Part 1: Write a XML Schema Definition (XSD) file of the following XML document (without namespace) (15 points)

```
<?xml version="1.0" encoding="UTF-8"?>
<diaries>
    <!-- 'diaries' contain a sequence of 'diary' -->
    <!-- 'diaries' are stored sequentially by 'created_at' in ascending order -->
    <diary id="d100" created_at="2022-09-16T13:00:00"
                     updated_at="2022-09-16T13:05:00">
        <!-- 'diary'contains title, date, author, content and zero or more tags -->
        <!-- 'diary' has 'id' as its required ID,
                         'created_at' and 'updated_at' as required attributes -->
        <title>Cry is ok</title>
        <!-- 'date' contains year, month, day, hour and minute.-->
        <date>
            <year>2022</year>
            <month>9</month>
            <day>16</day>
            <hour>13</hour>
            <minute>0</minute>
        </date>
        <!-- 'author' contains firstname and lastname -->
        <author>
            <firstname>Amina</firstname>
            <lastname>Mehmood</lastname>
        </author>
        <content>
            It's okay to cry when there's too much on your mind.
            The clouds rain too when things get heavy.
        </content>
        <!-- 'tag' is optional and has maxOccurs is unbounded -->
        <tag>cloud</tag>
        <tag>cry</tag>
        <tag>mind</tag>
    </diary>
</diaries>
```

Part 2: Write full XML path expression of the following xml path expression (5 points)

2.1. /a/b[1]/c

2.2. /a/b[c]/@d

2.3. //a/b/../../c

2.4. /a/b[./c= "d"]/e

2.5. /a//b/@*

Part 3: Based on the structure of XML document shown in Part 1, write XML Path expressions to perform the following queries (10 points)

3.1. List authors who wrote diaries in 2022. (Remove the duplicate items)

3.2. Show the title of the last diary.

3.3. Show the content of the diary whose id is ‘d234’.

3.4. Count diaries by author “Amina Mehmood”.

3.5. Show the titles of diaries that contain at least three tags.

3.6. Show the ids of diaries whose title contains a word “Queen”.

3.7. List the contents of diaries that have ‘birthday’ tag or ‘valentine’ tag. Show only the string value of each content.

3.8. Show the dates of diaries that do not have any tag.

3.9. Let N_i be the number of tags in diary i. Show the maximum value of N_i in the document. (แสดงจำนวน tag สูงสุดที่มีอยู่ใน diary)

3.10. List authors of diaries from 2020 to 2021 with ‘bitcoin’ tag.

Syntax Page

XML Schema

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
    { element declaration }
</xs:schema>
```

Element declaration

```
<xs:element  name= "x" type= "y"  default="z"  fixed="y"  minOccurs="m" maxOccurs="n"   />
```

Attribute declaration

```
<xs:attribute  name= "x"  type= "y" use="optional (default) | prohibited | required" >
```

Declaring Element with compleType

```
<xs:element name="z">
    <complexType>
        { Context }
    </complexType>
</xs:element>
```

Where { Context } is

```
<compositor>
    { ElementList }
</compositor>
{ AttibuteList }
```

where <compositor> is <all>, <sequence> , <choice>

```
<sequence  minOccurs="m"  maxOccurs="n" > . . . </sequence>
```

XML Schema Data Types

- xs:ID
- xs:boolean
- xs:integer
- xs:decimal
- xs:string
- xs:duration
- xs:dateTime
- xs:date
- xs:time

XPath Syntax

```
/axes::node-test()[predicates]/axes::node-test()[predicates]
```

Axes	Node Test

1. ancestor
2. ancestor-or-self
3. attribute
4. child
5. descendant
6. descendant-or-self
7. following
8. following-sibling
9. namespace
10. parent
11. preceding
12. preceding-sibling
13. self
	• Node Kind Test
	  • node()
	  • document-node()
	  • element()
	  • text()
	  • comment()
	  • processing-instruction()
	  • attribute()
	• Node Name Test
	  • qualified-name e.g. tag, p:tag
	  • ns-prefix:* e.g. p:*
	  • *:local-name e.g. *:tag
	  • *

- **String Matching Functions**
  - contains( ) , starts-with( ) , ends-with( )
- **Functions on Sequences**
  - empty( ) , exists( ) , distinct-values( ) , count( ) , avg( ) , min( ) , max( )
- **Context Functions**
  - position( ) , last( )
- **Boolean Function/Operators**
  - not( ) , and , or