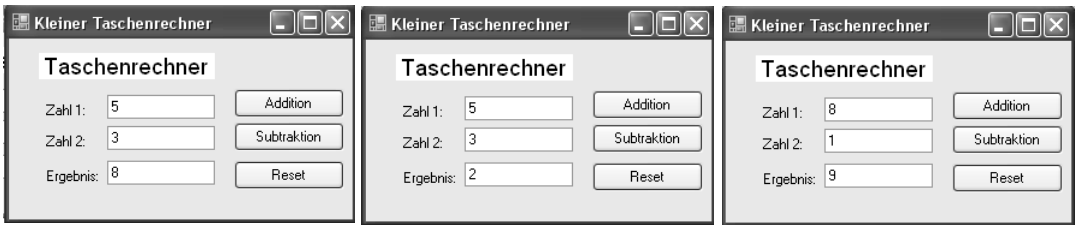
2 Objektorientierte Programmierung

2.1 Klassenbildung

Der Taschenrechner als Klasse

Eine **Klasse** ist eine Schablone, ein Muster oder eine Vorlage, die die Attribute (Eigenschaften) und Methoden (Funktionalitäten) einer bestimmten Art von Objekten definiert.
Ausgehend von Klassen können **Objekte** erzeugt werden. Man spricht von der **Instanziierung** von Objekten. Da jedes Objekt auf Grundlage seiner Klasse **instanziiert** (erzeugt) wurde, übernimmt es automatisch die Eigenschaften und Methoden der Klasse. Die Eigenschaftswerte verschiedener Objekte einer Klasse unterscheiden sich oft.

Aufgabe 1:



Die abgebildeten Taschenrechner sind Realisierungen der Klasse Taschenrechner, man spricht von Objekten der Klasse.
Vervollständigen Sie folgenden Lückentext mit den Begriffen:

Reset, Methoden, Eigenschaften, Werten

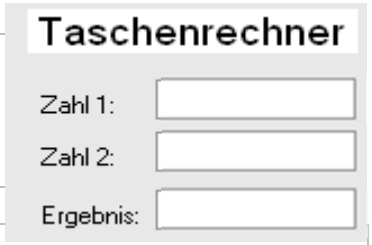
Ein Taschenrechner besitzt Attribute, die zur Berechnung benötigt werden, man spricht von den _____ der Klasse. **Attribute** sind, vereinfacht gesagt, Variablen einer Klasse.

Die oben abgebildeten Taschenrechner-Objekte unterscheiden sich in den _____ der Attribute. _____ des kleinen Taschenrechners sind Addition und Subtraktion.

Des Weiteren verfügt der Taschenrechner über die Methode _____.

Aufgabe 2:

Tragen Sie die Attributwerte des Taschenrechner-Objekts ein, wenn ein Benutzer die Reset-Methode aufruft.



Klassen und abgeleitete Klassen | Vererbung

Man kann eine neue Klasse auf der Basis einer bereits vorhandenen Klasse bilden. Die bereits vorhandene Klasse heißt dann **Basisklasse**, die neue Klasse ist eine **Ableitung der Basisklasse**.

Die abgeleitete Klasse erhält alle Eigenschaften und Methoden, die in der Basisklasse bereits angelegt sind. Diese Übernahme von Elementen und Funktionen heißt **Vererbung**.

Die abgeleitete Klasse ergänzt die Basisklasse um spezielle Eigenschaften und Methoden. Ein Objekt dieser Klasse ist sowohl ein Objekt der abgeleiteten Klasse als auch eines der Basisklasse. Man sagt auch, dass die abgeleitete Klasse eine **Spezialisierung** der Basisklasse darstellt. Umgekehrt ist die Basisklasse eine **Generalisierung** der abgeleiteten Klasse.

Aufgabe 3:

Die zuvor beschriebene Klasse Taschenrechner lässt sich als Basisklasse für eine Klasse **RechnerKlasse4** verwenden.

Nennen Sie zwei neue, sinnvolle Methoden der Klasse RechnerKlasse4.

Nennen Sie alle Eigenschaften der Klasse RechnerKlasse4.

Nennen Sie alle Methoden der Klasse RechnerKlasse4.

