Aufgaben

1. Gehen Sie die einzelnen Beispiele der Folien durch und probieren Sie das eine oder andere Programm in Visual Studio selbst aus. Setzen Sie Break Points, starten Sie die Programme und werfen Sie während der Laufzeit jeweils einen Blick auf den Typ und den Inhalt der Variablen. Experimentieren Sie etwas mit den Programmen, Visual Studio und dem Debugger. Versuchen Sie sich zu erklären, warum etwas so ist, wie es ist.
   Lesen Sie dazu jeweils auch in den angegebenen Quellen nach (färbiger Kasten jeweils rechts auf jeder Folie), um Ihre beim Testen gemachten Erfahrungen auch erklären zu können. Zudem bieten die Quellen weit mehr Anwendungsbeispiele und Informationen, als wir im Unterricht aus zeitlichen Gründen durchmachen könnten.

2. Erstellen Sie unter Visual Studio für jedes Aufgabenblatt (jede Abgabe) eine eigene Solution, in die Sie jeweils die Programm (Projekte) einfügen. Die Solution sollte die Bezeichnung für
   Teamabgaben: OOP21_<Gruppe>_[AufgabenNr]_[Nachname1]_[Nachname2]
   zB: OOP21_A_03_Mustermann_Musterfrau

   Die Namen der Projekte werden bei den einzelnen Unteraufgaben jeweils angegeben.

3. Programm **Kontoverwaltung** als **2er-Team**-Projekt. Beachten Sie bei der Bearbeitung das dieses Programm in Aufgabe 4 erweitert wird. Versuchen Sie also bestmöglich Erweiterungen zu antizipieren.

   a. Schreiben Sie ein Programm, welches eine **Kontoverwaltung** abbildet. Definieren Sie dazu eine Klasse **Konto**, das sowohl den Kontoinhaber (Name) als auch den Kontostand (Betrag) aufnimmt. Ihr Programm soll in einem Container beliebig viele Konten verwalten können.
      Implementierten Sie Konto Methoden zur **Kontoeröffnung**, zum **Einzahlen**, als auch zum **Abheben** von einem Konto. Sehen Sie dabei vor, dass am Konto kein negativer Saldo entstehen darf. Geben Sie nach einigen Ein- und Auszahlungen die Kontodaten (Namen und aktueller Kontostand) auf `cout` aus. **(3)**

   b. Erweitern Sie Ihr Programm aus a.) nun so, dass die Kontodaten in eine **Textdatei** gespeichert werden können und beim nächsten Programmstart wieder geladen werden können. **(3)**

   c. Erweitern Sie nun Ihre Klassen bzw. das Programm aus b.) so, dass das Konto auch eine **Historie aller Transaktionen** speichert. Bieten Sie auch eine Methode, die einen **Kontoauszug** mit jeweils 10 Transaktionen pro Ausgabe Seite und einer Zwischensumme (aller dieser 10 Transaktionen) auf `cout` ausgibt. **(3)**

d. Testen Sie ihren Code ausreichend in main()!!
Stellen Sie sicher, dass **jede (!) Methode hinreichend getestet** wird, bevor die Software ausgeliefert wird. Schreiben Sie eigenen Testcode dafür. **(1)**

Abgabe

- Klicken Sie unter Visual Studio mit der **rechten Maustaste im Solution Explorer** auf den Solutionname (erste Zeile). Wählen Sie den Menüpunkt **Clean Solution** aus. Damit sollen (fast) alle Compilate und temporären Dateien gelöscht sein.
  Wechseln Sie mit dem Windows Explorer in das Solution-Verzeichnis und löschen Sie dort auch das .vs Verzeichnis (Achtung: hidden folder!) und alle x64 bzw. Debug-Verzeichnisse in den jeweiligen Projektunterverzeichnissen.

  Achtung: Liefern Sie aber etwaige Datendateien der Programme mit, damit man beim Testen auch gleich einen Datenbestand hat.

- Erstellen Sie nun eine ZIP-Datei aus dem gesamten verbleibenden Inhalt des Solution-Verzeichnisses. Der Name sollte

- Die Abgabe erfolgt im Moodle unter Assignment 03
  Beachten Sie, dass es für jede Gruppe einen bestimmten Upload Link gibt.

- bis spätesten **vor Beginn der nächsten Laboreinheit**