	Islam

Contact Information

Phone Number:	123-1000
Address:	Room 100, Multimedia University

Parent Information

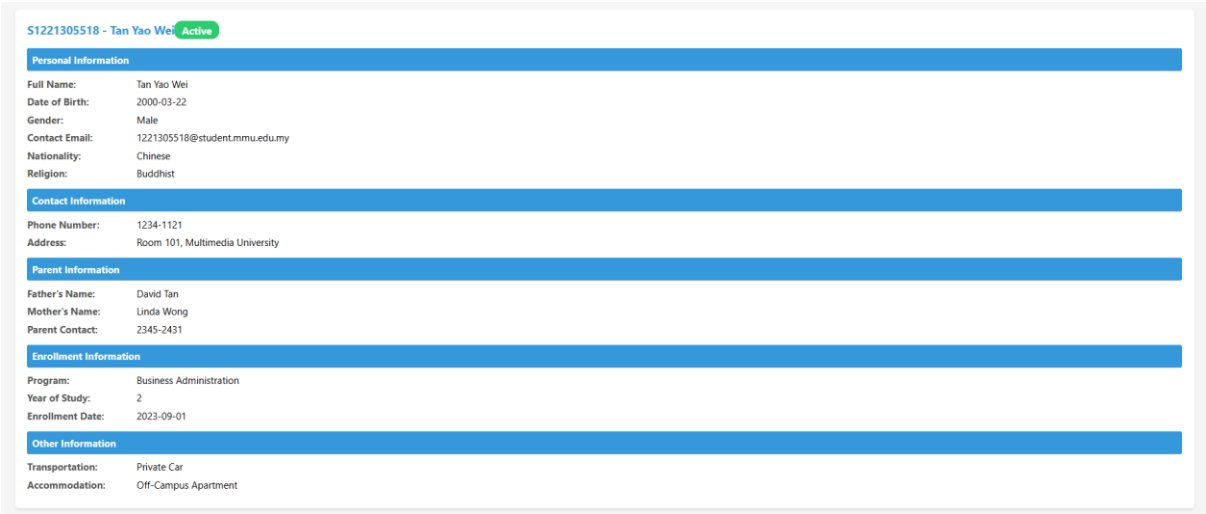
Father's Name:	Tan Joon Kok
Mother's Name:	Mary Wong
Parent Contact:	1234-2000

Enrollment Information

Program:	Computer Science
Year of Study:	3
Enrollment Date:	2026-01-01

Other Information

Transportation:	University Bus
Accommodation:	On-Campus Hostel



S1221305518 - Tan Yao Wei Active

Personal Information

Full Name:	Tan Yao Wei
Date of Birth:	2000-03-22
Gender:	Male
Contact Email:	1221305518@student.mmu.edu.my
Nationality:	Chinese
Religion:	Buddhist

Contact Information

Phone Number:	1234-1121
Address:	Room 101, Multimedia University

Parent Information

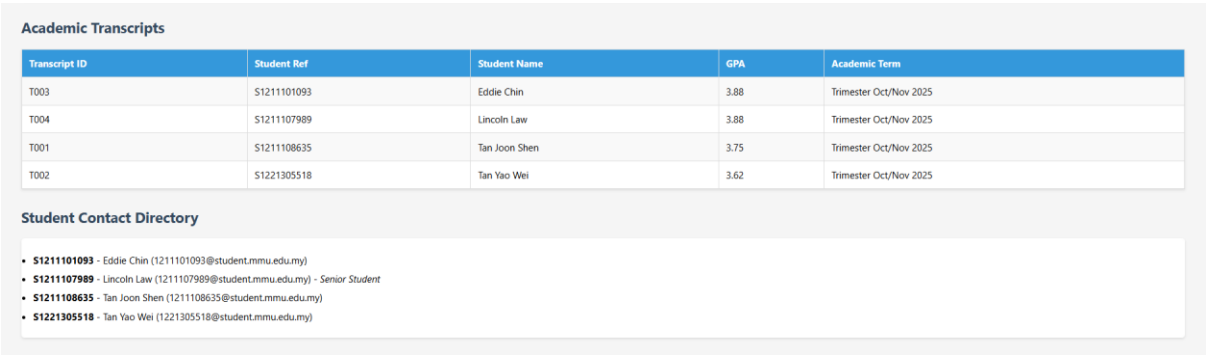
Father's Name:	David Tan
Mother's Name:	Linda Wong
Parent Contact:	2345-2431

Enrollment Information

Program:	Business Administration
Year of Study:	2
Enrollment Date:	2023-09-01

Other Information

Transportation:	Private Car
Accommodation:	Off-Campus Apartment



Academic Transcripts

Transcript ID	Student Ref	Student Name	GPA	Academic Term
T003	S1211101093	Eddie Chin	3.88	Trimester Oct/Nov 2025
T004	S1211107989	Lincoln Law	3.88	Trimester Oct/Nov 2025
T001	S1211108635	Tan Joon Shen	3.75	Trimester Oct/Nov 2025
T002	S1221305518	Tan Yao Wei	3.62	Trimester Oct/Nov 2025

Student Contact Directory

- **S1211101093** - Eddie Chin (1211101093@student.mmu.edu.my)
- **S1211107989** - Lincoln Law (1211107989@student.mmu.edu.my) - *Senior Student*
- **S1211108635** - Tan Joon Shen (1211108635@student.mmu.edu.my)
- **S1221305518** - Tan Yao Wei (1221305518@student.mmu.edu.my)