Aufgabe 4:

In einem Computerprogramm gibt es eine Basisklasse Fahrzeug. Die Klasse Fahrzeug besitzt die Eigenschaften *Länge, Breite Höhe* und *Geschwindigkeit* sowie die Methoden *Beschleunigen* und *Bremsen*. Von dieser Klasse Fahrzeug wird die Klasse Kipplaster abgeleitet. Die Klasse Kipplaster besitzt zusätzlich die Eigenschaft *Ladevolumen* und die Methode *KippbrückeHeben* und *KippbrückeSenken*.
Welche Eigenschaften und Methoden besitzt die Klasse Kipplaster?

Alle Eigenschaften der Klasse Kipplaster

Alle Methoden der Klasse Kipplaster

Vorteil der Klassenbildung

Wir könnten Taschenrechner in der linearen Struktur, die wir bisher kennen gelernt haben, als einfache Konsolenanwendung programmieren. Aus folgenden Gründen werden wir aber den Taschenrechner in eine eigene Klasse fassen:

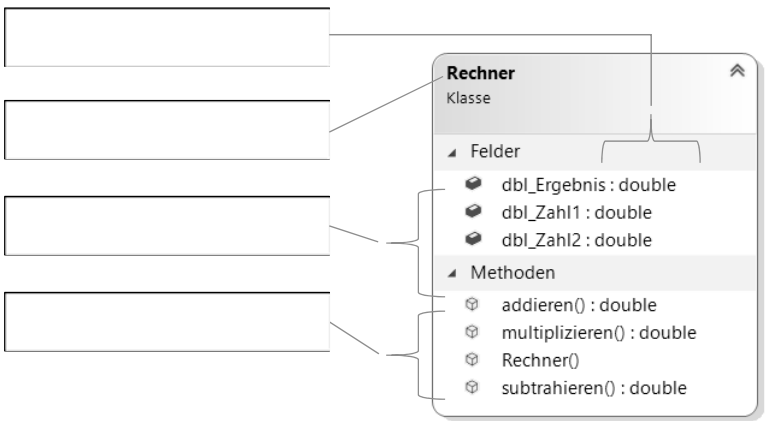
- Der Taschenrechner kann leicht in anderen Programmen wiederverwendet werden.
- Erweitert man die Klasse Taschenrechner um bspw. die Methode Multiplikation, steht diese allen anderen Programmen, die die Klasse verwenden, zur Verfügung.
- Korrigiert man Fehler in einer Basisklasse, so stehen diese Korrekturen den abgeleiteten Klassen zur Verfügung.

Aufgabe 5:

Nennen Sie 2 Vorteile der Klassenbildung:

Aufgabe 6:

Beschriften Sie die abgebildete Klasse mit folgenden Begriffen: ***Funktionalitäten, Attribute, Klassenname, Datentypen***

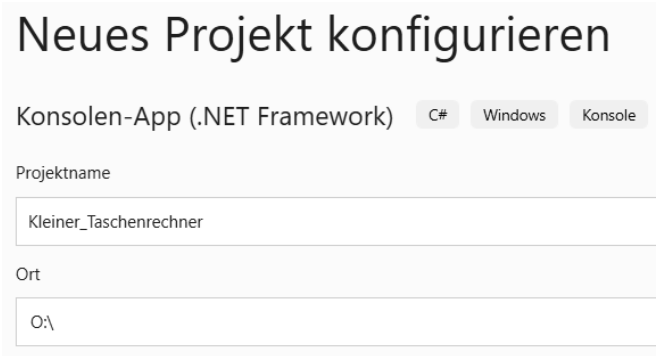


2.2 Deklaration von Attributen und Methoden, Methodenaufruf

Aufgabe 7:
Erstellen Sie das C#-Projekt **Kleiner_Taschenrechner** gemäß folgender Anleitung.

