Hochschule Worms
University of Applied Sciences
Fachbereich Informatik
Prof. Dr. Bernd Ruhland

Objektorientierte Programmierung

Aufgabe u03b Aggregation / Array aus Objekten

Lernziel: Klassendefinition, Attribute und Methoden; **Aggregation** und Codewiederverwendung

Für ein Funkmasten-Register wird ein Verwaltungsprogramm gebraucht:

Eine Besonderheit ist, dass das Programm bereits bei der Datenerfassung prüfen muss, ob eine Überschreitung der Höhe des Funkmasts vorliegt. Die Höhenbegrenzung hängt von der Zahl der Antennen ab. Ein Funkmast mit bis zu vier Antennen darf maximal 30 Meter hoch sein, ein Funkmast mit mehr als vier Antennen darf maximal 50 Meter hoch sein.

Daten:

Das Programm soll die Daten von max. 100 Funkmasten verwalten. Pro Funkmast werden
- Anzahl der Antennen,
- Reichweite,
- Höhe und
- geografische Position
verwaltet. Die geografische Position ist durch die geografische Breiten- und Längenangabe als Gleitkommazahlen festgehalten.

Definieren Sie eine **Klasse „cFunkMast“** mit den geeigneten Datenelementen.

Programmieren Sie einen universellen Konstruktor für diese Klasse. Der Konstruktor soll folgende Vorgabe-Werte eintragen:
- Anzahl Antennen: 0
- Reichweite: 0.0
- Hoehe: 0.0
- geografische Breite: 48.79
- geografische Laenge: 8.17

Und er muss die Höhenkorrektur durchführen.

Eingabe:

Die Klasse soll eine Methode ***eingabe()*** besitzen zur Eingabe der Werte durch den Anwender. Auch diese Methode muss die Höhenkorrektur durchführen. (Tipp: private Methode, die im Konstruktor und bei ***eingabe()*** verwendet wird.)

Ausgabe:

Erstellen Sie eine Methode ***ausgabe()*** für diese Klasse. Diese Funktion gibt die Daten eines Objekts aus, wenn es sich **nicht** um ein „leeres“ also unbesetztes Objekt im Array handelt.

Die Klasse cFunkmast besitzt außerdem folgende **private** Mitgliedsfunktion:

„korrHoeheAntennen()“ zur Überprüfung und Korrektur der Höhe und der Antennenanzahl des Funkmasts wie in der Aufgabenstellung beschrieben. Überlegen Sie sich zunächst passende Typen für Übergabeparameter und Returncode, setzen Sie dann die Funktionalität um.

Überlegen Sie sich, an welchen Stellen Sie die Methode verwenden können.

Aufgabenteil a)

Das Programm soll eine Liste der Funkmasten ausgeben, in Form einer Tabelle: Die Spalten der Tabelle sind „Antennenzahl“, „Leistung“, „Höhe“ und „Position“. Zunächst kommt eine passende Überschrift. Dann folgen so viele Zeilen, wie es Funkmasten gibt, für die Daten eingegeben wurden.

Tipp: Anhand des **Werts 0** des Attributs **Antennen** soll die Ausgabe-Funktion unterscheiden können, ob es sich um ein „leeres“ Objekt handelt.

Definieren Sie im Hauptprogramm die 100 Funkmasten als ein Array aus Objekten der Klasse „cFunkMast“. Befüllen Sie die ersten 5 Objekte mit Werten durch Aufruf der Methode ***eingabe()***.

Aufgabenteil b)

Definieren Sie eine **Klasse „cGeoPos“** für die geografische Position

Mit den privaten Attributen für die Längen- und Breitenangaben (Longitude, Latitude) als Gleitkommazahlen.

Erstellen Sie einen universellen Konstruktor, der die geografische Position über Vorgabewerte auf 0.0 initialisiert.

Erstellen Sie eine öffentliche Funktion setGeoPos(), mit der die Werte gesetzt werden. Beachten Sie dabei, dass die Längenangaben nie größer als +180.0 Grad und nie kleiner als -180.0 sein können, sowie dass die Breitenangaben nie größer als +90.0 Grad und nie kleiner als -90.0 sein können. Rechnen Sie nicht passende Werte um. Ein Kreis hat 360,0 Grad.

Erstellen Sie eine öffentliche Methode printGeoPos(), mit der die Werte ausgegeben werden.

Aggregieren Sie die Klasse „cFunkmast“, indem Sie ein Objekt vom Typ cGeoPos als Attribut aufnehmen. Ersetzen Sie also die beiden Attribute für die geografische Länge und Breite durch ein Objekt der Klasse cGeoPos. Erstellen Sie die innere Konstruktorenkaskade für das Datenelement der Geoposition.

Testen Sie Ihr bestehendes Hauptprogramm mit der geänderten Klasse.