Objektorientierte Programmierung

Einstieg

1. Eine einfache Klasse

(a) Erzeugen Sie mit CMake eine Projektmappe für die von Ihnen verwendete Entwicklungsumgebung. Sie soll zunächst lediglich die Datei `main.cpp` enthalten.

(b) Erstellen Sie in C++ ein Hello-World-Programm mit selbstgewählter Begrüßung.
Bauen Sie die Projektmappe. Beheben Sie ggf. auftretende Fehler. Testen Sie Ihr Programm.

(c) Legen Sie in einem separaten Paar aus Header- (`.hpp`) und Quelltextdatei (`.cpp`) eine neue Klasse `Greeting` an. Implementieren Sie in dieser Klasse lediglich eine öffentliche Methode `void Print() const`, die dieselbe Begrüßung ausgibt, wie Ihre `main`-Funktion. Legen Sie keine weiteren Methoden und auch keinen Konstruktor, Destruktor, etc. an.
Bauen Sie die Projektmappe. Beheben Sie ggf. auftretende Fehler.

(d) Binden Sie die soeben angelegte Headerdatei in Ihrer Datei `main.cpp` ein. Erzeugen Sie in der `main`-Funktion eine Instanz Ihrer Klasse `Greeting`. Ersetzen Sie die vorherige Ausgabe der Begrüßung durch einen Aufruf der `Print`-Methode der angelegten `Greeting`-Instanz.
Bauen Sie die Projektmappe. Beheben Sie ggf. auftretende Fehler. Testen Sie Ihr Programm.

2. Ein eigener Konstruktor

Arbeiten Sie mit der Projektmappe aus der vorherigen Aufgabe weiter.

(a) Legen Sie in Ihrer Klasse `Greeting` einen eigenen Konstruktor an, der eine benutzerdefinierte Begrüßung als Parameter übernimmt, diese in einer privaten Membervariable speichert und in `Print` ausgibt.
Bauen Sie die Projektmappe. Welche Fehlermeldung erhalten Sie? Wieso tritt dieser Fehler auf?

(b) Beheben Sie den Fehler, indem Sie die Erzeugung der `Greeting`-Instanz anpassen.
Bauen Sie die Projektmappe. Testen Sie Ihr Programm mit verschiedenen Begrüßungen. Beheben Sie ggf. weitere auftretende Fehler.

(c) Legen Sie in Ihrer Klasse `Greeting` einen eigenen Destruktor an, der eine selbstgewählte Verabschiedung ausgibt.
Bauen Sie die Projektmappe. Wann und warum wird die Verabschiedung ausgegeben? Beheben Sie ggf. auftretende Fehler.

3. Eine weitere Methode

Arbeiten Sie mit der Projektmappe aus der vorherigen Aufgabe weiter.

(a) Legen Sie in Ihrer Klasse `Greeting` einen neue öffentliche Methode `void Tick()` an. Diese inkrementiert eine neue private Membervariable, die mit `0` initialisiert wird.
Bauen Sie die Projektmappe. Testen Sie Ihr Programm. Beheben Sie ggf. auftretende Fehler.

(b) Ergänzen Sie die Ausgabe der Begrüßung in `Greeting::Print` um die Ausgabe der Anzahl der bisherigen Aufrufe von `Greetings::Tick`.
Rufen Sie in Ihrer `main`-Funktion *vor* der Ausgabe der Begrüßung mehrere Male die Methode `Tick` Ihrer `Greeting`-Instanz auf.
Bauen Sie die Projektmappe. Testen Sie Ihr Programm mit verschiedenen Anzahlen an `Tick`-Aufrufen. Beheben Sie ggf. auftretende Fehler.

Aufgaben für die Abgabe

4. Ein Sportverein

Beginnen Sie mit einer neuen Projektmappe, die lediglich die Datei `main.cpp` enthält.

(a) Legen Sie in einem Paar aus Header- und Quelltextdatei eine Klasse `TeamStatistics` an. Der Konstruktor übernimmt ein Kürzel für den Teamnamen (nur Großbuchstaben, keine Leerzeichen etc.). Eine öffentliche Methode gibt dieses Kürzel aus.

(b) Das Kürzel für den Teamnamen wird zur Laufzeit des Programms in `main` mit der Tastatur eingegeben. Legen Sie anschließend eine Instanz der Klasse `TeamStatistics` an. Danach wird das eingegebene Kürzel mit der geeigneten Methode der angelegten `TeamStatistics`-Instanz wieder am Bildschirm ausgegeben.

(c) Bauen Sie Ihre Projektmappe. Testen Sie Ihr Programm mit verschiedenen Kürzeln. Beheben Sie ggf. auftretende Fehler.

