

**Assignment #1**

**First Year(Second Semester)**

**(Object Oriented Programming)**

**Spring ,2020(Software Engineering Department)**

**NED University of Engineering and Technology**

**Objectives:**

To get a hands on experience in creating Object Model (Class Diagram) of real world problems.

To gain an idea of practical implementation of some concepts like, definition of classes, Data members, member functions, Constructors, Composition etc**.**

**LO 1**: Understand how to apply the major object-oriented concepts to make classes and programs in C++.

**LO 2**: Understand the concept of object orientation can be used in effective designing of Class Diagrams.

**Instructions:**

**It should be clear that your assignment will not get any credit if:**

* The assignment is submitted after due date.
* The assignment is copied from Internet or from any other student.
* The submitted assignment does not open or file is corrupt.
* It is in some format other than .doc (MS Word Document).
* All types of plagiarism are strictly prohibited.

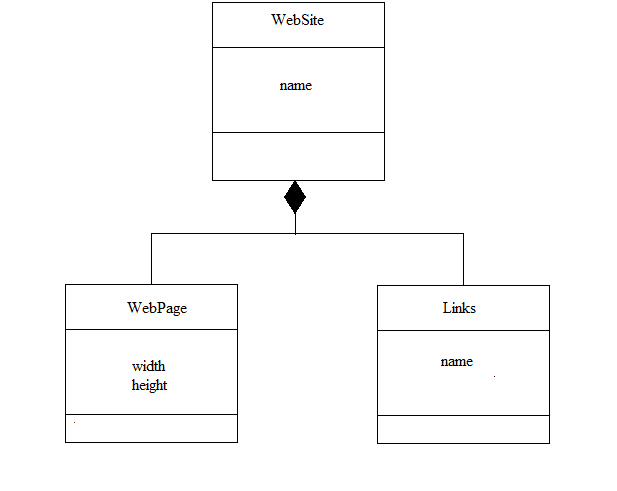
**Lectures Covered:** This assignment covers Lecture # **1-15**

**Assessment Rubrics for Assignment:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Rubric** | **Poor**  **(0-5)** | **Average**  **(5-7)** | **Good**  **(10)** |  |
| Ability to analyze problem and identify requirement, apply algorithm, Identify correct input/output | No submission,  or  Unable to identify any input and output | Submitted with partial attempt | Able to identify  correctly all input  and output and  provide alternative |  |
| Ability to apply required data type or data structure | No submission | Submitted with partial attempt | Submitted with  all the right  answer |  |

Question.1 **Problem Statement:**

We have the following part of class diagram showing composition relationship:



Demonstrate the concept of “Has a” relationship: You are required to implement following class diagram (complete program) in C++ to demonstrate the concept of Composition( Website is composed of webpages and links)with all data members, constructors, member functions .

Add atleast 4 webpages objects in a website class.

And each webpage should have atleast two hyperlinks.

|  |  |  |  |
| --- | --- | --- | --- |
| **Class Name** | **Attribute Name** | **Attribute Data Type** | **Functions** |
| WebSite | name | Character | -Default constructor  -One parameter overloaded constructor () |
| WebPage | width | Double | -Default constructor  --Two parameter overloaded constructor () |
| height | Double |
| Links | name | Character | Default constructor  -One parameter overloaded constructor () |

See the sample output to view the messages you need to print in constructors of all classes. In case of parameterize constructors, you can select any of data member(s) according to the type of constructor.

**Question no.2**

Design a class named Fan to represent a fan. The class contains:

• Three constants named **SLOW, MEDIUM and FAST** with value 1, 2, and 3 to denote the fan speed

. • An **int** data field named **speed** that specifies the speed of the fan (default **SLOW**).

• A **boolean** data field named **on** that specifies whether the fan is o (default false)

• A **double** data field named **radius** that specifies the radius of the fan (default 5).

• A string data field named color that specifies the color of the fan (default blue).

• A default constructor that creates a default fan.

• The accessor and mutator methods for all four data fileds.

• The method named **Showdata()** that returns a string description for the fan. If the fan is on, the method returns the fan speed, color and radius in one combined string. If the fan is not on, the method returns fan color and radius along with the string “fan is off” on one combined string.

**Question no.3**

A class RentedVehicle that has:

• One private instance variable baseFee of type double

• One constructor to initialize the instance variable

• One instance method getCost () that returns the base fee –

Accessor methods for the instance variables

A subclass **FuelVehicle that :**

• has one additional private instance variable Kms indicating the total number of kilometers traveled.

• one constructor to initialize the instance variables.

• one instance method getMileageFees to return the fees due to mileage based on the following:

If Kms < 100 mileagefees=0.2\*kms

If 100<=Kms<= 400 mileagefees=0.3\*kms

If Kms>400 mileagefees=0.3 times 400 plus 0.5 times the extra kilometers above 400. –

accessor methods

**A Car class which is a subclass of FuelVehicle that :**

• has one additional private instance variable Seats

• has one constructor to initialize the instance variables

• overrides getCost method by adding seats\*baseFee to mileageFees

• accessors

**A Truck class which is a subclass of FuelVehicle that:**

• has one private instance variable capacity

• has one constructor to initialize the instance variables

• overrides getCost method by adding baseFee\*capacity to mileageFees

• accessors

**A Bicycle class that extends RentedVehicle that:**

• has one additional private instance variable nDays indicating the number of days it is rented.

• has one constructor to initialize the instance variables

• overrides getCost method to return baseFee \* nDays

accessors

Implement all five classes with their accessor and mutator methods.