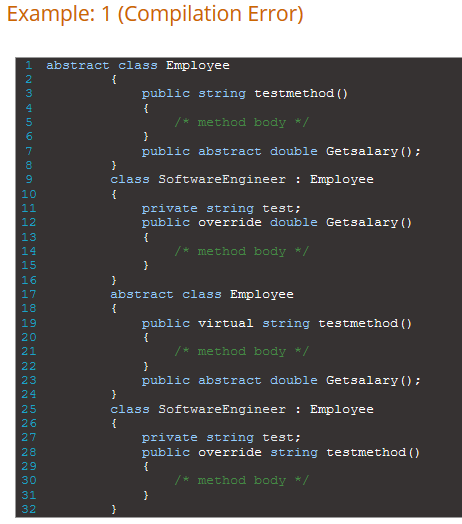
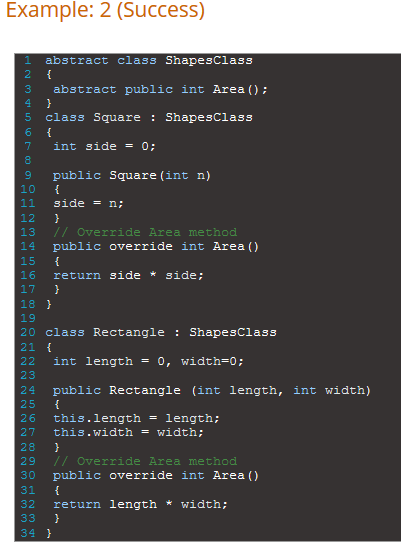
**Abstract class:**

1. Class with the abstract method defined on it is called abstract class
2. Can never be instantiated
3. Used to provide the abstraction
4. It can have some concrete implementation so it don’t provide 100% of abstraction.
5. Methods without the body inside abstract class are called abstract method
6. Method implementation is provided by the subclasses.
7. Method can never be Final or static.
8. Any class inherited must provide the implementation for the abstract method.



The above will lead to compilation error to fix this

1. Implement all the abstract methods in the sub class.
2. If child class doesn’t implement any abstract methods then we can mark child as abstract class too.



**Features:**

1.Can contain both abstract non abstract members.

2. cannot be sealed because doesn’t make sense to have this which cannot be overridden

3. class can have modifier like public, private, protected, internal. But abstract member cannot have private modifier.

4. can have instance variable

5. can have constructor destructor

6. Implicit virtual methods

7. Abstract property behaves like abstract method

8. Purpose is to define some base functionality that can be shared among multiple class and also have their own ways of doing the same thing.

**When to use:**

When two or more subclass are trying to do a similar thing in different way through different implementation. These class inherits same base class and provide different implementation for an abstract method.

Defines generic type of behaviour on top of object oriented class hierarchy that will be used in child class to provide the detailed implementation.

**Interface:**

1. Pure abstract class contains only the abstract methods.
2. Child class has to provide the implementation for all the abstract class.
3. Object of an interface cant be created however, you can create object references.
4. All the methods declared are public and abstract by default.
5. You cant specify the access modifiers on methods inside interface
6. Parent object reference is used to point the child object.
7. In inheritance hierarchy we need to put the parent class and then we have to put the interface parents.

Interface members:

1. Methods
2. Properties
3. Indexer
4. Events

An interface cant have:

1. Constants
2. Fields
3. Operator
4. Constructor
5. Destructor
6. Cannot be instantiated

### Can inherit Interface or Abstract Base Class

### While building class hierarchy often need to decide what should be at the root level of the class hierarchy interface, abstract class or concrete class.

### Interface:

### When need to implement multiple inheritance

### Loosely coupled code

### Deriving too many things might base class might lead to bloated with too much of functionality.

### Independent of classes so can be added to as many class as want.

### When there is a complete separation of concern is achieved interface is a best option (why the class is what it is how and when where it can be used)

### Class:

### Reasonable common data or functionality to be shared with derived class.

### Implementing same interface on too many things lead to lot of repetition of the code

### Can have default implementation which can be overridden in child

### 

### 2

### Abstract class:

### 1.When you want to share the code between several closely related class

### 2. Expect child class to share many common methods and properties and access modifier other than public.

### 3. Non static and non final field with that define methods that can access and changes the status of an object.

### Interface:

### Take advantage of multiple inheritance

### You want to define the behaviour of the object and doesn’t care who is going to use it

### Mostly used for the unrelated object so that there is a loose coupling

### IEnumerable, IComparable