Übungen OOSWE/Progr. 2 (C++) – KE 7

Der C/C++-Coding Styleguide ist einzuhalten.
Folgende Einstellungen sind für Debug und Release (All Configurations) vorzunehmen:

Einstellung	**Wert**
Solution Platform	x86
Properties->Conf. Properties->C/C++->General->Warning Level	Level4 (/W4)
Properties->Conf. Properties->C/C++->General->Treat Warnings As Errors	Yes (/WX)
Properties->Conf. Properties->C/C++->General->SDL checks	Yes (/sdl)
Properties->Conf. Properties->C/C++->Code Generation->Basic Runtime Checks	Default
Properties->Conf. Properties->C/C++->Code Generation->Security Checks	Enable Security Checks (/GS)
Properties->Conf. Properties->C/C++->Language->C++ Language Standard	ISO C++ 20 Standard

Legen Sie sich eine Solution an, die alle Aufgaben als Projekte enthält.

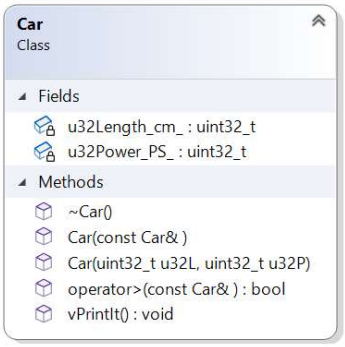
Die geforderten Kommentare sind in Englisch zu hinterlegen.

Aufgabe 1: Operator-Überladung anhand der Klasse Car

Es soll eine Klasse Car im namespace Vehicle realisiert werden. Der Class Designer scheint allerdings = delete noch nicht richtig zu erkennen (Kopierkonstruktor).

```
class Car
   {
   public:
      Car(uint32_t u32L, uint32_t u32P);
      virtual ~Car(void);
      Car(const Car&) = delete;
      //bool operator>(const Car&);
      void vPrintIt(void);

   private:
       uint32_t u32Length_cm_;
       uint32_t u32Power_PS_;
   };
```



In main soll ein Array von fünf Cars statisch instanziiert und initialisiert werden. Es soll nur ein Konstruktor implementiert werden, der mittels Initialisierungsliste die beiden private Attribute u32Length_cm_ und u32Power_PS_ initialisiert. Das Array soll nach der **abgeleiteten** Größe u32Size aufsteigend mit BubbleSort sortiert werden. u32Size soll kein Attribut sein, das es den bestehenden Attributen berechnet werden kann (abgeleitetes Attribut).

u32Size = u32Length_cm_ + u32Power_PS_.

Realisieren Sie den Bubblesort-Algorithmus direkt in main.

Die Klasse Car soll mittels einer Printfunktion vPrintIt u32Size, u32Length_cm_ und u32Power_PS_ ausgeben.

Warum gibt es in Bubblesort einen Fehler? Schreiben Sie die Antwort über main als Kommentar.

Beheben Sie den Fehler, indem Sie einen Operator überladen.
Geben Sie vor und nach dem Sortieren das Array mittels einer Schleife auf dem Bildschirm aus.

Im vorliegenden Beispiel (Bubblesort) ist ein weiterer wichtiger Operator schon per default vorhanden. Um welchen handelt es sich hierbei? Was ist die „Gefahr“ bei der Verwendung dieses Default-Operators? Schreiben Sie die Antwort als Kommentar über main.

Übungen OOSWE/Progr. 2 (C++) – KE 7

Aufgabe 2: Operator-Überladung der Klasse Complex

2.1 Legen Sie sich ein neues Projekt an und kopieren Sie sich die Klasse Complex aus KE02_AG1. Fügen Sie der Klasse noch hinzu:

- Operatorfunktion für Addition (Objekt wird nicht überschrieben)
- Operatorfunktion für Subtraktion (Objekt wird nicht überschrieben)
- Operatorfunktion für Multiplikation (Objekt wird nicht überschrieben)
- Operatorfunktion für Division (Objekt wird nicht überschrieben)
- Operatorfunktion für Invers (1/z) (**Objekt wird überschrieben – Verkettung sei möglich**)