5.0 Modify XML Portion

(a) Java Program

```
import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.transform.OutputKeys;
import javax.xml.transform.Transformer;
import javax.xml.transform.TransformerFactory;
import javax.xml.transform.dom.DOMSource;
import javax.xml.transform.stream.StreamResult;

import org.w3c.dom.Document;
import org.w3c.dom.Element;
import org.w3c.dom.NodeList;

import java.io.File;

/**
 * ModifyStudentXML - Java Program to Modify XML Structure
 *
 * This program performs the following modifications:
 * 1. Adds ContactInformation (PhoneNumber and Address) to each Student
 * 2. Adds CurrentGPA to each Student's EnrollmentInfo
 * 3. Adds Nationality, Religion, ParentInfo, Transportation, Accommodation, Notes
 * 4. Outputs the modified XML to console and file
 */
public class ModifyStudentXML {

   public static void main(String[] args) {
      String inputFile = (args.length > 0) ? args[0] : "StudentSystem.xml";
      String outputFile = (args.length > 1) ? args[1] : "StudentSystem_Modified.xml";

      System.out.println("====================================");
      System.out.println("   XML MODIFICATION PROGRAM");
      System.out.println("====================================");
      System.out.println("Input File:  " + inputFile);
      System.out.println("Output File: " + outputFile);
      System.out.println("====================================\n");

      try {
         // Parse the original XML
         Document doc = parseXml(inputFile);
         System.out.println("[INFO] XML file parsed successfully.");

         // Modify the XML structure
         modifyXml(doc);
         System.out.println("[INFO] XML modifications completed.\n");
```

```
        // Display updated XML to console
        System.out.println("=== UPDATED XML STATEMENT ===\n");
        outputXml(doc, new StreamResult(System.out));
        System.out.println("\n============================\n");

        // Save to output file
        outputXml(doc, new StreamResult(new File(outputFile)));
        System.out.println("[SUCCESS] Modified XML saved as: " + outputFile);

    } catch (Exception e) {
        System.err.println("[ERROR] An error occurred during XML processing:");
        e.printStackTrace();
    }
}

/**
 * Parse XML file and return Document object
 */
private static Document parseXml(String file) throws Exception {
    DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
    DocumentBuilder builder = factory.newDocumentBuilder();
    Document doc = builder.parse(new File(file));
    doc.getDocumentElement().normalize();
    return doc;
}

/**
 * Modify XML structure by adding new elements
 */
private static void modifyXml(Document doc) {
    NodeList students = doc.getElementsByTagName("Student");
    int studentCount = students.getLength();

    System.out.println("[INFO] Found " + studentCount + " student(s) to modify.");
    System.out.println("[INFO] Adding new child elements to each student:");
    System.out.println("     - ContactInformation (if missing)");
    System.out.println("     - Nationality (if missing)");
    System.out.println("     - Religion (if missing)");
    System.out.println("     - ParentInfo (if missing)");
    System.out.println("     - Transportation (if missing)");
    System.out.println("     - Accommodation (if missing)");
    System.out.println("     - Notes (if missing)");
    System.out.println("     - CurrentGPA (if missing)\n");

    String[] nationalities = {"Malaysian", "Chinese", "Malaysian", "Malaysian"};
    String[] religions = {"Buddhism", "Christian", "Islam", "Buddhism"};
    String[][] parents = {
        {"Chin Long", "Mei Ren"},
        {"Tan Joon Kok", "Mary Wong"},
```

