ulm university universität ulm

2. Zulassungsklausur zur Veranstaltung

Objektorientierte Programmierung

im Sommersemester 2023

Prüfer: Prof. Dr. Matthias Tichy Fakultät Ingenieurwissenschaften, Informatik, Psychologie

01.07.2023, 09-10:15 Uhr **Bearbeitungszeit: 45 min**

Nachname:	Vorname:	Matrikelnummer:
Studiengang und Abschluss:		Account:

Hiermit erkläre ich, dass ich prüfungsfähig bin.
Sollte ich nicht auf der Liste der angemeldeten Studierenden aufgeführt sein, dann nehme ich hiermit zur Kenntnis, dass diese Prüfung nicht gewertet werden wird.

_____

Datum, Unterschrift des Prüfungsteilnehmers

Zur allgemeinen Beachtung:

- Füllen Sie das Deckblatt vollständig und korrekt aus.
- Lesen Sie sich zunächst die Klausur sorgfältig durch (sie besteht aus 3 Seiten).
- Zum Erreichen der Vorleistung benötigen Sie **12** der insgesamt *24* Punkte.
- Im Ordner ∼/`materials` finden Sie das zu bearbeitende Projekt. Zum Import mittels Eclipse verwenden Sie 'Import → Existing Projects into workspace → oop_admission_exam_02_groupA_eclipse'. Zum Import mittels IntelliJ verwenden Sie 'Open → oop_admission_exam_02_groupA_intellij'.
- Sie dürfen im Projekt beliebig Hilfsmethoden und Klassen hinzufügen. Die Signaturen der vorgegebenen Methoden und Interfaces dürfen nicht verändert werden. Die schon implementierten Methoden dürfen nicht geändert werden.
- **Vor Beginn der Bearbeitungszeit** haben Sie 5 Minuten Zeit, um einen Editor/eine IDE Ihrer Wahl zu starten und sich in den Materialien zurecht zu finden.
- **Nach Ablauf der Bearbeitungszeit** haben Sie 5 Minuten Zeit, um Ihre Lösung im Ordner ∼/`export` abzulegen. **Bleiben Sie sitzen!**
- Abgaben sind als Projektexport in Zip-Form abzugeben. Achten Sie darauf Ihr **gesamtes** Projekt, mit allen zugehörigen Ordnern, zu exportieren.
- Abgaben die Compilerfehler produzieren werden nicht bewertet.

Punkteverteilung				
1 von 3	2 von 12	3 von 9	$\sum$ von 24	**Note**
				Korrektur

Das Projekt

Das Ihnen zur Verfügung gestellte Projekt soll ein Turnier in einem Spiel (Tic-Tac-Toe) implementieren. Bei diesem Turnier treten eine Menge von Spielen gegen einander an (jeder gegen jeden). Ein Spieler spielt jeweils zwei Spiele gegen jeden anderen Spieler. Am Ende wird der Sieger des Turniers über die Anzahl an gewonnenen Spielen bestimmt.
Das Projekt besteht aus den folgenden 6 Klassen. Im Folgenden finden Sie eine kurze Übersicht über die einzelnen Klassen:

oop.sose2023.admission_exam.Main Diese Klasse initialisiert ein neues Turnier zwischen einer Menge von Spielern, und startet das Turnier.

oop.sose2023.admission_exam.group01.Game Diese Klasse implementiert den Spielzustand und die Spiellogik für ein einzelnes Spiel zwischen zwei Spielern.

oop.sose2023.admission_exam.group01.Player Diese Klasse implementiert den Zustand der einzelnen Spieler.

oop.sose2023.admission_exam.group01.RankingElement Diese Klasse speichert die Daten für die Abschlusstabelle zu einem einzelnen Spieler. Siehe Aufgabe 1.

oop.sose2023.admission_exam.group01.Tournament Diese Klasse implementiert ein Turnier, in dem jeder Spieler gegen jeden anderen Spieler jeweils in zwei Spielen spielt.

oop.sose2023.admission_exam.group01.util.InputManagement Diese Klasse implementiert die Logik, um Eingaben über die Konsole als Koordinaten zu interpretieren.