1. Erstellen Sie die .Net-Framework-Konsolenanwendung **Kleiner_Taschenrechner**.
   Menü: Datei → Neu → Projekt: Konsolen-App (.NET Framework)

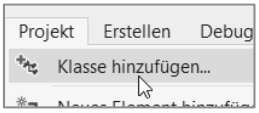
→ **Weiter**



→ Projektname: **Kleiner_Tachenrechner**

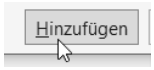
→ Erstellen

2. Fügen Sie eine neue Klasse ein:
   Menü: Projekt → Klasse hinzufügen



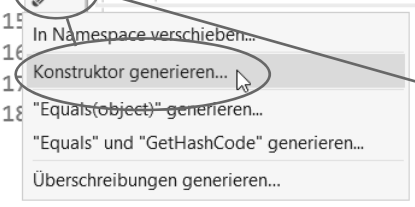
3. Klassen auswählen

4. Name: **Rechner.cs**

5. 

6. Deklarieren Sie die öffentlichen Variablen/Felder der Klasse.

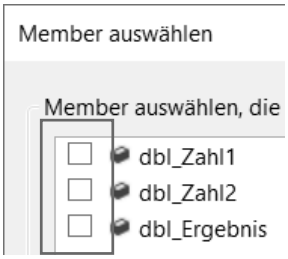
```
1   using System;
2   using System.Collections.Generic;
3   using System.Linq;
4   using System.Text;
5   using System.Threading.Tasks;
6
7   namespace Kleiner_Taschenrechner
8   {
        0 Verweise
9       class Rechner
10      {
11          public double dbl_Zahl1 = 0;
12          public double dbl_Zahl2 = 0;
13          public double dbl_Ergebnis = 0;
14
15
16
17
18
```



Eintippen der Deklarationen
Tipp: Übernehmen Sie die Vorschläge der Autovervollständigung (IntelliSense) mit [Enter].

Konstruktor erzeugen
Zum Erzeugen eines Objekts wird der Konstruktor einer Klasse aufgerufen. Erzeugt wird dieser in einer Leerzeile durch Aufruf einer Schnellaktion mit [Strg] +[.] oder durch Anklicken des Symbols.

7. Heben Sie die vorgeschlagene Auswahl auf.
   → Ok




```
7   namespace Kleiner_Taschenrechner
8   {
        1-Verweis
9       class Rechner
10      {
11          public double dbl_Zahl1 = 0;
12          public double dbl_Zahl2 = 0;
13          public double dbl_Ergebnis = 0;
14
            0 Verweise
15          public Rechner()
16          {
17          }
18      }
19
20  }
```

Konstruktor der Klasse Rechner

Rechner.cs

8. Deklarieren Sie unter den Attributen die Methode **addieren()**.

```
//Methode Addieren
0 Verweise
public double addieren()
{
    dbl_Ergebnis = dbl_Zahl1 + dbl_Zahl2;
    return dbl_Ergebnis;
}
```

Sichtbarkeit **public**: von außen aufrufbar.

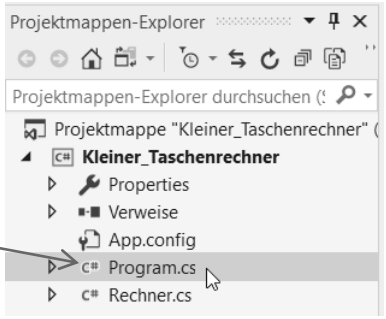
Datentyp des Rückgabewertes der Methode

Name der Methode

Rückgabewert der Methode

9. Deklarieren Sie unter der Methode **addieren()** die Methode **subtrahieren().**

10. Wechseln Sie über den Projektmappen-Explorer zurück ins Hauptprogramm **Program.cs**.



Doppelklick

11. Erfassen Sie in **Programm.cs** den abgebildeten Programmcode.

```
static void Main(string[] args)
{
      Rechner Rechner1;
      Rechner1 = new Rechner();
      Console.Write("Zahl 1: ");
      Rechner1.dbl_Zahl1 = Convert.ToDouble(Console.ReadLine());
      Console.Write("Zahl 2: ");
      Rechner1.dbl_Zahl2 = Convert.ToDouble(Console.ReadLine());
      Console.WriteLine("Plus:  " + Rechner1.addieren());
      Console.WriteLine("Minus: " + Rechner1.subtrahieren());
      Console.ReadKey();
}
```

12. Speichern Sie alle geöffneten Dateien.

13. Starten Sie das Programm.