❮ Info2 (Ahaus) – Sommersemester 17

Aufgabenblatt 6: Klassen und Vererbung

Kapitel 1: Praktikumsaufgaben

6.1.1: Klasse Rechteck

Für ein Geometrieprogramm, das intern mit kartesischen Koordinaten rechnet, wollen Sie eine Klasse rechteck erstellen, die alle für Ihr Programm benötigen Funktionen eines Rechtecks bereitstellt. Im Vorfeld der Programmierung haben Sie die folgenden wichtigen Anforderungen identifiziert:

1. Ein Rechteck soll durch Angabe von zwei Ecken oder durch Angabe von linker oberer Ecke sowie Höhe und Breite angelegt werden können.
2. Ein Rechteck soll um einen bestimmten Wert vergrößert (um den Mittelpunkt „aufgeblasen“) oder verkleinert (auf den Mittelpunkt hin „geschrumpft“) werden können.
3. Es soll geprüft werden können, ob ein bestimmter Punkt im Rechteck liegt oder nicht.
4. Es soll das größte in zwei Rechtecken liegende Rechteck berechnet werden können.
5. Es soll das kleinste zwei Rechtecke umfassende Rechteck berechnet werden können.
6. Es soll die Fläche des Rechtecks berechnet werden.
7. Es soll der Mittelpunkt des Rechtecks berechnet werden.
8. Das Rechteck soll um einen bestimmten Versatz verschoben werden können.

Erstellen Sie eine vollständige C++-Klasse `rechteck`, die diesen Anforderungen genügt. Entscheiden Sie selbst, welche der Anforderungen besser durch Funktionen und welche besser durch Operatoren zu realisieren sind. Erstellen Sie eine Hilfsklasse `punkt`, die sowohl zur Speicherung von Daten im Rechteck als auch als Parameter für Funktionen oder Operatoren verwendet wird.

6.1.2: Konstruktoren ergänzen

Für ein C++-Programm haben Sie die folgenden Klassen erstellt:

```
class aaa
    {
    private:
        int ai;
        float af;
    };

class bbb
    {
    private:
        char bc;
        char bpc[100];
    };
```

```
class ccc : public aaa
    {
    private:
         bbb mb;
         double cd;
    };
```

Erstellen Sie die noch fehlenden Konstruktoren für die Klassen `aaa`, `bbb` und `ccc`, damit Sie die Klasse ccc anschließend wie folgt instantiieren können,

```
void main()
    {
    int ai = 0;
    float af = 2.3f;
    char bc = 'x';
    char *bpc = "yyy";
    double cd = 4.567;

    ccc instanz_ccc(ai, af, bc, bpc, cd);
    }
```

wobei die bei der Instantiierung übergebenen Parameter (`ai`, `af`, `bc`, `pbc`, `cd`) zur Initialisierung der gleich benannten Membervariablen der beteiligten Klassen verwendet werden sollen.

6.1.3: Person als Basisklasse von Student und Dozent

Erstellen Sie die Klassen `student` und `dozent` als Kindklassen der Klasse `person` aus der Aufgabe Klasse Person. Die Klasse Student soll als zusätzliche Information gegenüber der Person den Studiengang des Studenten enthalten, die Klasse Dozent das gelehrte Fach.

Erstellen Sie für die neuen Klassen jeweils geeignete Konstrukturen sowie die notwendigen Getter- und Setter-Methoden.

Sehen Sie zusätzlich für alle Klassen eine Methode `tagewerk` vor, die den Namen der Person und ihre Tätigkeit (der Student studiert in seinem Studiengang, der Dozent lehrt sein Fach) ausgibt. Sehen Sie auch eine geeignete Ausgabe für die Basisklasse vor.

Passen Sie die Zugriffsspezifikationen der Basisklasse ggf. an.

6.1.4: Anpassung der Stack-Klasse

Ändern Sie die Klasse stack aus der Aufgabe Klasse Stack so, dass auf dem Stack keine Instanzen der Klasse Person mehr abgelegt werden. Stattdessen soll der Stack Zeiger auf Objekte der Klasse Person `person*` verwalten. Mit den Methoden `push` und `pop` soll ein entsprechender Zeiger auf dem Stack abgelegt oder vom Stack entnommen werden.

Erstellen Sie verschiedene Instanzen der Klassen Student und Dozent aus der Aufgabe Person als Basisklasse von Student und Dozent und legen Sie Zeiger auf die Instanzen auf dem Stack ab.

Entnehmen Sie die Abgelegten Adressen danach dem Stack und rufen Sie die Methode `tagewerk` für die entsprechenden Instanzen auf. Passen Sie die Klassen ggf. an, so dass Sie das gewünschte Ergebnis erhalten.