**1.** класс может наследовать (реализовывать) один или несколько интерфейсов. Поэтому в C#, интерфейсы используются для достижения множественного наследования.  
**1B.** класс может наследовать только один абстрактный класс.

**2а.** интерфейс не может предоставить никакого кода, только подпись.  
**2b.** абстрактный класс может предоставить полный код по умолчанию и / или только детали, которые должны быть переопределены.

**3А.** Интерфейс не может иметь модификаторов доступа для подсистем, функций, свойств и т. д. Все считается общедоступным.  
**3B.** абстрактный класс может содержать модификаторы доступа для подсистем, функций, свойств.

**4А.** Интерфейсы используются для определения периферийных способностей класса. Например. A Ship и A Car могут реализовать интерфейс IMovable .  
**4B.** абстрактный класс определяет основную идентичность класса и там он используется для объектов.

**5А.** Если различные реализации только совместно используют сигнатуры методов, то лучше использовать интерфейсы.  
**5B.** Если различные реализации имеют один и тот же вид и используют общее поведение или состояние, то лучше использовать абстрактный класс.

**6A.** если мы добавим новый метод к интерфейсу, то мы должны отследить все реализации интерфейса и определить реализацию для нового метода.  
**6Б.** Если мы добавим новый метод в абстрактный класс, то у нас есть возможность обеспечить реализацию по умолчанию, и поэтому весь существующий код может работать должным образом.

**7А.** Интерфейс не может иметь определенных полей.  
**7B.** абстрактный класс может иметь определенные поля и константы.

**8A.** интерфейс не может иметь конструктора.  
**8B.** абстрактный класс может иметь реализованные конструкторы по умолчанию.

**9А.** Интерфейс может наследовать только от других интерфейсов.  
**9Б.** Абстрактный класс может наследовать от интерфейсов, абстрактного класса или даже класса.