Objektorientiertes Programmieren

Übungsbeispiele

1 Parallel Deep Miner

Verwenden Sie das vierte Beispiel als Basis und erweitern Sie es um eine parallele Ausführung der Roboterbewegungen. Nun soll nur mehr der Computer gegen sich selbst antreten können. Folgende Punkte müssen dabei beachtet werden:

- Verwenden Sie Threads und Mutexes, um eine fehlerfreie Parallelität zu gewährleisten.
- Verwenden Sie für Datenstrukturen Klassen der STL.
- Nutzen Sie Interfaces als Elternklassen und Polymorphismus für Ihre Objekte.
- Verwenden Sie sinnvolle Klassen. Beachten Sie dabei die richtige Verwendung von Zugriffsmodifikatoren, Gettern/Settern, den verschiedenen Konstruktoren und Destruktoren sowie der sinnvollen Strukturierung von Funktionen und Daten. Bilden Sie Ihre Klassen in einem entsprechenden Diagramm ab.
- Überprüfen Sie alle Parameterübergaben an Funktionen und Benutzereingaben auf Fehler und verhindern Sie so, dass Ihr Programm bei ungültigen Eingaben nicht mehr richtig funktioniert, Eingaben sollen so lange wiederholt werden, bis sie korrekt sind und der Spielfluss erst dann fortgesetzt werden.
- Testen Sie Ihren Code ausgiebig und berücksichtigen Sie Randbedingungen.

Stufe 1

Siehe Beispiel 4, es sollen aber mindestens 5 Roboter gleichzeitig in der Spielwelt sein, diese sollen als Threads implementiert werden und Mutexes verwenden, um race conditions zu verhindern. Geben Sie zu Beginn der Ausführung die Summe der abbaubaren Werte der Spielwelt aus und am Ende die Summe der Roboterpunkte. Diese Summen müssen gleich sein, ansonsten wurden Werte falsch abgebaut.

Stufe 2

Ihr Programm soll nun auch die Zeit messen, die jeder einzelne Thread und Ihr Programm insgesamt für die Ausführung braucht. Recherchieren Sie dazu wie Zeitmessung in C++ möglich ist (siehe chrono-Bibliothek).

Stufe 3

Erweitern Sie die Roboter um die Möglichkeit, sich gegenseitig während des Sammelns auch zu bekämpfen. Befindet sich ein Roboter auf einem benachbarten oder demselben Feld, führen sie vor dem Abbauen von Werten auch einen Angriff auf den anderen Roboter durch. Dieser soll natürlich trotz Parallelität fehlerfrei ablaufen, wie Sie die Regeln genau ausgestalten, bleibt Ihnen überlassen, ein Roboter soll aber nicht durch einen einzigen Angriff aus dem Spiel genommen werden können, oder wenn, dann nur mit sehr geringer Wahrscheinlichkeit.

Bewertung

Aspekt	**Bewertung**
Threads	15%
Mutexes	10%
Diagramm	5%
Fehlerprüfung	10%
Stufe 1	30%
Stufe 2	15%
Stufe 3	15%