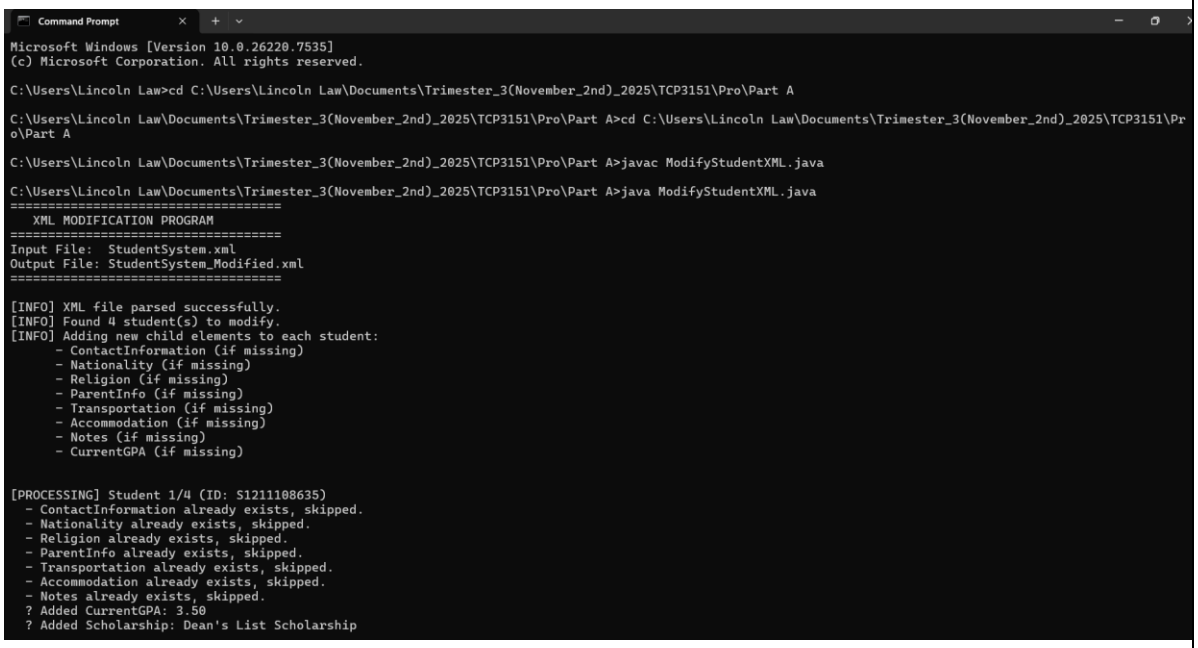
```
        {"Li Chen", "Wei Chen"},
        {"David Tan", "Linda Wong"}
    };
    String[] transports = {"University Bus", "Private Car", "Motorcycle", "Public
Transport"};
    String[] accommodations = {"On-Campus Hostel", "Off-Campus Apartment", "On-
Campus Hostel", "Off-Campus Rental"};
    String[] scholarships = {"Dean's List Scholarship", "Merit Scholarship",
"Excellence Scholarship", "None"};

    for (int i = 0; i < studentCount; i++) {
        Element student = (Element) students.item(i);
        String studentID = student.getAttribute("studentID");

        System.out.println("\n[PROCESSING] Student " + (i+1) + "/" + studentCount +
                " (ID: " + studentID + ")");

        // Add ContactInformation if not exists
        if (student.getElementsByTagName("ContactInformation").getLength() == 0) {
            Element contact = doc.createElement("ContactInformation");

            Element phone = doc.createElement("PhoneNumber");
            phone.setTextContent("555-" + (1000 + i));
            contact.appendChild(phone);

            Element address = doc.createElement("Address");
            address.setTextContent("Room " + (100 + i) + ", University Street");
            contact.appendChild(address);

            student.appendChild(contact);
            System.out.println("  ✓ Added ContactInformation");
        } else {
            System.out.println("  - ContactInformation already exists, skipped.");
        }

        // Add Nationality if not exists
        if (student.getElementsByTagName("Nationality").getLength() == 0) {
            Element nationality = doc.createElement("Nationality");
            nationality.setTextContent(nationalities[i % nationalities.length]);
            student.appendChild(nationality);
            System.out.println("  ✓ Added Nationality: " + nationalities[i %
nationalities.length]);
        } else {
            System.out.println("  - Nationality already exists, skipped.");
        }

        // Add Religion if not exists
        if (student.getElementsByTagName("Religion").getLength() == 0) {
            Element religion = doc.createElement("Religion");
```

```
            religion.setTextContent(religions[i % religions.length]);
            student.appendChild(religion);
            System.out.println("  ✓ Added Religion: " + religions[i % religions.length]);
        } else {
            System.out.println("  - Religion already exists, skipped.");
        }

        // Add ParentInfo if not exists
        if (student.getElementsByTagName("ParentInfo").getLength() == 0) {
            Element parentInfo = doc.createElement("ParentInfo");

            Element fatherName = doc.createElement("FatherName");
            fatherName.setTextContent(parents[i % parents.length][0]);
            parentInfo.appendChild(fatherName);

            Element motherName = doc.createElement("MotherName");
            motherName.setTextContent(parents[i % parents.length][1]);
            parentInfo.appendChild(motherName);

            Element parentContact = doc.createElement("ParentContact");
            parentContact.setTextContent("555-" + (2000 + i));
            parentInfo.appendChild(parentContact);

            student.appendChild(parentInfo);
            System.out.println("  ✓ Added ParentInfo");
        } else {
            System.out.println("  - ParentInfo already exists, skipped.");
        }

        // Add Transportation if not exists
        if (student.getElementsByTagName("Transportation").getLength() == 0) {
            Element transportation = doc.createElement("Transportation");
            transportation.setTextContent(transports[i % transports.length]);
            student.appendChild(transportation);
            System.out.println("  ✓ Added Transportation: " + transports[i %
transports.length]);
        } else {
            System.out.println("  - Transportation already exists, skipped.");
        }

        // Add Accommodation if not exists
        if (student.getElementsByTagName("Accommodation").getLength() == 0) {
            Element accommodation = doc.createElement("Accommodation");
            accommodation.setTextContent(accommodations[i %
accommodations.length]);
            student.appendChild(accommodation);
            System.out.println("  ✓ Added Accommodation: " + accommodations[i %
accommodations.length]);
```

```
        } else {
            System.out.println("  - Accommodation already exists, skipped.");
        }

        // Add Notes if not exists
        if (student.getElementsByTagName("Notes").getLength() == 0) {
            Element notes = doc.createElement("Notes");
            student.appendChild(notes);
            System.out.println("  ✓ Added Notes (empty element)");
        } else {
            System.out.println("  - Notes already exists, skipped.");
        }

        // Add CurrentGPA to EnrollmentInfo
        Element enrollment = (Element)
student.getElementsByTagName("EnrollmentInfo").item(0);
        if (enrollment != null &&
enrollment.getElementsByTagName("CurrentGPA").getLength() == 0) {
            Element gpa = doc.createElement("CurrentGPA");
            double gpaValue = 3.50 + (i * 0.1);
            gpa.setTextContent(String.format("%.2f", gpaValue));
            enrollment.appendChild(gpa);
            System.out.println("  ✓ Added CurrentGPA: " + gpa.getTextContent());
        } else {
            System.out.println("  - CurrentGPA already exists or EnrollmentInfo not
found, skipped.");
        }

        // Add Scholarship if not exists
        if (student.getElementsByTagName("Scholarship").getLength() == 0) {
            Element scholarship = doc.createElement("Scholarship");
            scholarship.setTextContent(scholarships[i % scholarships.length]);
            student.appendChild(scholarship);
            System.out.println("  ✓ Added Scholarship: " + scholarships[i %
scholarships.length]);
        } else {
            System.out.println("  - Scholarship already exists, skipped.");
        }
    }

    System.out.println("\n[INFO] Total modifications completed successfully!");
    System.out.println("[INFO] Each Student now has 10 direct child elements:");
    System.out.println("     1. PersonalInfo");
    System.out.println("     2. EnrollmentInfo (with CurrentGPA)");
    System.out.println("     3. AcademicRecord (EMPTY)");
    System.out.println("     4. ContactInformation");
    System.out.println("     5. Nationality");
    System.out.println("     6. Religion");
    System.out.println("     7. ParentInfo");
```

```
        System.out.println("     8. Transportation");
        System.out.println("     9. Accommodation");
        System.out.println("     10. Notes (EMPTY)");
        System.out.println("     11. Scholarship");
    }

    /**
     * Output XML document to specified result (console or file)
     */
    private static void outputXml(Document doc, StreamResult result) throws
Exception {
        Transformer transformer =
TransformerFactory.newInstance().newTransformer();

        // Set output properties for pretty printing
        transformer.setOutputProperty(OutputKeys.INDENT, "yes");
        transformer.setOutputProperty(OutputKeys.ENCODING, "UTF-8");
        transformer.setOutputProperty(OutputKeys.OMIT_XML_DECLARATION, "no");
        transformer.setOutputProperty("{http://xml.apache.org/xslt}indent-amount",
"4");

        // Transform and output
        transformer.transform(new DOMSource(doc), result);
    }
}
```

