

Übungen zu Klassen

1 Klasse Rechteck

Gegeben ist das untenstehende Programm, bestehend aus einem vordefinierten Hauptprogramm und einer vordefinierten Klasse Punkt. Das Programm soll, wie aus der main-Routine ersichtlich ist, den Umfang und die Fläche eines Rechtecks unter Anwendung der Klasse Rechteck berechnen. Schreiben Sie die hierzu erforderliche Klasse Rechteck unter folgenden Annahmen:

Die Klasse Rechteck enthält zwei private Elemente P1 und P2 vom Typ Punkt (= Eckpunkte des Rechtecks). Der Konstruktor der Klasse hat als Übergabe zwei Parameter vom Typ Punkt.

Die public Methode Flaeche der Klasse Rechteck besitzt keine Übergabeparameter und gibt einen double-Wert, die Fläche des Rechteckes, zurück.

Die public Methode Umfang der Klasse Rechteck gibt kein Resultat zurück, besitzt aber als Übergabeparameter eine Referenz auf einen double, durch den der berechnete Umfang an die aufrufende Instanz übertragen wird.



```
#include <iostream>
#include <cmath>

class Punkt {
private:
    double x, y;
public:
    Punkt(double X=0, double Y=0) { x=X; y=Y; } //Konstruktor
    void set (double X , double Y) { x=X; y=Y; } //Methode
    double getX () { return x; } // "-"
    double getY () { return y; } // "-"
};

class Rechteck {
    Punkt P1(x,y);
    Punkt P2(x,y);

public:
    Rechteck(Punkt P1, Punkt P2) {
        this->P1 = P1;
        this->P2 = P2;
    }

    double Flaeche() const {
        return (P2.getY() - P1.getY()) * (P2.getX() - P1.getX());
    }

    void Umfang(double &U) const {
        U = 2 * ((P2.getY() - P1.getY()) + (P2.getX() - P1.getX()));
    }
};

void main() {
    // Zwei Variablen vom Typ "Punkt" zur Eingabe der Rechteckkoordinanten
    Punkt A, B;

    // Eingabe der Koordinaten der Punkte A und B
    double x,y;
    cout << "Koordinaten x und y Punkt A : "; cin >> x >> y; A.set(x,y);
    cout << "Koordinaten x und y Punkt B : "; cin >> x >> y; B.set(x,y);

    // Erzeugen eines Objekts der Klasse "Rechteck" und
    // Berechnung des Umfangs und der Flaeche des Rechteckes
    Rechteck RE( A, B);
    cout << "Die Rechteckflaeche betraegt " << RE.Flache() << endl;

    double Umfang;
    RE.Umfang(Umfang);
    cout << "Der Umfang des Rechteckes betraegt " << Umfang << endl;
}
```

2 Klasse Semester

Eine Klasse `Semester` soll angeben, welche Studenten die Prüfungen bestanden haben, und den Notendurchschnitt aller Studenten berechnen.

Die Klasse `Semester` mit folgenden Eigenschaften ist zu schreiben:

- Die Studenten sind in einer privaten Variable `studenten` vom Typ `vector<Student>` zu speichern. Der Typ `Student` ist unten angegeben.
- Eine öffentliche Methode `studentHinzufuegen(Student student)` fügt einen Studenten als neues Element zum privaten Vektor `studenten` hinzu.
- Die öffentliche Methode `bestanden()` gibt die Vornamen und Namen aller Studenten auf den Bildschirm aus, die einen Notendurchschnitt grösser oder gleich 4 haben.
- Die öffentliche Methode `gesamtDurchschnitt()`, gibt den Gesamtnotendurchschnitt aller Studenten als `double` zurück.



```
const int AnzPruefungen = 10;           //Anzahl der Prüfungen

struct Student{                         //Klasse für einen Studenten
    string name,vorname;                 //Name und Vorname
    double noten[AnzPruefungen];         //In den Prüfungen erhaltene Noten
    double durchschnitt(){               //Berechnet den Notendurchschnitt
        double durchschnitt=0.;
        for(int i=0;i< AnzPruefungen;i++) durchschnitt+=noten[i];
        return durchschnitt/AnzPruefungen;
    }
};

class Semester{
```

```
};
```

3 SIMcards für Mobiltelefone

Schreiben Sie die Klasse `SIMcard` mit den nachfolgend beschriebenen Eigenschaften. Verwenden Sie dabei den unten gegebenen Typ `Eintrag`, in dem jeweils ein Name mit einer Telefonnummer gespeichert wird.

- Die Klasse hat als private Elemente ein Telefonbuch vom Typ `vector<Eintrag>` und die PIN (Sperrcode) als `int`-Zahl.
- Der Konstruktor besitzt einen Referenzparameter vom Datentyp `int`, in dem die PIN der `SIMcard` gespeichert wird.
- Die öffentliche Methode `trageEin` vom Typ `bool` trägt in das Telefonbuch einen neuen Eintrag ein. Übergabeparameter sind ein Name, eine Telefonnummer sowie die PIN. Bei Angabe einer falschen PIN wird kein Eintrag vorgenommen und die Methode gibt den Wert `false` zurück, ansonsten `true`.
- Die öffentliche Methode `sucheNummer` vom Typ `int` sucht die zu einem Namen gehörige Telefonnummer im Telefonbuch. Die Suche wird nur durchgeführt, wenn die übergebene PIN derjenigen der `SIMcard` entspricht. Die Methode gibt den Wert 0 zurück, wenn kein Eintrag vorhanden ist und -1, wenn die übergebene PIN falsch ist.



```
struct Eintrag{
    string  name;
    int     nummer;
}
```

4 Verpackungsanlage

In einem Versandbetrieb soll zum Verpacken einer Ware vom Typ Quader geprüft werden, ob diese in einen Karton passt, ebenfalls vom Typ Quader. Ergänzen Sie die Klasse Quader wie folgt: Die Methode `SortiereAbmessungen` sortiert die Quaderabmessungen `d[0]`, `d[1]` und `d[2]` der Grösse nach. Die Methode `PasstIn` gibt den Wert `true` zurück, wenn die Ware in den Karton hineinpasst, andernfalls `false`.

```
class Quader {
private:
    double d[3];                //Abmessungen eines Quaders
                                //(Laenge, Breite, Hoehe)
    bool   bereitsSortiert = false;

    void SortiereAbmessungen() {    // sortiert d[0..2], aber ohne auf-
                                    // wändigen Sortieralgorithmus, da nur
                                    // 3 Elemente sortiert werden müssen

    }
public:
    Quader(double l, double b, double h) {    //Konstruktor
        d[0] = l;   d[1] = b;   d[2] = h;
    }

    bool PasstIn( Quader Karton ) {

    }
};
```

