**Quiz zum Thema Records**

**Frage 1**

Welches Schlüsselwort wird in C# verwendet, um einen Record zu definieren?  
a) class  
b) struct  
c) record  
d) data

**Frage 2**

Was unterscheidet einen Record standardmäßig von einer normalen Klasse?  
a) Records werden auf dem Stack gespeichert  
b) Records vergleichen Objekte anhand ihrer Werteigenschaften, nicht Referenzen  
c) Records können keine Methoden enthalten  
d) Records können nicht vererbt werden

**Frage 3**

Welches Ergebnis liefert folgender Code?

public **record** Person(string Name, int Alter);

var p1 = new Person("Anna", 25);

var p2 = new Person("Anna", 25);

Console.WriteLine(p1 == p2);

a) True  
b) False  
c) Compilerfehler  
d) NullReferenceException

**Frage 4**

Welche der folgenden Aussagen zu Records ist **falsch**?  
a) Man kann Records mit with-Ausdrücken kopieren  
b) Records können init-Properties haben  
c) Records sind immer immutable  
d) Records können vererbt werden

**Frage 5**

Welche Ausgabe erzeugt dieser Code?

public record Auto(string Marke, string Modell);

var a1 = new Auto("VW", "Golf");

var a2 = a1 with { Modell = "Polo" };

Console.WriteLine($"{a1.Modell}, {a2.Modell}");

a) Golf, Golf  
b) Golf, Polo  
c) Polo, Polo  
d) Compilerfehler

**Frage 6**

Was ist der Vorteil von with bei Records?  
a) Es verändert die bestehende Instanz  
b) Es erzeugt eine neue Instanz mit kopierten Werten  
c) Es kopiert nur Referenzen  
d) Es löscht Eigenschaften

**Frage 7**

Welcher Konstruktor wird bei diesem Record aufgerufen?

public record Produkt

{

public string Name { get; init; }

public decimal Preis { get; init; }

public Produkt() => (Name, Preis) = ("Unbekannt", 0);

}

Wenn man new Produkt(); aufruft, was passiert?  
a) Name = null, Preis = 0  
b) Name = "Unbekannt", Preis = 0  
c) Compilerfehler  
d) NullReferenceException

**Frage 8**

Wie kann man ein Record als sealed deklarieren, sodass keine Vererbung möglich ist?  
a) public record sealed Kunde(...)  
b) sealed record Kunde(...)  
c) public sealed record Kunde(...)  
d) record Kunde sealed(...)

**Frage 9**

Was ist der Standardzugriffstyp von Records, wenn kein Modifikator angegeben wird?  
a) public  
b) internal  
c) private  
d) protected

**Frage 10**

Welche Ausgabe erzeugt dieser Code?

public record Tier(string Art);

public class Hund(string Art, string Rasse) : Tier(Art);

var h = new Hund("Hund", "Labrador");

Console.WriteLine(h);

a) Hund { Art = Hund, Rasse = Labrador }  
b) Tier { Art = Hund }  
c) Labrador  
d) Compilerfehler

**Lösungen**

**Frage 1**

**Antwort:** c) record  
➡️ Records werden mit dem Schlüsselwort record definiert.

**Frage 2**

**Antwort:** b) Records vergleichen Objekte anhand ihrer Werteigenschaften, nicht Referenzen  
➡️ Während Klassen per Default Referenzgleichheit (ReferenceEquals) nutzen, vergleichen Records die Inhalte (Properties) ihrer Objekte.

**Frage 3**

**Antwort:** a) True  
➡️ p1 und p2 haben denselben Wert in allen Properties → Value-basierte Gleichheit → == liefert True.

**Frage 4**

**Antwort:** c) Records sind immer immutable  
➡️ Records *können* immutable gestaltet werden (z. B. mit init), sind es aber nicht zwingend. Man kann auch set-Properties verwenden.

**Frage 5**

**Antwort:** b) Golf, Polo  
➡️ with erstellt eine **neue Instanz** basierend auf a1.

* a1.Modell bleibt "Golf"
* a2.Modell wird "Polo".

**Frage 6**

**Antwort:** b) Es erzeugt eine neue Instanz mit kopierten Werten  
➡️ with verändert die alte Instanz nicht, sondern erstellt eine Kopie mit ggf. geänderten Werten.

**Frage 7**

**Antwort:** b) Name = "Unbekannt", Preis = 0  
➡️ Der parameterlose Konstruktor wird aufgerufen → Initialisierung über den Body (Name, Preis) = ("Unbekannt", 0);.

**Frage 8**

**Antwort:** c) public sealed record Kunde(...)  
➡️ So wird ein Record als sealed markiert, d. h. er kann nicht weiter vererbt werden.

**Frage 9**

**Antwort:** b) internal  
➡️ Standardmäßig sind Records wie Klassen ohne Modifikator internal.

**Frage 10**

**Antwort:** a) Hund { Art = Hund, Rasse = Labrador }  
➡️ Records haben eine automatisch generierte ToString()-Implementierung, die die Properties und ihre Werte ausgibt.