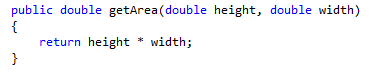
**1050 Programming Logic**Lab 5 (23 points total)

Name: \_\_**Colleen Overbaugh**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. **Identify and describe the following elements of the method header given the following method called getArea(). You must identify where they are in the method header example, other possible values and what the significance of each is (what does it mean?) (12 points – 2 points each)**



1. **Scope- The part of the code where the variable can be accessed. The scope here would be the whole code seen. This is where the variable can be seen and accessed.**
2. **Static vs. Non-Static- Static objects can be accessed from the class that it belongs to and non-static objects must be accessed from the instance in which it belongs. Static objects can only call other static methods or properties of the same class. The area where it would be specified that it is static would be in the beginning such as static void Main().**
3. **Return Type- The return type is what results are expected to be returned. In this case, the results of height \* width would be in the form of a double would be expected.**
4. **Method Name (Identifier)- The method name identifier would be the name in Main that identifies the method of the code. The method name would be getArea because that is the method behind the code, to get results.**
5. **Parameters- Parameters are used to pass values or variables to the method in the code. Double height and double width would be the parameters in the provided example.**
6. **Method Body- The method body is the task that the method is to do. The body will be found in {}. In this case, the return height \* width would be the body of the method.**
7. **Explain the difference between a user-defined method and a method that is provided with a framework. What should we consider when creating a user-defined method? (3 points)**

**The user-defined method is created by the user and is not visible to other methods. Methods that are provided from the framework are predefined. When creating a user-defined method, we should remember to not “reinvent the wheel.” It is always best to reuse classes.**

1. **Discuss the difference between a static and non-static method (2 points)**

**Static methods do not have access to any non-static objects directly, on other static methods in the same class. Non-static methods have access to other non-static methods in an object.**

1. **Use the attached code. Note: you will have to extract the code and open it in Visual Studio before starting. Add a method to the Dog class called bark(). It should have the following characteristics: (3 points)**
   1. Zero parameters
   2. No return value
   3. Should execute Console.WriteLine("{0} is Barking...", name);
2. **Add a method to the Dog class called doTrick(). It should have the following characteristics: (3 points)**

* Should accept a single string parameter called trickName
* No return value
* Should execute:

Console.WriteLine("{0} is so smart! {0} is doing a(n) {1}", name, trickName);