Fortgeschrittene objektorientierte Programmierung
LVA 185.211, VU, 3 ECTS, 2014 S

3. Übungsaufgabe

Lernen Sie Eiffel und entwickeln Sie ein Programm in Eiffel (siehe unten). Jedes Mitglied einer Übungsgruppe soll sich in etwa gleichem Maße an dieser Aufgabe beteiligen. Ziel der Aufgabe ist das Sammeln von Erfahrungen zu

- dynamisch überprüften Zusicherungen,
- kovarianten Eingangsparametertypen und “CAT-Calls”
- sowie einigen weiteren praktischen Eigenschaften von Eiffel.

Zur Lösung der Übungsaufgabe verwenden Sie am besten ISE-Eiffel. Das als GNU-Projekt in Konkurrenz dazu entwickelte SmartEiffel ist weder mit dem Eiffel-Standard noch mit ISE-Eiffel kompatibel und scheint seit einiger Zeit nicht mehr weiterentwickelt zu werden.

ISE-Eiffel:

Die Firma ISE wurde von Bertrand Meyer, dem Entwickler von Eiffel, gegründet und vertreibt vor allem Eiffel-Produkte und Support. Die Eiffel-Entwicklungsumgebung von ISE, EiffelStudio, ist für die Entwicklung von Open-Source-Software auf vielen Plattformen frei verfügbar. Die kommerzielle Version unterscheidet sich davon nur in den Lizenzvereinbarungen und im Support. Sowohl zu EiffelStudio als auch zu Eiffel selbst finden Sie umfangreiche Online-Dokumentation.

Fragen:

Beantworten Sie anhand einer Beispielaufgabe (siehe unten) folgende Fragen:

- Wie hoch ist der Aufwand um Zusicherungen im Eiffel-Code zu formulieren?
- Wie stark wirkt sich die Überprüfung von Zusicherungen auf die Laufzeit aus?
- Vorbedingungen dürfen im Untertyp nicht stärker und Nachbedingungen nicht schwächer werden um Ersetzbarkeit zu garantieren. Der Eiffel-Compiler überprüft diese Bedingungen. Ist es (trotz eingeschalteter Überprüfung von Zusicherungen) möglich, diese Bedingungen zu umgehen? Wenn ja, wie?
- Eiffel erlaubt kovariante Eingangsparametertypen. Unter welchen Bedingungen führt das zu Problemen, und wie äußern sich diese? Können Sie ein Programm schreiben, in dem die Verwendung kovarianter

Eingangsparametertypen zu einer Exception führt?
- Vereinfachen kovariante Eingangsparametertypen die Programmierung? Unter welchen Bedingungen ist das so?
- Welche Spracheigenschaften von Eiffel finden Sie interessant und würden Sie gerne auch in anderen Sprachen sehen? Welche Eigenschaften von Eiffel empfinden Sie dagegen als störend?

Aufgabe:

Folgende Aufgabenstellung soll als Anregung dienen. Sie können auch jede beliebige andere Aufgabe vergleichbaren Umfangs in Eiffel lösen. Es geht nicht um die Lösung der Aufgabe selbst, sondern um die Beantwortung obiger Fragen. Alle Mitglieder jeder Übungsgruppe sollen dabei mitwirken.

Schreiben Sie ein Programm zur Verwaltung der Ergebnisse von Mäuserennspielen. Jedes Spiel verwendet eine bestimmte Version des Rennprogramms und erstreckt sich über mehrere Runden mit teilweise unterschiedlichen Levels. Die Anzahl der Siege der einzelnen Spieler(innen) in jedem Level werden festgehalten und können später abgefragt und nach verschiedenen Kriterien ausgewertet werden. Auch die unterschiedlichen Versionen des Rennprogramms und deren Levels sollen zusammen mit deren Entwickler(inne)n festgehalten und abgefagt werden können. Beispielsweise soll es möglich sein, herauszufinden, ob die Spielergebnisse dadurch beeinflusst werden, dass Spieler(innen) auch als Entwickler(innen) der entsprechenden Versionen oder Levels mitgewirkt haben, oder ob Spielergebnisse mit der Anzahl der Teilnahmen an Spielen (mit konkreten Versionen bzw. Levels oder allgemein) oder an der Entwicklung korrelieren.

Zur Erreichung des Ziels ist es günstig, zwischen verschiedenen Arten von Personen zu unterscheiden: Spieler(innen), Entwickler(innen) von Versionen und Entwickler(innen) von Levels. Natürlich kann ein und dieselbe Person zu mehreren dieser Arten gehören, und in verschiedenen Programmteilen sind durch Verwendung von Typen nur bestimmte Arten von Personen zugelassen. Achten Sie darauf, dass Sie durch mehrere Typen von Personen und mehrere Typen verwalteter Daten (Spielergebnisse, Versionen, Levels, etc.) parallele Typhierarchien einführen, sodass sich Einsatzmöglichkeiten für CAT-Calls ergeben.

Bei der Verwendung von Zusicherungen sind der Fantasie kaum Grenzen gesetzt. Beispielsweise könnte man die Anzahl der Runden, Levels und Spieler(innen) pro Spiel sowohl nach unten als auch nach oben begrenzen, eventuell auch abhängig von bestimmten Versionen und Levels. Man könnte auch die Anzahl der Teilnahmen von Personen an Spielen bzw. in der Entwicklung in einem bestimmten Zeitraum nach unten und oben beschränken, vielleicht abhängig von der Art der Personen. Überlegen Sie sich Zusicherungen, welche die Möglichkeiten und vor allem Grenzen der Verwendung von Zusicherungen demonstrieren.

Nehmen Sie im Nachhinein Änderungen an allen Teilen des Programms vor, hauptsächlich solche, die sich auf Zusicherungen auswirken. Führen Sie bewusst Situationen herbei, in denen Zusicherungen verletzt und CAT-Calls nötig werden, die Ausnahmen werfen. Versuchen Sie das System in diesen Situationen zu überlisten, sodass das Programm ohne Ausnahmen läuft, obwohl die eigentlichen Ursachen der Probleme noch immer existieren.

Achten Sie generell auf ein sinnvolles Sichtbarmachen von “Features” in anderen Programmteilen sowie auf Eiffel-typische Kommentare (= sprechende Kommentare an den richtigen Stellen).