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_09_Mustermann.Musterfrau

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

3. **NSys – Implementierung eines frei wählbaren Stellenwertsystems**

   Ein Zahlensystem wird zur schriftlichen Darstellung von Zahlen verwendet. Eine Zahl wird dabei nach festgelegten Regeln als Folge von Zahlzeichen, auch Ziffern genannt, dargestellt. Wir verwenden meist im Alltag das 10er System, ein Stellenwertsystem, bei dem wir eine Zahl durch die Ziffern 0, 1, 2, …, 9 (die ersten zehn der natürlichen Zahlen) darstellen. Im Hexadezimalsystem werden bspw. noch die Buchstaben A bis F hinzugenommen, um eine Darstellung der Basis 16 abbilden zu können.
   $36_{10} = 24_{16}$, $93_{10} = 5D_{16}$

   Schreiben Sie eine Klasse **NSys,** mit der Sie die **Darstellung ganzzahliger Werte** in einem beliebigen, frei definierbaren Stellenwertsystem ermöglichen. Damit sollten Sie dann z.B. nicht nur das Dual-, das Oktal, das Dezimal- oder das Hexadezimalsystem abbilden können, sondern jedes beliebige Zahlensystem dazwischen oder darüber hinaus auch (also z.B. ein 13er System, ein 6-er System, ein 32-er System usw.). Als Ziffern können Sie die 0, 1, 2, …, 9, A, B, C, …, X, Y, Z verwenden. Sie haben also 36 Ziffern zur Verfügung.
   Tipp: Um weitere Bearbeitungen in der Klasse zu vereinfachen empfehlen wir, in der Klasse den Wert in einem einheitlichen Format (10ner) abzulegen.

   a.) Für die gesamt Aufgabe gilt:
   Versehen Sie alle Methoden mit einer **Fehlerbehandlung**, bei der Sie im Fehlerfall entsprechende **Exceptions** werfen bzw. etwige Exceptions abfangen und behandeln. Schreiben Sie hierfür eine eigene Klasse **nsys_exception**, die von *std::exception*

abgeleitet ist. Sie können auch weitere Ableitungen davon vorsehen, wenn Sie es für sinnvoll/notwendig halten. **(1)**

b.) Starten Sie nun mit der Klasse **NSys**.
Sehen Sie zur Initialisierung möglichst komfortable **Konstruktoren** vor, mit denen Sie Werte zuweisen und das Zahlensystem festlegen können, z.B. parametrisiert durch
- die Angabe der Zahlenbasis (z.B. 10, 2, 16, 8, usw.) oder
- der frei wählbaren Ziffern (z.B. „01“, „0123456789“, „0123456789ABCDEF“) die verwendet werden können **(1)**

c.) Implementieren Sie **Zuweisungsoperatoren**. Eine Zuweisung sollte auch zahlensystem-übergreifend funktionieren, also z.B. sollte eine NSys Objekt, das eine Oktalzahl darstellt einfach einem NSys Objekt das eine Hexzahl darstellt zugewiesen werden können; oder eine Dezimalzahl einer Hexadezimalzahl, eine Binärzahl einer Hexzahl, usw. Selbstverständlich sollten die ganzzahligen C++ Basistypen auch berücksichtigt werden. **(1)**

d.) Schreiben Sie eine Methode **toString()**, die ein NSys Objekt in einen String konvertiert. **(1)**

e.) Schreiben Sie eine Methode **parse()**, die einen String in eine NSys Objekt konvertiert. **(1)**

f.) Implementieren Sie mindestens 2 **arithmetische Operatoren** der vier Grundrechnungsarten (+, -, *, /, +=, -=, ++, --, …) **(1)**

g.) Implementieren Sie die **Ein-/Ausgabe Stream** << und >> **(1)**

h.) Was, wenn Sie nun den << und >> Operator zusätzlich auch noch für den Bitshift verwenden möchten?
Implementieren Sie auch noch die **Leftshift-** und **Rightshift Operatoren**. **(2 Bonuspunkte)**

i.) Könnten Sie z.B. ein NSys Objekt einfach einem C++ Integer zuweisen und umgekehrt? Könnten Sie ein NSys-Objekt auch einer Funktion *foo* als Parameter übergeben, wenn diese einen Integer als Parameter erwarten würde?

Implementieren Sie etwaige **weitere Operatoren**, wenn Sie es für notwendig/sinnvoll halten (z.B. zur Typkonvertierung, usw.) **(1)**

j.) Nun möchten Sie die gängigen Zahlensysteme möglichst einfach verwenden.
Leiten Sie **eigene Klassen** hierfür von NSys ab (**NSys10, NSys2, NSys16, NSys8**)

Was erben die jeweiligen Spezialisierungen und was müssen Sie hierfür nochmals für jede Klasse implementieren? **(1)**

k.) Schreiben Sie **Testcode (!!)** und testen Sie jede Ihrer Methoden und Operatoren einzeln. Stellen Sie sicher, dass Ihre Klassen so funktionieren, wie Sie sie geplant haben. Testen Sie bewusst auch die Fehlerfälle (**exception handling**!) Ihre Klassen sollten so fehlerfrei und stabil arbeiten, wie nur irgendwie möglich. **(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.
- 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 09
  Beachten Sie, dass es für jede Gruppe einen bestimmten Upload Link gibt.
- bis spätesten **vor Beginn der nächsten Laboreinheit**