5. Detailliertere Ausgabe

Arbeiten Sie mit der Projektmappe aus der vorherigen Aufgabe weiter.

(a) Legen Sie geeignete Membervariablen in `TeamStatistics` an, um die Anzahl der gespielten Spiele, der insgesamt erhaltenen Punkte sowie der insgesamt geschossenen und erhaltenen Tore zu speichern. Initialisieren Sie diese auf `0`.

(b) Legen Sie eine private Methode zur Berechnung der Tordifferenz an. Diese gibt *vorzeichenbehaftete* Ganzzahlen zurück.

(c) Passen Sie die Ausgabemethode so an, dass die Ausgabe der folgenden Form entspricht:

```
[KÜRZEL] [Spiele] [Punkte] [geschossene Tore]:[erhaltene Tore] [Tordifferenz]
```

Die Tordifferenz soll immer mit Vorzeichen (sowohl + als auch −) ausgegeben werden. Geben Sie dazu unmittelbar vor der Ausgabe der Tordifferenz `std::showpos` und unmittelbar nach der Ausgabe der Tordifferenz `std::noshowpos` in den Ausgabestream.
Beispiel:

```
std::cout << std::showpos << 5 << std::noshowpos;
```

(d) Bauen Sie Ihre Projektmappe. Testen Sie Ihr Programm. Beheben Sie ggf. auftretende Fehler.

6. Spiele

Arbeiten Sie mit der Projektmappe aus der vorherigen Aufgabe weiter.

(a) Legen Sie eine geeignete Methode in `TeamStatistics` an, mit der Sie das Torergebnis eines einzelnen Spiels angeben können (geschossene und erhaltene Tore). Diese Methode aktualisiert die Membervariablen von `TeamStatistics` entsprechend. Ein Sieg gibt drei Punkte, ein Unentschieden einen Punkt.

(b) Passen Sie Ihre `main`-Funktion so an, dass nach Eingabe des Team-Kürzels eine beliebige Anzahl an Spielergebnissen in folgendem Format eingegeben werden kann:

```
[geschossene Tore] [erhaltene Tore]
```

Die Eingabe eines Spielergebnisses wird mit der Enter- bzw. der Return-Taste abgeschlossen. Danach kann ein weiteres Spielergebnis eingegeben werden, bis dem Programm durch Eingabe von Strg+D signalisiert wird, dass der Benutzer keine weiteren Ergebnisse eingeben, sondern das Programm beenden möchte.
Bevor das Programm endet, wird die errechnete Statistik des Teams angezeigt.

(c) Bauen Sie Ihre Projektmappe. Testen Sie Ihr Programm für verschiedene Anzahlen von Spielen mit unterschiedlichem Ausgang. Beheben Sie ggf. auftretende Fehler. Erleichtern Sie sich Ihre Arbeit, indem Sie Pipes verwenden und ggf. die Ausgabe in eine Datei umleiten (siehe vorheriges Praktikum).

(d) Testen Sie Ihr Programm mit folgender Eingabe:

```
WH
0 0
0 1
3 1

```

Denken Sie an den Zeilenwechsel nach dem letzten Spielergebnis.
Es soll folgende Ausgabe entstehen:

```
WH 3 4 3:2 +1
```

Beheben Sie ggf. auftretende Fehler.

Vertiefung

7. Eine Datenstruktur für Spielergebnisse

Arbeiten Sie mit der Projektmappe aus der vorherigen Aufgabe weiter.

(a) Legen Sie in einer neuen Headerdatei eine Datenstruktur für ein Spielergebnis an. Initialisieren Sie die Felder für die geschossenen und die erhaltenen Tore mit `0`.

(b) Lassen Sie `TeamStatistics` nun Instanzen dieser Datenstruktur verarbeiten anstelle von einzelnen Parametern und Membervariablen für die geschossenen und erhaltenen Tore.

(c) Lesen Sie das Spielergebnis von der Tastatur nicht in zwei einzelne Variablen ein sondern in die beiden Felder einer Instanz der neuen Datenstruktur. Übergeben Sie diese Instanz an die entsprechende Methode Ihrer `TeamStatistics`-Instanz.

(d) Fügen Sie der oben angelegten Header-Datei drei `inline`-Funktionen hinzu, die eine Instanz der Datenstruktur als Parameter aufnehmen:

- Eine gibt die Tore im passenden Format aus,
- eine ermittelt die Tordifferenz (verschieben Sie diese Methode aus `TeamStatistics`),
- eine gibt die Tordifferenz im passenden Format aus.

Passen Sie die Ausgabemethode in `TeamStatistics` entsprechend an.