C# Assignment 3 Object-Oriented Programming

1. What are the six combinations of access modifier keywords and what do they do?
   - Public: member can be accessed anywhere
   - Protected: member can be accessed in the current class and child classes
   - Private: member can only be accessed in the current class
   - Internal: member can only be accessed in the current assembly
2. What is the difference between the static, const, and readonly keywords when applied to a type member
   - Const: A variable declared as const must be assigned a value at declaration, and this value may not then change at a later time. If you know the value will never, ever, *ever* change for any reason, use const. A const value is also implicitly static.
   - Static: A static member (variable, method, etc) belongs to the type of an object rather than to an instance of that type. If you need a field to be a property of a type, and not a property of an instance of that type, use static.
   - Readonly: A readonly field is one where assignment to that field can only occur as part of the declaration of the class or in a constructor. If you're unsure of whether or not the value will change, but you don't want other classes or code to be able to change it, use readonly.
3. What does a constructor do?
   - is a special method which shares the same name of the class and doesn't have any return type
   - constructor is used to create an object of the class and initialize class members
   - if there is no constructor in the class, the compiler will provide us a default constructor, and it's parameter-less
   - if we create any constructor ourselves, the default constructor will be replaced
   - constructor can be overloaded
   - constructor cannot be inherited, so a constructor cannot be overriden
   - by default derived class constructor will make a call to a base class constructor
4. Why is the partial keyword useful?

- It provides a special ability to implement the functionality of a single class into multiple files and all these files are combined into a single class file when the application is compiled.

5. What is a tuple?
    - A tuple is a data structure that contains a sequence of elements of different data types. It can be used where you want to have a data structure to hold an object with properties, but you don't want to create a separate type for it.
6. What does the C# record keyword do?
    - Record can be either value type or reference type. It's like Class but equality operators check equality of address pointers for class instances while they check for equality of values for records.
7. What does overloading and overriding mean?
    - Overloading: when we have two or more methods having same name but different signature/parameter lists in the same class.
    - Overriding: happens between base class and derived class, same method signature(access modifier, method name, same input parameters), derived classes can have different implementations for the same methods
8. What is the difference between a field and a property?
    - A field is a variable that is associated with a class or struct, or an instance of a class or struct. A field declared with the static modifier defines a static variable, and a field declared without this modifier defines an instance variable. A static field is associated with a type, whereas an instance variable is associated with an instance.
9. How do you make a method parameter optional?
    - set a default value for the parameter
10. What is an interface and how is it different from abstract class?
    - An abstract class permits you to make functionality that subclasses can implement or override whereas an interface only permits you to state functionality but not to implement it. A class can extend only one abstract class while a class can implement multiple interfaces.
11. What accessibility level are members of an interface?
    - Interface members are public by default because the purpose of an interface is to enable other types to access a class or struct.
12. True. Polymorphism allows derived classes to provide different implementations of the same method.

13. True. The override keyword is used to indicate that a method in a derived class is providing its own implementation of a method.
14. False. The new keyword is used to indicate that a method in a derived class is providing its own implementation of a method.
15. False. Abstract methods can be used in a normal (non-abstract) class.
16. True. Normal (non-abstract) methods can be used in an abstract class.
17. True.Derived classes can override methods that were virtual in the base class.
18. True. Derived classes can override methods that were abstract in the base class.
19. False. In a derived class, you can override a method that was neither virtual non abstract in the base class.
20. False. A class that implements an interface does not have to provide an implementation for all of the members of the interface.
21. True. A class that implements an interface is allowed to have other members that aren't defined in the interface.
22. False. A class can have more than one base class.
23. True. A class can implement more than one interface.