Technische Hochschule Georg Agricola

Allgemeine Angaben

Modulprüfung	Klausur	Datum: 17.03.2022
Modulname: **Objektorientierte Programmierung**	Modulnummer:	40050200
Prüfungsdauer: **60 min + 15 min**	Prüfer: Welp	

Name: _____ Vorname:_____

Matrikelnummer: _____

Bewertung

Aufgabe	1	2	3	4						Summe
Erreichbare Punkte	9	12	8	18						47
Erzielte Punkte										
Unterschrift Prüfer				Erzielte Punkte in %					Note	

Aufgabe 1: Von C nach C++

1. Schreiben Sie eine Funktion `gesamtwiderstand` zur Berechnung des Gesamtwiderstands von zwei parallel oder seriell verschalteten elektrischen Widerständen. Der Funktion sollen als Parameter die beiden Widerstandswerte `r1` und `r2` übergeben werden sowie ein Kenner `k` vom Typ `int`, der angibt, ob der Gesamtwiderstand für eine Parallelschaltung (`k=0`) oder für eine Reihenschaltung (`k=1`) berechnet werden soll. Die Angabe des Kenners sei optional. Wird er weggelassen, soll eine Reihenschaltung berechnet werden.

   Lösen Sie dieses Problem, indem Sie

   1. `gesamtwiderstand` überladen
   2. `gesamtwiderstand` als Funktion mit Defaultparametern definieren.

   Geben Sie für beide Lösungswege die Funktions**definitionen** an. (6 Punkte)

2. Was gibt folgendes Programmfragment auf dem Bildschirm aus? (3 Punkte)

```
long n[3] = {300,200,100};
long& r = n[0];
long* p = &n[2];
n[1] = r/2;
p=p-1;
r = *p + n[0];
cout << n[0] << "  " << n[1] << "  " << r << endl;
```

Aufgabe 2: Klassen und Objekte

Definieren Sie eine Klasse `Kreis` für die Darstellung von Kreisen. Die Klasse soll über folgenden Eigenschaften verfügen *(Anm.: Lassen Sie in der Klasse genügend Platz um nach und nach Ergänzungen vornehmen zu können)*:

- Der Radius des Kreises soll als Attribut in der Klasse gespeichert werden. Der direkte Zugriff auf den Radius eines Kreises soll nicht möglich sein. Der Zugriff soll über entsprechende setter- und getter-Methoden erfolgen. **Definieren (implementieren)** Sie die Methoden als inline-Funktionen. Negative Radien sollen nicht mögilch sein.
- Die Klasse soll über geeignete Kontruktoren zur Initialisierung von `Kreis`-Objekten verfügen. Ein `Kreis`-Objekt soll dabei über die Angabe des Radius initialisiert werden können. Eine Objektinstanziierung ohne Parameter soll auch möglich sein. In diesem Fall soll das Objekt mit einem Radius von 1.0 initialisiert werden (*siehe Hinweis*). **Definieren** Sie die Methoden als inline-Funktionen.
- Die Klasse soll eine Methode `getUmfang()` zur Verfügung stellen, die den Umfang des Kreises zurückliefert. **Definieren** Sie die Funktion ausserhalb der Klasse.
- Die Klasse soll über ein Klassenattribut verfügen, in welchem die Anzahl der `Kreis`-Objekte gespeichert wird (Objektzähler). Die Anzahl der `Kreis`-Objekte soll über eine Klassenmethode abfragbar sein. **Definieren** Sie das Attribut und die Methode.
- Ferner soll der zweistellige `/`-Operator für die Klasse `Kreis` überladen werden. Dieser soll die "Division" eines Kreises mit einem Wert `x` erlauben. Ein entsprechender Ausdruck soll einen Kreis mit dem `1/x`-fachen Umfang liefern. **Deklarieren** Sie den Operator. *(siehe Hinweis)* (12 Punkte)

Hinweis

Die Klasse `Kreis` soll z.B. folgendermaßen verwendet werden können

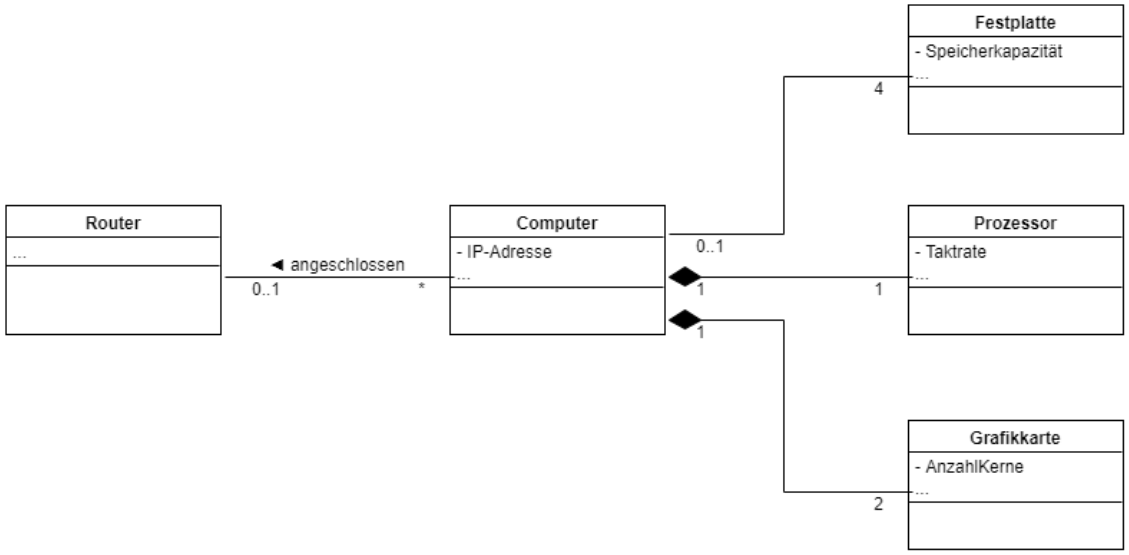
```
int main()
{
    Kreis k1;
    Kreis k2(3.0);
    Kreis k3;

    cout << "Kreis mit Radius " << k1.getRadius() << " hat Umfang " << k1.getUmfang() << endl;
    cout << "Kreis mit Radius " << k2.getRadius() << " hat Umfang " << k2.getUmfang() << endl;

    k3=k2/3;
    cout << "k2/3 hat Radius " << k3.getRadius() << " und Umfang " << k3.getUmfang() << endl;

    cout << "Es gibt insgesamt " << Kreis::getZaehler() <<" Kreise" << endl;
}
```

