Rawlston Morton

ASP.Net Hw05

1. What is the purpose of the null conditional operator? Does it apply to value types or reference types? Why or why not?

- `?` nullable operator is to set value types to nullable. Reference types are already nullable by default.

2. What is the null coalescing operator? What does it do?

- The null-coalescing operator ?? returns the value of its left-hand operand if it isn't null; otherwise, it evaluates the right-hand operand and returns its result.

3. How is the null conditional operator different from a nullable type? What is the purpose of nullable types?

- If the variable is null then the variable is assigned null. If it contains data, that data is assigned to the variable.

4. When you create an automatic property and initialize the property with a value, \*\*can the value be changed later? Can another value be assigned to the property? Can the value be changed in the constructor?\*\*

- Yes the value can be changed later.

- Yes another value can be assigned to the property.

- You can create multiple constructors with multiple initializations.

5. What is an object initializer and what is it for? What is an collection initializer and what is it for?

- allows me to create an object and specify its property values in a single step.

- Instead of calling the constructor you specify the parameters in curly braces after the type like so.

```cs

Product kayak = new Product {

Name = "Kayak",

Category = "Water Craft",

Price = 275M

};

```

6. What does the is keyword do?

- Performs a type test. Returns true if it is the type, false if it isn't.

- The is keyword performs a type check and, if a value is of the specified type, will assign the value to a new variable

7. How are extension methods defined?

- Extension methods are defined in static classes within the same namespace as the class the extension methods applies to.

8. What is implicit typing or type inference? How do you implement it?

- The `var` keyword allows you to define a local variable without explicitly specifying the variable type

9. What is a default implementation of an interface? What is the purpose of a default implementation of an interface?

- methods in interfaces with concrete implementations.

- makes it possible to update interfaces without breaking the existing implementations of them.

10. What are asynchronous methods? Why should we use them? How are they used?

- Asynchronous methods perform work in the background and notify you when they are complete, allowing your code to take care of other business while the background work is performed.

- Allow your application to take advantage of multi core cpus.

- In ASP.NET Core, asynchronous methods can be used to improve the overall performance of an application by allowing the server more flexibility in the way that requests are scheduled and executed. Two C# keywords—async and await—are used to perform work asynchronously.

11. When you use the await keyword, do you have to use the async keyword? Why or why not?

- When you use the await keyword, you must also add the async keyword to the method signature

- This is because await and async are implemented using some clever compiler tricks, meaning that they allow a more natural syntax, but they do not change what is happening in the methods to which they are applied.

12. \*(Not in the book)\* What is the yield keyword used for? How do you use it?

- you indicate that the method, operator, or get accessor in which it appears is an iterator.

- Using yield to define an iterator removes the need for an explicit extra class (the class that holds the state for an enumeration, see IEnumerator<T> for an example) when you implement the IEnumerable and IEnumerator pattern for a custom collection type. [Microsoft Documentation](https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/keywords/yield)

13. What is the effect of using the `nameof()` expression?

- A nameof expression produces the name of a variable, type, or member as the string constant:

- [Microsoft nameof expression](https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/operators/nameof)