Homework # 9

01286131 Object-Oriented Programming

Software Engineering Program,

Department of Computer Engineering,

School of Engineering, KMITL

By

65011277 Chanasorn Howattanakulphong

Name: _____ _____

ID:

Object-Oriented Programming Homework #9 Mar 31st, 2023 Objects and Programs

1. Given that we want to design classes that support defining HTML elements and combining them to form an HTML document, incrementally implements supported classes to accomplish this task.

Initially, the rough interface for defining an HTML document would look like the following definition:

```
class Writer;

class Doc_element { public:
    void write_document(const Writer& w) const;

 void write_to(const Writer& w, int lv) const;

 static Doc_element text(const std::string& t);

    explicit Doc_element(const std::string& n);
 Doc_element(
        const std::string& n, const std::vector<Doc_element>& children);
private:
    /* implementation */
};
```

- At the minimum, the doc element should be able to represent plain **text** element and regular **HTML element** (like <p>, , and <table>) which can also contain child elements.
    - We are not concerned about whether the whole document structure will be a valid HTML. 1

1.1) Draw a class diagram and object diagrams representing the design for supported classes for generating HTML document

```
// Example use cases int
main()
{    auto t = Doc_element::text("Text001");
    auto e = Doc_element("em", {t, Doc_element("p", {t})});
    // `e' will represent the following HTML nodes:
    // <em>Text001 <p>Text001</p> </em>

    auto tr = Doc_element(
        "tr",
        {
            Doc_element("td", {t}), Doc_element("td", {t}),
            Doc_element("td", {t})
        });
    auto tbl = Doc_element("table", {tr, tr, tr});
    /* `tbl' will represent the following HTML nodes:
       <table>
         <tr>  <td>Text001</td> <td>Text001</td> <td>Text001</td>  </tr>
           <tr>  <td>Text001</td> <td>Text001</td> <td>Text001</td>  </tr>
        <tr>  <td>Text001</td> <td>Text001</td> <td>Text001</td>  </tr>
       </table>
    */ }
```

- Draw a class diagram for the class hierarchy for **document node** base class and its derived classes which implements `Doc_element` data structure `Doc_element`
- From the example use cases above, draw an object diagram for `e` and `tbl` objects `e` `tbl`. Create more use
- cases and draw an object diagram representing objects created in the use cases

1.2) Implement all supported classes and write a test program to verify the correctness of the implementation.

```
<!DOCTYPE html>
<html>
<table>
  <tr>
    <td>
      Text001
    </td>
    <td>
      Text001
    </td>
    <td>
      Text001
    </td>
  </tr>
  <tr>
    <td>
      Text001
    </td>
    <td>
      Text001
    </td>
    <td>
      Text001
    </td>
  </tr>
  <tr>
    <td>
      Text001
    </td>
    <td>
      Text001
    </td>
    <td>
      Text001
    </td>
  </tr>
</table></html>
```

```
<!DOCTYPE html>
<html>
<em>
  Text001
  <p>
    Text001
  </p>
</em></html>
<!DOCTYPE html>
<html>
<table>
  <tr>
    <td>
      SE
    </td>
    <td>
      14
    </td>
  </tr>
</table></html>
```