**01: UML Diagram (Study Application)**

**Overview**

Over the next several weeks you will be creating a study application that will let a player review terms and definitions for a course. This first assignment is to create UML (*Unified Modeling Language*) diagrams for the application.

This first version of your application will need to:

* Ask for and store the player's name.
* Display terms and their matching definitions.

Based on what the application will need to do, what classes will you need? What properties and attributes will those classes have? Create UML diagrams to help you visualize how your code will be structured.

**Requirements**

To receive full points for this assignment:

* Create three custom class diagrams
* For each custom class, have at least two attributes and at least one operation (or constructor)
* Create two object diagrams (objects can be instances of the same class, or from two different classes)

As C# is an object-oriented language, all code you create must be inside of a class (either Program or a custom class). Below the default class Program is included, however it is recommended that you create as little code as possible inside of Program for your study application. Instead, create most of your code in your custom classes.

**Learning Goals and Objectives**

While creating this assignment, try to reinforce your understanding of the following:

* What does it mean that UML is a modeling language, and how does that relate to learning C#?
* What are the three areas of the basic UML class diagram?
* The top section: if it is a class diagram, it holds the class name. If it is an object diagram, it holds the object's identifier.
* The middle section: Attributes (essentially, the class or object's data).
* In C# these are also called fields, variables, and properties.
* The bottom section: Operations (essentially the actions or verbs).
* In C# these are also called methods or functions.

This assignment addresses the following course learning objectives:

* Define and use appropriate programming terminology.
* Use object-oriented concepts in the design of an application, including class definitions, methods, constructors, overloaded methods, and object instances.

To complete this assignment you can fill in the template below and upload it. You can also draw your diagrams on paper and upload a photo, or make them in another application (draw.io, visio, etc.).

**01: UML Diagram (Study Application)**

Name: Donald Knechtel 10/05/2020

**Default class Program:**

|  |
| --- |
| **Program** |
|  |
| Main(); |

**Minimum three custom class diagrams:**

|  |
| --- |
| **Student** |
| Name: string |
|  |

|  |
| --- |
| **Application** |
|  |
| start(): Void  askName(): void |

|  |
| --- |
| **Review** |
| ClassTerm: string  BooleanTerm: string  IdentifierTerm: string  FloatTerm: string  MethodTerm: string |
| |  | | --- | | Review()  Review(string classTerm)  Review(string classTerm, string booleanTerm)  Review(string classTerm, string booleanTerm, identifierTerm)  Review(string classTerm, string booleanTerm,string identifierTerm, string floatTerm)  Review(string classTerm, string booleanTerm, string identifierTerm, string floatTerm, string methodTerm.)  DisplayDefinitionInfo(): void | |

**Two object diagrams:**

|  |
| --- |
| **termReview2** |
| ClassTerm: "A value that is either true or false."  BooleanTerm: "..."  IdentifierTerm: "..."  FloatTerm: "..."  MethodTerm: "..." |
| |  | | --- | | DisplayDefinitionInfo(): void | |

|  |
| --- |
| **termReview4** |
| ClassTerm: "A value that is either true or false."  BooleanTerm: "A value that is either true or false."  IdentifierTerm: "A name that a programmer creates for something in their application."  FloatTerm: "..."  MethodTerm: "..." |
| |  | | --- | | DisplayDefinitionInfo(): void | |