Hochschule Deggendorf Dr. Peter Jüttner	
Vorlesung: Objektorientierte Programmierung	SS 2019
Übung 9	Termin 23.5.19

C++-Ein-/Ausgabe und überladene Operatoren

1. Bildschim / Konsole

Schreiben Sie ein Programm, dass nacheinander einen Text, eine Ganzzahl und eine Gleitkommazahl mittels der C++-Ein-/Ausgabeoperatoren <<, >> von Tastatur einliest und auf Bildschirm wieder ausgibt.
Die Ausgabe der Zahlen soll dabei bei der Ganzzahl in den Formaten Hex und Dezimal erfolgen, die Gleitkommazahl soll mit 2 und 10 Nachkommastellen ausgegeben werden.

```
#include <iostream>
#include <iomanip>
#include <fstream>

using namespace std;

int main(void)
{
// Aufgabe 1. Teil

  char s[10];
  int i;
  double d;
  cout << "Bitte Text eingeben\n";
  cin >> s;
  cout << "Bitte eine Ganze Zahl eingeben\n";
  cin >> i;
  cout << "Bitte eine Gleitkommazahl eingeben\n";
  cin >> d;
  cout << "Zeichenkette. " << s << endl;
  cout << "ganze Zahl in Hex: " << hex << i << endl;
  cout << "ganze Zahl in Dez: " << dec << i << endl;
  cout << "Gleitkommazahl mit 2 Nachkommastellen:" << setprecision(3) << d <<
endl;
  cout << "Gleitkommazahl mit 2 Nachkommastellen:" << setprecision(11) << d
<< endl;
```

```
  system("PAUSE");
  return(0);
}
```

2. Überladene Operatoren

Ergänzen Sie das Programm der Klasse complex aus der 2. Übung um folgende überladene Operatoren (ein entsprechendes Programm finden Sie auf dem Laufwerk, falls Sie Ihre Lösung nicht mehr zugreifbar haben)

- << für die Ausgabe auf Bildschirm
- + für die Addition zweier complex Zahlen
- * für die Multiplikation zweier complex Zahlen
- - für die Differenz zweier complex Zahlen

Probieren Sie die überladenen Operatoren aus, auch in zusammengesetzten Ausdrücken.

```
#include <iostream>
#include <stdio.h>

using namespace std;

class complex
{ float real;
  float imag;

  public:

  complex()
  { real = 0.0;
    imag = 0.0;
  };

  complex(float r, float i)
  { real = r;
    imag = i;
  };

  ~complex()
  {
  };

  // Addition zweier Complex
  complex operator+(complex c)
  { complex r;
    r.real = real + c.real;
    r.imag = imag + c.imag;
    return r;
  };
```

```
    // Multiplikation zweier Complex
  complex operator*(complex c)
  { complex r;
    r.real = real*c.real - imag*c.imag;
    r.imag = imag*c.real + real*c.imag;
    return r;
  };

  // Subtraktion zweier Complex
  complex operator-(complex c)
  { complex r;
    r.real = real - c.real;
    r.imag = imag - c.imag;
    return r;
  };

  friend ostream& operator<<(ostream&, complex);
};

// Ausgabe auf Bildschirm

ostream& operator<<(ostream& o, complex c)
{ o << c.real << "+" << c.imag << "i";
  return o;
};

main()
{

  complex c1(1.0, 2.0);
  complex c2(3.0, 4.0);
  complex c3(1.0, 2.0);

  cout << (c1+c2) * c3 << endl;

  system("PAUSE");

}
```

3. **Addieren von Complex und Gleitkommazahlen**
   Probieren Sie auch eine Gleitkommazahl zu einer Complex-Zahl mittels + zu addieren.
   Überlegen Sie, wie Sie dieses Problem **ohne** einen weiteren überladenen +-Operator **aber** mit Hilfe eines Konstruktors lösen können.

   Tipp: Der C++-Compiler versucht immer einen passenden Konstruktor auf einer Parameterposition anzuwenden, falls eine Methode (oder Operator) mit dem passenden Typ nicht definiert ist.

   Die Klasse complex kann um einen Konstruktor ergänzt werden, der aus einem

float Wert ein Objekt der Klasse complex erzeugt (dem Realteil wird der float Wert zugewiesen, dem Imaginärteil 0). Wird jetzt der überladene +-Operator,der eigentlich für zwei Parameter vom Typ complex definiert ist, mit einem Parameter vom Typ float aufgerufen, so ruft der Compiler implizit den Konstruktor auf, der aus einem float Wert ein complex Objekt erzeugt. Anschließend wird der überladene +-Operator ausgeführt.

…

```
complex(float r)
{ real = r;
  imag = 0;
};

…

main()
{
  float f = 1.5;

  complex c1(1.0, 2.0);

  cout << c1+f << endl;

  system("PAUSE");

}
```