Testaufgabe WiSe 2022

Berechnung der Flugstrecke einer Drohne

Für einen Quadrocopter zur Materialauslieferung soll ein Programm geschrieben werden, mit dessen Hilfe aus einer Liste einzelner Wegepunkte die Gesamtflugstrecke berechnet werden kann.

Vereinfachend wird dabei angenommen, dass sich der Quadrocopter in einer Flughöhe bewegt, in der sich keine Kollisionsprobleme ergeben, sodass die direkte Strecke zwischen zwei Wegepunkten als Entfernung herangezogen werden kann.

Zudem wird vorausgesetzt, dass die Wegepunkte in der Reihenfolge abgeflogen werden, in der sie in das Programm eingegeben wurden, sodass keine Wegeoptimierung durch Umsortieren der anzufahrenden Ziele gefordert wird.

Repräsentation eines Wegepunkts

Zur programmatischen Realisierung eines Wegepunkts ist eine Klasse *Point* zu deklarieren. Diese Klasse hält drei Attribute:

Typ	**Bezeichner**	**Beschreibung**
double	m_latitude	Geografische Breite in Dezimalschreibweise
double	m_longitude	Geografische Länge in Dezimalschreibweise
std::string	m_name	Name, der den geografischen Ort beschreibt

Durch einen speziellen Konstruktor sind diese Attribute zu belegen. Die Attribute können mit einer Methode *void Set(...)* überschrieben werden. Sowohl für den Konstruktor als auch für die *Set* Methode soll der Name des Wegepunkts als Default Parameter realisiert werden. Dabei gilt für die *Set* Methode, dass der Name nicht überschrieben wird, wenn der Parameter ein Leerstring ist, das Attribut aber nicht.

Für die Attribute sind Getter zu schreiben:

- Latitude()
- Longitude()
- Name()

Zusätzlich ist eine Methode *Print()* zu implementieren, mit der (unter der Verwendung von *std::cout*) die Attribute auf der Konsole ausgegeben werden können.

Repräsentation der Wegeliste

Für die Wegeliste ist eine Klasse *PointList* zu deklarieren. Dazu soll der Container *std::vector* der Standardbibliothek eingesetzt werden, der Zeiger auf Elemente der Klasse *Point* halten soll.

Mittels einer Methode

```
void Add(Point& arg)
```

kann ein Wegepunkt der Liste zugefügt werden.

Ebenso wie die Klasse *Point* erhält auch die Klasse *PointList* eine Methode *Print()*, mit der die ganze Liste der Wegepunkte auf der Konsole ausgegeben werden kann. Innerhalb dieser Methode ist die *Print*-Methode der Wegepunkte zu verwenden.

Da der STL Container der Klasse *PointList* Zeiger auf Elemente der Klasse *Point* enthält, ist dafür Sorge zu tragen, dass

(1) die internen Elemente des STL Containers unabhängig von den Objekten an den Methodenschnittstellen sind, damit die Lebenszeit der Containerelemente nicht an die Lebenszeit der Schnittstellenelemente gekoppelt ist,
(2) beim Kopieren einer Wegepunkteliste Original und Kopie voneinander unabhängig sind, also in allen Fällen eine *tiefe Kopie* erzeugt wird.

Berechnung der Wegstrecke

Zur Berechnung der Wegstrekce des Quadrocopters wird ein Rechner implementiert, der durch die Klasse *DistanceCalculator* repräsentiert wird.

Da dieser Rechner nur einmal instanziiert werden muss, wird er als *Singleton* implementiert. Dazu erhält die Klasse ein Attribut

```
static DistanceCalculator * m_pTheInstance;
```

Dieses Attribut ist mit *nullptr* zu initialisieren.

Zudem wird eine statische Methode *GetInstance()* implementiert, die dieses Attribut zurückliefert, steht (am Anfang) das Attribut noch auf *nullptr*, so wird innerhalb dieser Methode eine Instanz auf dem Heap erstellt und dem statischen Attribut zugewiesen. (Achten Sie hier bitte auf eine saubere Garbage Collection!)

Damit die Klasse nicht von außen instanziiert werden kann, wird der Standard Konstruktor (und ggfs. auch alle anderen Konstruktoren) *private* definiert.

Mathematik zur Berechnung

In der Methode *Distance(.)* des Rechners wird die Strecke zwischen zwei Wegepunkten berechnet:

```
double DistanceCalculator::Distance(const Point & from, const Point & to)
{
	double lat1 = GetRadians((from.Latitude() + to.Latitude()) / 2);
	double dx = 111.3 * cos(lat1) * (from.Longitude() - to.Longitude());
	double dy = 111.3 * (from.Latitude() - to.Latitude());

	return sqrt(dx * dx + dy * dy);
}
```

Die hier verwendete Methode *GetRadians* muss als statische Hilfsmethode implementiert werden, die restlichen Funktionen befinden sich in der C-Bibliothek. Zu deren Verwendung ist <*math.h*> zu inkludieren.

Zusammenfügen der Elemente

Zur Verwendung des Rechners wird die Klasse *PointList* um die Methode

```
	double GetDistance();
```

erweitert.

Hier wird in einer Schleife über alle eingegebenen Wegepunkte die Wegstrecke des Quadrocopters berechnet und als Wert (in km) zurückgegeben.

Als Beispiel für eine solche Berechnung seien die folgenden Zeilen eine Orientierungshilfe:

```
	PointList* pWayPoints = new PointList();

	pWayPoints->Add(*(new Point(48.803242, 9.221968, std::string("70372 Stuttgart, Kreuznacher Strasse 47"))));
	pWayPoints->Add(*(new Point(48.811465, 9.229727, std::string("70374 Stuttgart, Gnesener Strasse 69"))));
	pWayPoints->Add(*(new Point(48.801934, 9.235032, std::string("70374 Stuttgart, Ruhrstrasse 50"))));
	pWayPoints->Add(*(new Point(48.803242, 9.221968, std::string("70372 Stuttgart, Kreunzancher Strasse 47"))));

	std::cout << "Distance: " << pWayPoints->GetDistance() << std::endl;
```

Für die Distanz ergibt sich in obigem Beispiel ein Wert von 3,17608 km.