Aufgabe 1 - Ranking Element **3 Punkte**

Implementieren Sie die Klasse `RankingElement`, die diejenigen Informationen über einen Spieler verwaltet, die benutzt werden, um das die Abschlusstabelle zu erstellen. Abbildung **1** zeigt das Klassendiagramm für die zu implementierende Klasse `RankingElement`. Ihre Implementierung soll nur die dargestellten Attribute und Methoden implementieren. Achten Sie auf die Schreibweise der Namen, die angegebenen Sichtbarkeiten und die festgelegten Typen. Übernehmen Sie auch die im Klassendiagramm vorgegebenen Sichtbarkeiten: **Private** Attribute oder Methoden werden mit einem vorangestellten – gekennzeichnet. **Öffentliche** Attribute oder Methoden werden mit einem vorangestellten + gekennzeichnet. Die Klasse soll nur einen Konstruktor zur Verfügung stellen. Dieser soll einen Spieler-Namen und die Anzahl der Siege des Spielers übergeben bekommen. Methoden, die mit dem Präfix `get` beginnen, sollen jeweils den Wert des gleichnamigen Attributes zurückgeben. Methoden, die mit dem Präfix `set` beginnen, sollen jeweils das gleichnamige Attribut auf den übergebenen Parameterwert setzen.

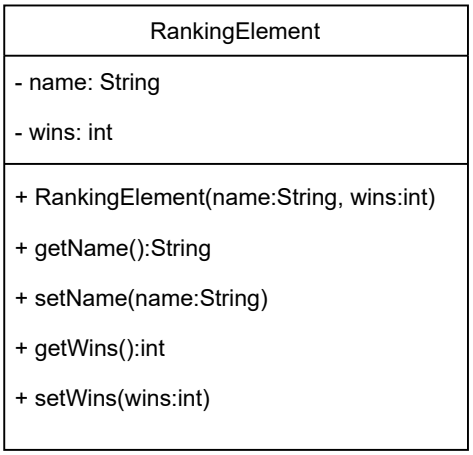

Abb. 1: Klassendiagramm zur Verwaltung von Rankinginformationen.

Aufgabe 2 - Spiele erzeugen **9 Punkte**

Implementieren Sie in der Klasse `Tournament` die Methode `createTournamentGames()`. Diese Methode erzeugt für die Spieler in einem Turnier eine Liste von Spielen, sodass jeder Spieler gegen jeden anderen Spieler zwei Spiele spielt. In jedem Paar von Spielen zwischen zwei Spielern soll jeder Spieler einmal das Startrecht haben (also der erste Spieler sein). Die Reihenfolge der Spiele ist egal.

Beispiel: Bei drei Spielern (Spieler1, Spieler2 und Spieler3) sollen die folgenden Spiele in beliebiger Reihenfolge erzeugt werden:

- Spieler1 gegen Spieler2
- Spieler2 gegen Spieler1
- Spieler1 gegen Spieler3
- Spieler3 gegen Spieler1
- Spieler2 gegen Spieler3
- Spieler3 gegen Spieler2

Aufgabe 3 - Ranking erstellen **12 Punkte**

Implementieren Sie in der Klasse `Tournament` die Methode `calculateRanking()`. Diese Methode berechnet die Abschlusstabelle (Ranking) eins Turniers unter den Spielern. Ein Sieg in einem Spiel zählt einen Punkt für den siegreichen Spieler. Die Abschlusstabelle wird nach der Anzahl dieser Punkte sortiert. Die Reihenfolge zwischen Spielern, die gleich viele Punkte haben ist egal. Geben Sie das Ergebnis in Form einer `List` von Spielern zurück. In der zurückgegebenen Liste sollen die Spieler als Erstes vorkommen, die die meisten Siege erzielt haben. Verwenden Sie für ihre Implementierung die Informationen, die in der `Player` Klasse gespeichert sind.

Beispiel:

- Spieler1 hat 6 Siege.
- Spieler2 hat 3 Siege.
- Spieler4 hat 4 Siege.
- Spieler5 hat 4 Siege.

Das Ergebnis der `calculateRanking`-Funktion ist eine Liste mit den Inhalten in der folgenden Reihenfolge (mit `RankingElement` als RE): `[RE("Spieler1", 6), RE("Spieler4", 4), RE("Spieler5", 4), RE("Spieler2", 3)]` oder `[RE("Spieler1", 6), RE("Spieler5", 4), RE("Spieler4", 4), RE("Spieler2", 3)]`.