(b) Screenshot of the XML statement generated in the command prompt



```
[PROCESSING] Student 2/4 (ID: S1221305518)
  - ContactInformation already exists, skipped.
  - Nationality already exists, skipped.
  - Religion already exists, skipped.
  - ParentInfo already exists, skipped.
  - Transportation already exists, skipped.
  - Accommodation already exists, skipped.
  - Notes already exists, skipped.
  ? Added CurrentGPA: 3.60
  ? Added Scholarship: Merit Scholarship

[PROCESSING] Student 3/4 (ID: S1211107989)
  - ContactInformation already exists, skipped.
  - Nationality already exists, skipped.
  - Religion already exists, skipped.
  - ParentInfo already exists, skipped.
  - Transportation already exists, skipped.
  - Accommodation already exists, skipped.
  - Notes already exists, skipped.
  ? Added CurrentGPA: 3.70
  ? Added Scholarship: Excellence Scholarship

[PROCESSING] Student 4/4 (ID: S1211101093)
  - ContactInformation already exists, skipped.
  - Nationality already exists, skipped.
  - Religion already exists, skipped.
  - ParentInfo already exists, skipped.
  - Transportation already exists, skipped.
  - Accommodation already exists, skipped.
  - Notes already exists, skipped.
  ? Added CurrentGPA: 3.80
  ? Added Scholarship: None
```


Command Prompt
[INFO] Total modifications completed successfully!
[INFO] Each Student now has 10 direct child elements:
1. PersonalInfo
2. EnrollmentInfo (with CurrentGPA)
3. AcademicRecord (EMPTY)
4. ContactInformation
5. Nationality
6. Religion
7. ParentInfo
8. Transportation
9. Accommodation
10. Notes (EMPTY)
11. Scholarship
[INFO] XML modifications completed.
=== UPDATED XML STATEMENT ===
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<?xml-stylesheet type="text/xsl" href="StudentSystem.xsl"?><!-- ========== HIGHLIGHTED REQUIREMENTS ========== --><!-- (a) At least 10 DIRECT child elements under Student: --><!-- 1. PersonalInfo --><!-- 2. EnrollmentInfo --><!-- 3. AcademicRecord (EMPTY) --><!-- 4. ContactInformation --><!-- 5. Nationality (NEW) --><!-- 6. Religion (NEW) --><!-- 7. ParentInfo (NEW) --><!-- 8. Transportation (NEW) --><!-- 9. Accommodation (NEW) --><!-- 10. Notes (EMPTY) --><!-- (b) At least 1 empty element: AcademicRecord, CompletedCourses, Notes --><!-- (c) At least 2 attributes: studentID, enrollmentStatus, transcriptID, studentRef, universityCode, academicYear --><StudentInformationSystem academicYear="2025" universityCode="MMU001">
<Student enrollmentStatus="Inactive" studentID="S1211108635">
<PersonalInfo>
<Name>Tan Joon Shen</Name>
<DateOfBirth>2004-07-19</DateOfBirth>
<Gender>Male</Gender>
<ContactEmail>1211108635@student.mmu.edu.my</ContactEmail>
</PersonalInfo>
<EnrollmentInfo>
<Program>Computer Science</Program>
<YearOfStudy>3</YearOfStudy>

