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
   Abgaben: OOP21_<Gruppe>_[AufgabenNr]_[Nachname1]_[Nachname2]
   zB: OOP21_A_11_Mustermann.Max

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

3. **sizeCompare (3)**

   Schreiben Sie eine Template Function sizeCompare, welche zwei Parameter a und b mit jeweils beliebigem Typ entgegennimmt. Die Funktion sollte den Speicher-verbrauch der jeweils übergebenen Typen ermitteln und
   +1 zurückgeben, wenn der Speicherverbrauch des a größer ist als der von b
   0, wenn a und b den gleichen Speicherverbrauch haben und
   -1, wenn a weniger Speicherverbrauch hat als b.

4. **TComplex (4)**

   a. Erstellen Sie in einen **Namespace**, benannt nach den **ersten drei** Buchstaben Ihres Nachnamens und dem **erstem** Buchstaben ihres **Vornamens**.

      Bsp: **`Maximilian Tschuchnig -> TscM`**

   b. Erstellen Sie darin eine **Template Class TComplex** zum Rechnen mit komplexen Zahlen.
      Sie können hierfür Ihre Klasse Complex aus der Aufgabe LB08 als Grundlage verwenden.

c. Die Templateklasse sollte zudem die arithmetischen Operatoren +, -, * und / anbieten.

d. Zudem sollten Vergleichsoperatoren und Zuweisungsoperatoren implementiert werden.

e. Erstellen Sie eine spezialisierte Klasse mit dem Template Datentyp double.

5. **FixedQueue (3)**

a. Schreiben Sie eine Template Class **FixedQueue**, die eine Queue (FIFO) für beliebige Datentypen zur Verfügung stellt. Neben dem Datentyp sollte dem Template noch eine maximale Größe vom Datentyp `int` als Parameter übergeben werden können.

Bsp: `FixedQueue<int, 20> mq;`

Wenn die Queue die maximale Größe erreicht hat, sollten die ältesten Queueeinträge überschrieben werden. Ansonsten sollte die FixedQueue in deren Funktionalität soweit wie möglich einer std::queue entsprechen.

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.
- Löschen Sie das .vs Verzeichnis (Achtung: hidden folder!) im Verzeichnis der Solution, bevor Sie das zip-File der Solution erstellen. Liefern Sie auch etwaige Datendateien mit, damit man zum Testen auch gleich einen Datenbestand hat.
- Die Abgabe erfolgt im Moodle unter Assignment 11
  Beachten Sie, dass es für jede Gruppe einen bestimmten Upload Link gibt.
- bis spätesten **vor Beginn der nächsten Laboreinheit**