**Class Practice Questions**

**Question #1:**

You are designing an intruder detection system that has following modules (here module refers to one or more classes)

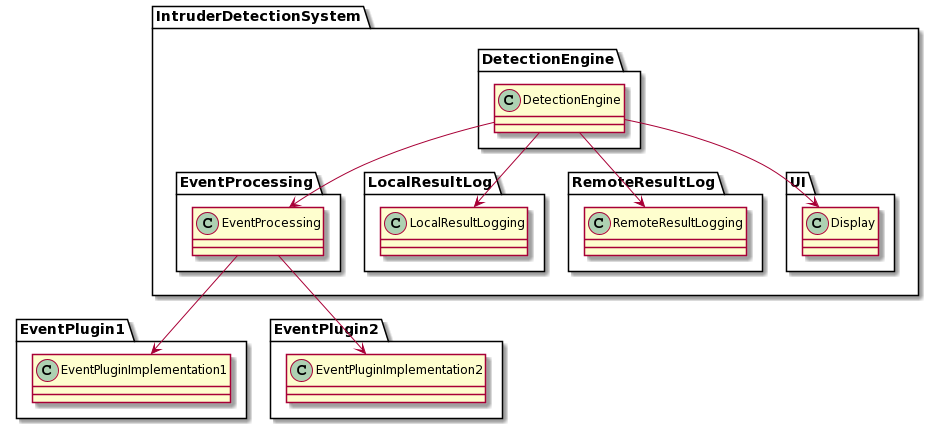
**EventProcessing**: This module processes events coming from EventPlugins and makes them available to the DetectionEngine module for analysis.

**EventPlugins**: These are external components that can be plugged into the system. They provide events which are then processed by the system to detect network intrusion.

**DetectionEngine**: This module implements actual algorithms that detect abnormal activity in the network based on available events.

**ResultLog**: These are two modules that log the detected intrusions locally on hard disk as well as send them remotely to the cloud.

A naive design is the one presented in the UML diagram below.



Your task is to improve this design keeping in mind SOLID design principles. You can add interfaces in any package if required.

After the improvement the design should reflect following characteristics:

1. High level modules should not depend on lower level modules.(hint: DetectionEngine is the highest level module, think about others yourself)
2. Interface with plugins to be designed such that new plugins can be added to the system without a need to make changes to the system
3. Currently there is only two types of logging available, the new design should support adding additional logging modules without impacting the DetectionEngine module
4. Show all relationships (e.g. inheritance, directional association between classes and interfaces)

Question #2:

You are building a software system that helps visualize the architecture of a house for the customers. Your system is a virtual reality system that accepts an architecture from a software developed by a third party

vendor (software supplier) and visualizes it as a virtual reality house. The customer can walk into the house to get a feel of it using the VR glasses. At this stage you are working with only one vendor and developing the system. The initial design of your system looks like below where your system “VirtualRealityCore” module interacts with the third party software “FastArchitect” module to extract architectural information.

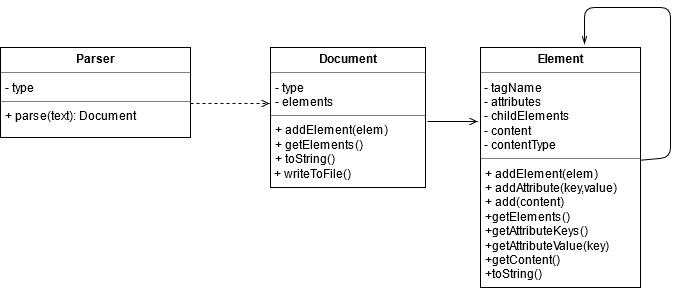
Note: In future you would be required to integrate with software developed by other vendors.

**Question #3:**

|  |  |
| --- | --- |
| <html>  <head></head>  <body>  <p class='text'> Hello world </p>  </body>  </html> | <book category='children'>  <title>Harry Potter</title>  <author>J K. Rowling</author>  <year>2005</year>  <price>29.99</price>  </book> |
| (a) HTML document representing a simple web page | (b) XML document representing a book's information |

HTML and XML are similar markup languages used to structure text: HTML is used for presentation of web content while XML is more general and can be used for representing any information. Both share similar rules and format and their samples are given above, though XML enforces the rules more strictly.

A HTML or XML document comprises of elements (defined by a start and matching end tag that are represented as angular brackets). Each element can contain: attributes (name-value pairs mentioned in the start tag) or body (content enclosed in the start and end tags). Body can be text, comment or further elements.



Consider a proposed UML class diagram for parsing and representing HTML/XML documents. **Parser** parses a given text based upon its type and produces a **Document** instance. **Document** contains 1 or more **Element** instances. XML documents require that one and only one element exists as a root, while there is no such constraint in HTML. A **Document** can be converted to its string (HTML or XML) representation using **toString()** function and so is any **Element**, which shall include content of its children also. Moreover, the **Document** can be written to a file also.

(a) Fill the following table to evaluate the above design in the light of SOLID principles

|  |  |  |
| --- | --- | --- |
| **Principle Name** | **Compliance**  **(yes/no)** | **Reason** |
| **S** |  |  |
| **O** |  |  |
| **L** |  |  |
| **I** |  |  |
| **D** |  |  |