(c) XML output generated from Java Program

```
=== UPDATED XML STATEMENT ===

<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<?xml-stylesheet type="text/xsl" href="StudentSystem.xsl"?><!-- ========== HIGHLIGHTED REQUIREMENTS ========== --><!-- (a) At least 10 DIRECT child elements
 under Student: --><!--     1. PersonalInfo --><!--     2. EnrollmentInfo --><!--     3. AcademicRecord (EMPTY) --><!--     4. ContactInformation --><!--
  5. Nationality (NEW) --><!--     6. Religion (NEW) --><!--     7. ParentInfo (NEW) --><!--     8. Transportation (NEW) --><!--     9. Accommodation (NEW)
--><!--     10. Notes (EMPTY) --><!-- (b) At least 1 empty element: AcademicRecord, CompletedCourses, Notes --><!-- (c) At least 2 attributes: studentID, en
rollmentStatus, transcriptID, studentRef, universityCode, academicYear --><StudentInformationSystem academicYear="2025" universityCode="MMU001">

    <Student enrollmentStatus="Inactive" studentID="S1211108635">

        <PersonalInfo>

            <Name>Tan Joon Shen</Name>

            <DateOfBirth>2004-07-19</DateOfBirth>

            <Gender>Male</Gender>

            <ContactEmail>1211108635@student.mmu.edu.my</ContactEmail>

        </PersonalInfo>

        <EnrollmentInfo>

            <Program>Computer Science</Program>

            <YearOfStudy>3</YearOfStudy>

            <EnrollmentDate>2026-01-01</EnrollmentDate>

            <CurrentGPA>3.50</CurrentGPA>
        </EnrollmentInfo>

        <AcademicRecord/>

        <ContactInformation>

            <PhoneNumber>123-1000</PhoneNumber>

            <Address>Room 100, Multimedia University</Address>

        </ContactInformation>
```

```
        <Nationality>Malaysian</Nationality>

        <Religion>Islam</Religion>

        <ParentInfo>

            <FatherName>Tan Joon Kok</FatherName>

            <MotherName>Mary Wong</MotherName>

            <ParentContact>1234-2000</ParentContact>

        </ParentInfo>

        <Transportation>University Bus</Transportation>

        <Accommodation>On-Campus Hostel</Accommodation>

        <Notes/>

        <Scholarship>Dean's List Scholarship</Scholarship>
    </Student>

    <Student enrollmentStatus="Active" studentID="S1221305518">

        <PersonalInfo>

            <Name>Tan Yao Wei</Name>

            <DateOfBirth>2000-03-22</DateOfBirth>

            <Gender>Male</Gender>

            <ContactEmail>1221305518@student.mmu.edu.my</ContactEmail>

        </PersonalInfo>

        <EnrollmentInfo>

            <Program>Business Administration</Program>

            <YearOfStudy>2</YearOfStudy>
```

```
            <EnrollmentDate>2023-09-01</EnrollmentDate>
            <CurrentGPA>3.60</CurrentGPA>
        </EnrollmentInfo>
        <AcademicRecord/>
        <ContactInformation>
            <PhoneNumber>1234-1121</PhoneNumber>
            <Address>Room 101, Multimedia University</Address>
        </ContactInformation>
        <Nationality>Chinese</Nationality>
        <Religion>Buddhist</Religion>
        <ParentInfo>
            <FatherName>David Tan</FatherName>
            <MotherName>Linda Wong</MotherName>
            <ParentContact>2345-2431</ParentContact>
        </ParentInfo>
        <Transportation>Private Car</Transportation>
        <Accommodation>Off-Campus Apartment</Accommodation>

        <Notes/>
        <Scholarship>Merit Scholarship</Scholarship>
    </Student>
    <Student enrollmentStatus="Active" studentID="S1211107989">
        <PersonalInfo>
            <Name>Lincoln Law</Name>
```

```
            <DateOfBirth>2003-11-08</DateOfBirth>
            <Gender>Male</Gender>
            <ContactEmail>1211107989@student.mmu.edu.my</ContactEmail>
        </PersonalInfo>
        <EnrollmentInfo>
            <Program>Engineering</Program>
            <YearOfStudy>4</YearOfStudy>
            <EnrollmentDate>2023-09-01</EnrollmentDate>
            <CurrentGPA>3.70</CurrentGPA>
        </EnrollmentInfo>
        <AcademicRecord/>
        <ContactInformation>
            <PhoneNumber>456-7890</PhoneNumber>
            <Address>Room 102, Multimedia University</Address>
        </ContactInformation>
        <Nationality>Malaysian</Nationality>
        <Religion>Christian</Religion>
        <ParentInfo>
            <FatherName>Li Chen</FatherName>
            <MotherName>Wei Chen</MotherName>
            <ParentContact>345-0987</ParentContact>
        </ParentInfo>
        <Transportation>Motorcycle</Transportation>
```

```
        <Accommodation>On-Campus Hostel</Accommodation>
        <Notes/>
        <Scholarship>Excellence Scholarship</Scholarship>
    </Student>
    <Student enrollmentStatus="Active" studentID="S1211101093">
        <PersonalInfo>
            <Name>Eddie Chin</Name>
            <DateOfBirth>2002-05-15</DateOfBirth>
            <Gender>Male</Gender>
            <ContactEmail>1211101093@student.mmu.edu.my</ContactEmail>
        </PersonalInfo>
        <EnrollmentInfo>
            <Program>Mathematics</Program>
            <YearOfStudy>1</YearOfStudy>
            <EnrollmentDate>2024-09-01</EnrollmentDate>
            <CurrentGPA>3.80</CurrentGPA>
        </EnrollmentInfo>
        <AcademicRecord/>
        <ContactInformation>
            <PhoneNumber>034-7456</PhoneNumber>
            <Address>Room 103, Multimedia University</Address>
        </ContactInformation>
        <Nationality>Malaysian</Nationality>
```

```
        <Religion>Buddhism</Religion>

        <ParentInfo>
            <FatherName>Chin Long</FatherName>
            <MotherName>Mei Ren</MotherName>
            <ParentContact>555-2003</ParentContact>
        </ParentInfo>
        <Transportation>Public Transport</Transportation>
        <Accommodation>Off-Campus Rental</Accommodation>
        <Notes/>
        <Scholarship>None</Scholarship>
    </Student>
    <Transcript studentRef="S1211108635" transcriptID="T001">
        <GPA>3.75</GPA>
        <CompletedCourses/>
        <AcademicTerm>Trimester Oct/Nov 2025</AcademicTerm>
    </Transcript>
    <Transcript studentRef="S1221305518" transcriptID="T002">
        <GPA>3.62</GPA>
        <CompletedCourses/>
        <AcademicTerm>Trimester Oct/Nov 2025</AcademicTerm>
    </Transcript>
```

```
    <Transcript studentRef="S1211101093" transcriptID="T003">
        <GPA>3.88</GPA>
        <CompletedCourses/>
        <AcademicTerm>Trimester Oct/Nov 2025</AcademicTerm>
    </Transcript>
    <Transcript studentRef="S1211107989" transcriptID="T004">
        <GPA>3.88</GPA>
        <CompletedCourses/>
        <AcademicTerm>Trimester Oct/Nov 2025</AcademicTerm>
    </Transcript>
</StudentInformationSystem>
============================
[SUCCESS] Modified XML saved as: StudentSystem_Modified.xml
C:\Users\Lincoln Law\Documents\Trimester_3(November_2nd)_2025\TCP3151\Pro\Part A>
```

Part B

Question 1

Create JNI program to calculate the Body Mass Index (BMI). The Java main program will request user to provide the weight(kg) and height (cm).

- First native method:
  - Input: weight(kg) and height(cm)
  - Convert the height from cm to m
  - BMI = weight(kg) / [height(m)]2
  - Output: BMI value
- Second native method:
  - Input: BMI value
  - Calculate the BMI category based on the table below:

Category	BMI range - kg/m^2
Severe Thinness	< 16
Moderate Thinness	16 - 17
Mild Thinness	17 - 18.5
Normal	18.5 - 25
Overweight	25 - 30
Obese Class I	30 - 35
Obese Class II	35 - 40
Obese Class III	> 40

Picture source: https://www.calculator.net/bmi-calculator.html?ctype=metric&cage=46&csex=m&cheightfeet=5&cheightinch=10&cpound=160&cheightmeter=158&ckg=49&printit=0

  - Output: BMI category
- Java main function displays the BMI value and category.
- Bonus will be given if student provide additional feature that is attractive and useful.