Ausgabe:

```
Kreis mit Radius 1 hat Umfang 6.283
Kreis mit Radius 3 hat Umfang 18.849
k/3 hat Radius 1 und Umfang 6.283
Es gibt insgesamt 3 Kreise
```

Aufgabe 3: Objektbeziehungen und Templates

In einer Netzwerkmanagement-Software gibt es die Klassen `Router`, `Computer`, `Festplatte`, `Prozessor` und `Grafikkarte`, die entsprechend dem folgendem Klassendiagramm miteinander in Beziehung stehen.



1. Welche Attribute werden in der Klasse `Computer` benötigt um die Beziehungen zu den Klassen `Router`, `Festplatte`, `Prozessor` und `Grafikkarte` zu implementieren? Beantworten Sie die Frage, in dem Sie die Klassendefinition für die Klasse Computer angeben (nur Attribute, Methoden sind nicht erforderlich). (4 Punkte)

2. **Definieren** Sie eine **Template**-Funktion `min`, die den kleineren von zwei Werten zurückliefert. Die beiden Werte sollen von beliebigem Typ sein. (4 Pkt.)

   mögliche Anwendung:

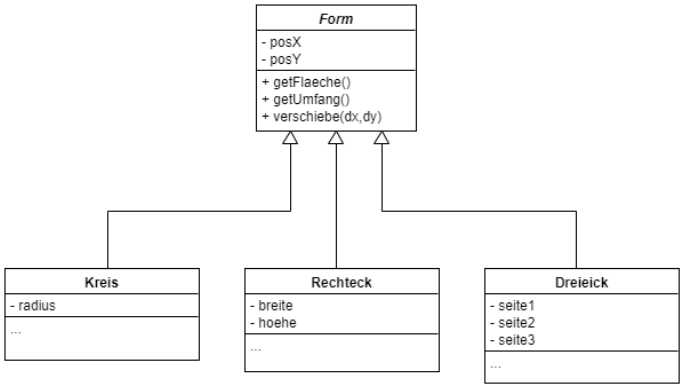
   ```
   cout << min(7,2) << endl;
   cout << min("Bochum","Dortmund") << endl;
   ```

   Ausgabe:

   ```
   2
   Bochum
   ```

Aufgabe 4: Vererbung und Polymorphismus

Gegeben sei folgendes Klassendiagramm für geometrische Formen:



Die Klasse `Form` ist folgendermaßen definiert:

```
class Form
{
    public:
        Form(int x, int y);
        void verschiebe(int dx, int dy);
        virtual double getFlaeche()=0;
        virtual double getUmfang()=0;
    private:
        int posX,posY;
};
```

1. Die Klasse `Form` ist eine **abstrakte** Klasse. (3 Punkte)
   a. Woduch wird sie zur abstrakten Klasse?
   b. Was bedeutet das für die Verwendbarkeit der Klasse?
2. Leiten Sie von der Klasse `Form` eine konkrete Klasse `Rechteck` ab. Geben Sie die Klassendefinition inklusive der Methodendefinitionen an. (5 Punkte)
3. Über welche Attribute verfügen Objekte der Klasse `Rechteck`? (2 Punkte)
4. Schreiben Sie eine Funktion `gesamtFlaeche` zur Berechnung der Gesamtfläche eines Feldes von geometrischen Formen (Kreise, Rechtecke, Dreiecke). (6 Punkte)
5. Welchen Aussagen zum Observer-Entwurfsmuster stimmen Sie zu? (2 Punkte)

   ☐ Wenn sich die Schnittstelle eines Subjektes ändert, hat das auch Auswirkungen auf den Beobachter.

   ☐ Bevor ein Beobachter über Zustandsänderungen eines Subjekts informiert wird, muss es sich bei dem Subjekt registrieren.

   ☐ Subjekte aktualisieren ihren Zustand sobald sich Beobachter bei den Subjekten registrieren.

   ☐ Ein Beobachter ruft sporadisch die Methode `notify()` des Subjekts auf, um über Zustandsänderungen des Subjekts informiert zu sein.

   ☐ Der Beobachter muss eine definierte Schnittstelle implementieren, die in einer abstrakten Basisklasse definiert ist, von der der Beobacher abgeleitet wird.