Für $z_1 = a_1 + b_1\mathrm{i}$ und $z_2 = a_2 + b_2\mathrm{i}$ gilt:

$$z_1 \pm z_2 = (a_1 \pm a_2) + (b_1 \pm b_2)\mathrm{i}$$

$$z_1 \cdot z_2 = (a_1 a_2 - b_1 b_2) + (a_1 b_2 + a_2 b_1)\mathrm{i}$$

$$\frac{z_1}{z_2} = \frac{z_1 \cdot \bar{z}_2}{z_2 \cdot \bar{z}_2} = \frac{a_1 a_2 + b_1 b_2 + (a_2 b_1 - a_1 b_2)\mathrm{i}}{a_2^2 + b_2^2} \quad (z_2 \neq 0 + 0 \cdot \mathrm{i})$$

Quelle: www.formelsammlung.de

Bei den Operatorfunktionen für die Division als auch für Invers kann es zu einer Division durch Null führen. Werfen Sie in diesem Fall die std::runtime_error Exception (Text: „Division by a complex number which is zero“).

2.2 Bisher wurden die Objektdaten in eher rudimentären Funktionen wie vPrintIt oder vPrintComplexNumber realisiert. Nachteil ist, dass diese nicht mit den normalen std::out und << verkettbar sind. Löschen Sie vPrintComplexNumber!

Ziel soll sein, dass eine solche Anweisung möglich ist:

```
std::cout << ComplexNumber1 << std::endl; //ComplexNumber1 is an object
```

Dies ist soll jetzt durch eine Operatorfunktion für den binären Operator << behoben werden. Sie sollten sich folgende Fragen beantworten können:

1. Welcher Datentyp verbirgt sich hinter std:cout?
2. Wie lautet die Assoziativität des <<-Operators?
3. Welcher Datentyp muss der linke Operand haben?
4. Welcher Datentyp muss der rechte Operand haben?
5. Welcher Operand bestimmt, ob der Operator als friend-Funktion oder Methode einer Klasse implementiert wird?

Implementieren Sie diesen Operator als friend-Funktion. Es sollen dabei nur Referenzen übergeben werden. Damit eine Verkettung mit ostream möglich ist, muss auch eine Referenz auf ostream zurückgegeben werden.

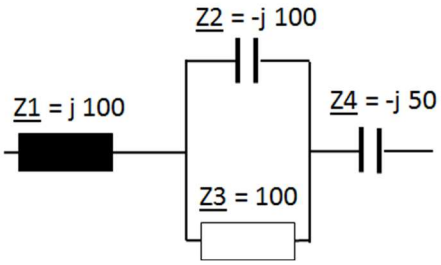
2.3 Testen Sie Sie Ihre Implementierung, indem Sie den komplexen Gesamtwiderstand der folgenden Schaltungen berechnen:

Reihenschaltung: Widerstände Z werden addiert!
Parallelschaltung: Leitwerte Y werden addiert!

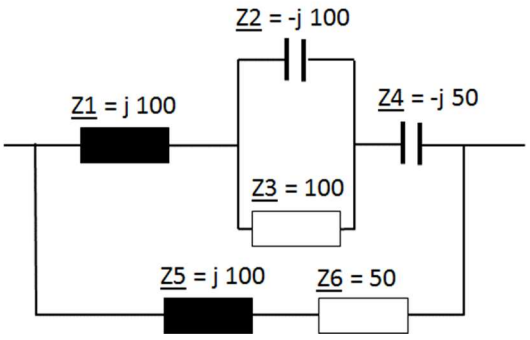
Umrechnung: $\underline{Z} = \frac{1}{\underline{Y}}$

Berechnen Sie mittels der Klasse Complex den Gesamtwiderstand der folgenden Schaltungen und geben Sie diesen auf der Console aus.

Schaltung 1: Lösung: 50 + j0



Schaltung 2: Lösung: 37.5 + j12.5



Schaltung 3: Lösung: 129.902 – j22.0588