BMICalculator.java

```
import java.util.Scanner;


public class BMICalculator {

   static {

      System.loadLibrary("BMICalculator");

   }


   public native double calculateBMI(double weight, double height);
```

```
    public native String getCategory(double bmi);

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        BMICalculator app = new BMICalculator();

        System.out.println("=== BMI Calculator ===");

        System.out.print("Enter weight (kg): ");
        double weight = scanner.nextDouble();
        System.out.print("Enter height (cm): ");
        double height = scanner.nextDouble();

        double bmi = app.calculateBMI(weight, height);
        String category = app.getCategory(bmi);

        System.out.printf("BMI: %.2f\nCategory: %s\n", bmi, category);

        double hM = height / 100.0;
        System.out.printf("[Tip] Ideal weight range: %.1fkg - %.1fkg\n",
                18.5 * hM * hM, 25.0 * hM * hM);
    }
}
```

BMICalculator.cpp

```
#include <jni.h>
#include <iostream>
#include "BMICalculator.h"

JNIEXPORT jdouble JNICALL Java_BMICalculator_calculateBMI
  (JNIEnv *env, jobject obj, jdouble weight, jdouble height) {
    double hM = height / 100.0;
    return (hM > 0) ? weight / (hM * hM) : 0.0;
}

JNIEXPORT jstring JNICALL Java_BMICalculator_getCategory
  (JNIEnv *env, jobject obj, jdouble bmi) {
    const char *result;
    if (bmi < 16.0) result = "Severe Thinness";
    else if (bmi < 17.0) result = "Moderate Thinness";
    else if (bmi < 18.5) result = "Mild Thinness";
    else if (bmi < 25.0) result = "Normal";
    else if (bmi < 30.0) result = "Overweight";
    else if (bmi < 35.0) result = "Obese Class I";
    else if (bmi < 40.0) result = "Obese Class II";
    else result = "Obese Class III";

    return env->NewStringUTF(result);
}
```

Output of the BMI Calculator

Severe Thinness

```
C:\Users\User\Desktop\mmu\Year 3\Trimester 1\Integrative Programming and Technologies\Project Guideline-20251114>java BM
ICalculator
WARNING: A restricted method in java.lang.System has been called
WARNING: java.lang.System::loadLibrary has been called by BMICalculator in an unnamed module (file:/C:/Users/User/Deskto
p/mmu/Year%203/Trimester%201/Integrative%20Programming%20and%20Technologies/Project%20Guideline-20251114/)
WARNING: Use --enable-native-access=ALL-UNNAMED to avoid a warning for callers in this module
WARNING: Restricted methods will be blocked in a future release unless native access is enabled

=== BMI Calculator ===
Enter weight (kg): 43
Enter height (cm): 170
BMI: 14.88
Category: Severe Thinness
[Tip] Ideal weight range: 53.5kg - 72.3kg
```

Moderate Thinness

```
C:\Users\User\Desktop\mmu\Year 3\Trimester 1\Integrative Programming and Technologies\Project Guideline-20251114>java BM
ICalculator
WARNING: A restricted method in java.lang.System has been called
WARNING: java.lang.System::loadLibrary has been called by BMICalculator in an unnamed module (file:/C:/Users/User/Deskto
p/mmu/Year%203/Trimester%201/Integrative%20Programming%20and%20Technologies/Project%20Guideline-20251114/)
WARNING: Use --enable-native-access=ALL-UNNAMED to avoid a warning for callers in this module
WARNING: Restricted methods will be blocked in a future release unless native access is enabled

=== BMI Calculator ===
Enter weight (kg): 47.5
Enter height (cm): 170
BMI: 16.44
Category: Moderate Thinness
[Tip] Ideal weight range: 53.5kg - 72.3kg
```

Mild Thinness

```
C:\Users\User\Desktop\mmu\Year 3\Trimester 1\Integrative Programming and Technologies\Project Guideline-20251114>java BM
ICalculator
WARNING: A restricted method in java.lang.System has been called
WARNING: java.lang.System::loadLibrary has been called by BMICalculator in an unnamed module (file:/C:/Users/User/Deskto
p/mmu/Year%203/Trimester%201/Integrative%20Programming%20and%20Technologies/Project%20Guideline-20251114/)
WARNING: Use --enable-native-access=ALL-UNNAMED to avoid a warning for callers in this module
WARNING: Restricted methods will be blocked in a future release unless native access is enabled

=== BMI Calculator ===
Enter weight (kg): 52
Enter height (cm): 170
BMI: 17.99
Category: Mild Thinness
[Tip] Ideal weight range: 53.5kg - 72.3kg
```

Normal

```
C:\Users\User\Desktop\mmu\Year 3\Trimester 1\Integrative Programming and Technologies\Project Guideline-20251114>java BM
ICalculator
WARNING: A restricted method in java.lang.System has been called
WARNING: java.lang.System::loadLibrary has been called by BMICalculator in an unnamed module (file:/C:/Users/User/Deskto
p/mmu/Year%203/Trimester%201/Integrative%20Programming%20and%20Technologies/Project%20Guideline-20251114/)
WARNING: Use --enable-native-access=ALL-UNNAMED to avoid a warning for callers in this module
WARNING: Restricted methods will be blocked in a future release unless native access is enabled

=== BMI Calculator ===
Enter weight (kg): 65
Enter height (cm): 170
BMI: 22.49
Category: Normal
[Tip] Ideal weight range: 53.5kg - 72.3kg
```

Overweight

```
C:\Users\User\Desktop\mmu\Year 3\Trimester 1\Integrative Programming and Technologies\Project Guideline-20251114>java BM
ICalculator
WARNING: A restricted method in java.lang.System has been called
WARNING: java.lang.System::loadLibrary has been called by BMICalculator in an unnamed module (file:/C:/Users/User/Deskto
p/mmu/Year%203/Trimester%201/Integrative%20Programming%20and%20Technologies/Project%20Guideline-20251114/)
WARNING: Use --enable-native-access=ALL-UNNAMED to avoid a warning for callers in this module
WARNING: Restricted methods will be blocked in a future release unless native access is enabled

=== BMI Calculator ===
Enter weight (kg): 78
Enter height (cm): 170
BMI: 26.99
Category: Overweight
[Tip] Ideal weight range: 53.5kg - 72.3kg
```

Obese Class I

```
C:\Users\User\Desktop\mmu\Year 3\Trimester 1\Integrative Programming and Technologies\Project Guideline-20251114>java BM
ICalculator
WARNING: A restricted method in java.lang.System has been called
WARNING: java.lang.System::loadLibrary has been called by BMICalculator in an unnamed module (file:/C:/Users/User/Deskto
p/mmu/Year%203/Trimester%201/Integrative%20Programming%20and%20Technologies/Project%20Guideline-20251114/)
WARNING: Use --enable-native-access=ALL-UNNAMED to avoid a warning for callers in this module
WARNING: Restricted methods will be blocked in a future release unless native access is enabled

=== BMI Calculator ===
Enter weight (kg): 92
Enter height (cm): 170
BMI: 31.83
Category: Obese Class I
[Tip] Ideal weight range: 53.5kg - 72.3kg
```

Obese Class II

```
C:\Users\User\Desktop\mmu\Year 3\Trimester 1\Integrative Programming and Technologies\Project Guideline-20251114>java BM
ICalculator
WARNING: A restricted method in java.lang.System has been called
WARNING: java.lang.System::loadLibrary has been called by BMICalculator in an unnamed module (file:/C:/Users/User/Deskto
p/mmu/Year%203/Trimester%201/Integrative%20Programming%20and%20Technologies/Project%20Guideline-20251114/)
WARNING: Use --enable-native-access=ALL-UNNAMED to avoid a warning for callers in this module
WARNING: Restricted methods will be blocked in a future release unless native access is enabled

=== BMI Calculator ===
Enter weight (kg): 107
Enter height (cm): 170
BMI: 37.02
Category: Obese Class II
[Tip] Ideal weight range: 53.5kg - 72.3kg
```

Obese Class III

```
C:\Users\User\Desktop\mmu\Year 3\Trimester 1\Integrative Programming and Technologies\Project Guideline-20251114>java BM
ICalculator
WARNING: A restricted method in java.lang.System has been called
WARNING: java.lang.System::loadLibrary has been called by BMICalculator in an unnamed module (file:/C:/Users/User/Deskto
p/mmu/Year%203/Trimester%201/Integrative%20Programming%20and%20Technologies/Project%20Guideline-20251114/)
WARNING: Use --enable-native-access=ALL-UNNAMED to avoid a warning for callers in this module
WARNING: Restricted methods will be blocked in a future release unless native access is enabled

=== BMI Calculator ===
Enter weight (kg): 125
Enter height (cm): 170
BMI: 43.25
Category: Obese Class III
[Tip] Ideal weight range: 53.5kg - 72.3kg
```

Question 2

The Caesar Cipher technique is one of the earliest and simplest method of encryption technique. You may refer to the link (https://www.youtube.com/watch?v=o6TPx1Co_wg) to learn more about the Caesar Cipher. Create a JNI program that perform the Caesar Cipher.

- First native method:
  - Create a character array that content is provided from the user
  - Output: original character array
- Second native method:
  - Input: original character array
  - Perform the Ceaser Cipher encryption
  - Output: encrypted character array
- Third native method:
  - Input: encrypted character array
  - Perform the Ceaser Cipher decryption
  - Output: decrypted character array

CMD Output:

```
C:\Users\User\Desktop>javac -h . CC1.java

C:\Users\User\Desktop>g++ -I"%JAVA_HOME%\include" -I"%JAVA_HOME%\include\win32" -shared -o CCC1.dll CCC1.cpp

C:\Users\User\Desktop>java CC1
Enter a text: i love Dr.Chong and TCP
Enter Key Value: 3
--------------------------------

Original text: I LOVE DR.CHONG AND TCP
Ciphertext   : L ORYH GU.FKRQJ DQG WFS
Plaintext    : I LOVE DR.CHONG AND TCP
```

```
//CC1.java
import java.util.Scanner;

class CC1 {
   private native char[] createCharArray(char[] input);
   private native char[] encryptCharArray(char[] input, int shift);
   private native char[] decryptCharArray(char[] input, int shift);

   static {
      System.loadLibrary("CCC1");
   }

   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);

      System.out.print("Enter a text: ");
      String text = scanner.nextLine();

      System.out.print("Enter Key Value: ");
      int shift=scanner.nextInt();
      System.out.print("--------------------------------\n\n");

      CC1 p = new CC1();
      char[] result = p.createCharArray(text.toCharArray());

      System.out.printf("%-13s: %s%n", "Original text", new String(result));

      //Encrypt
      char[] cipher = p.encryptCharArray(result, shift);
      System.out.printf("%-13s: %s%n", "Ciphertext", new String(cipher));

      //Decrypt
      char[] plain = p.decryptCharArray(cipher, shift);
      System.out.printf("%-13s: %s%n", "Plaintext", new String(plain));

      scanner.close();
   }
}
```

```
//CCC1.cpp
#include "CC1.h"

//First native method
JNIEXPORT jcharArray JNICALL Java_CC1_createCharArray
  (JNIEnv *env, jobject obj, jcharArray input) {

   jsize len = env->GetArrayLength(input);
   jchar *chars = env->GetCharArrayElements(input, NULL);

   if (chars == NULL){
     return NULL;
```

```
    }

    // for small letter change to capital letter
    for (jsize i = 0; i < len; i++) {
        if (chars[i] >= 'a' && chars[i] <= 'z') {
            chars[i] = chars[i] - 'a' + 'A';
        }
    }

    jcharArray output = env->NewCharArray(len);
    if (output == NULL) {
        env->ReleaseCharArrayElements(input, chars, 0);
        return NULL;
    }
    env->SetCharArrayRegion(output, 0, len, chars);
    env->ReleaseCharArrayElements(input, chars, 0);
    return output;
}

//Second native method
JNIEXPORT jcharArray JNICALL Java_CC1_encryptCharArray
  (JNIEnv *env, jobject obj, jcharArray input, jint shift) {

    jsize len = env->GetArrayLength(input);
    jchar *chars = env->GetCharArrayElements(input, NULL);

    if (chars == NULL){
      return NULL;
    }

    //encryption algorithms
    for (jsize i = 0; i < len; i++) {
        if (chars[i] >= 'a' && chars[i] <= 'z') {
            chars[i] = (chars[i] - 'a' + shift) % 26 + 'a';
        }
        else if (chars[i] >= 'A' && chars[i] <= 'Z') {
            chars[i] = (chars[i] - 'A' + shift) % 26 + 'A';
        }
    }

    // Create a new Java char array of length 'len' to store the processed characters
    jcharArray output = env->NewCharArray(len);
    // Copy the processed characters from the C++ array 'chars' into the new Java
array 'output'
    // starting at index 0 and copying 'len' characters
    env->SetCharArrayRegion(output, 0, len, chars);
    // Release the native pointer 'chars' obtained from GetCharArrayElements
    // 0 indicates that any changes made to 'chars' will be committed back to the
original Java array
    env->ReleaseCharArrayElements(input, chars, 0);
```

```
    return output;
}

//Third native method
JNIEXPORT jcharArray JNICALL Java_CC1_decryptCharArray
  (JNIEnv *env, jobject obj, jcharArray input, jint shift) {

    jsize len = env->GetArrayLength(input);
    jchar *chars = env->GetCharArrayElements(input, NULL);

    if (chars == NULL){
      return NULL;
    }

    //decryption algorithms
    for (jsize i = 0; i < len; i++) {
        if (chars[i] >= 'a' && chars[i] <= 'z') {
            chars[i] = (chars[i] - 'a' - shift + 26) % 26 + 'a';
        }
        else if (chars[i] >= 'A' && chars[i] <= 'Z') {
            chars[i] = (chars[i] - 'A' - shift + 26) % 26 + 'A';
        }
    }

    jcharArray output = env->NewCharArray(len);
    env->SetCharArrayRegion(output, 0, len, chars);
    env->ReleaseCharArrayElements(input, chars, 0);
    return output;

}
```

Project Assessment Form

Criteria	Sub-criteria	Unsatisfactory (0 - 1)	Poor (2 - 4)	Adequate (5 - 7)	Good (8 - 10)	Mark	Remark
Documentation (10%)	Description	No documentation.	Document is formatted poorly.	Formatting of the document is generally consistent and adequate.	Formatting of the document is professional.		
Part A (30%)	XML and logical view diagram	XML statement has many mistakes. Does not meet the XML requirement. Does not draw logical view diagram.	XML statement has some mistakes. Meet less than 60% of XML requirements. Logical view diagram consists of many mistakes.	XML statements has few mistakes. Meet between 60 - 80% of XML requirements. Logical view diagram consists of less mistakes.	XML statement is correct and well- organized. Meet 80 - 100% of XML requirements. Logical view diagram is correct.		
	DTD portion	Wrong DTD syntax. Most of the elements and attributes are written in incorrect way. Missing attribute declaration. Does not meet the DTD requirement.	DTD syntax is not constructed properly. Some of the elements and attributes are written in correct way. Missing some of the attribute declaration. Meet less than 60% of DTD requirement.	DTD syntax is constructed properly. Most of the elements and attributes are written in correct way. Attribute is declared. Meet between 60 - 80% of DTD requirement.	DTD syntax is well constructed. All the elements and attributes are written in correct way. Attribute is declared properly. Meet between 80 - 100% of DTD requirement.		
	XSLT and Java program	XSLT and Java program are incorrect. Does not meet the XSLT and program requirements.	XSLT does not consist of loop structure. Java program has significant logic errors. Meet less than 60% of XSLT and program requirements.	XSLT consists of a loop structure. Java program has a few logic errors. Meet between 60 - 80% of XSLT and program requirements.	XSLT consists of more than one loop structure. Java program is logically well designed. Meet between 80 - 100% of XSLT and program requirements.		
Part B (30%)	JNI Q1	Program is incorrect. Native method is wrong. Does not meet JNI requirements.	Program has significant logic errors. Native method is moderate correct. Meet less than 60% of JNI requirements.	Program has slight logic errors that do no significantly affect the results. Native method is written correctly. Meet between 60 - 80% of JNI requirements.	Program is logically well designed. Native method is written excellently. Meet between 80 - 100% of JNI requirements.		
	JNI Q2	Program is incorrect. Native method is wrong. Does not meet JNI requirements.	Program has significant logic errors. Native method is moderate correct. Meet less than 60% of JNI requirements.	Program has slight logic errors that do no significantly affect the results. Native method is written correctly. Meet between 60 - 80% of JNI requirements.	Program is logically well designed. Native method is written excellently. Meet between 80 - 100% of JNI requirements.		
	Run time	Does not execute due to syntax or runtime errors (endless loop, crashes etc.). User prompts are misleading or non-existent. No testing.	Executes with some errors. User prompts contain little information, poor design. Some testing has been performed.	Executes without errors. User prompts are understandable. Thorough testing has been performed	Executes without errors. Excellent user prompts. Thorough and organized testing has been performed and output from test cases is included.		
Others (10%)	Question handling	Not able to answer the questions asked.	Provide moderate answer for the questions asked.	Provide correct answer for the question asked.	Provide excellence answer for the question asked.		

Mark for Project Assessment (40%) = _____