Klausur B-AI2 PM 2 SS 2016, Prof. Dr. Bernd Kahlbrandt

Matrikelnummer	Vorname	Name

Aufgabe	1.1	1.2	1.3	1.4	2	3.1	3.2		
Punkte (ca.)	4	4	4	4	15	8	8		
Erreicht									
Aufgabe	4	5.1	5.2	5.3	6.1	6.2	6.3	$\sum$	Note
Punkte (ca.)	6	1	3	9	5	3	10	84	
Erreicht									

Datum: **27.06.2016** Uhrzeit: **09:00**
Raum: **BT 7, 11.Stock** Dauer (Minuten): **180**

Hinweise

1. Verfügbare Hilfsmittel: Java API Doc, Java Source Code, Laufwerk M u. a. mit Java Sprachspezifikation, Spezifikation der JVM, Skript, Folien (pdf), Abts Grundkurs Java, Goll Java als erste Programmiersprache (beides pdf).

2. Erlaubte „Mitbringsel“: Vorlesungmitschrift inklusive Skript/Folien mit Notizen, Wörterbuch Englisch oder Muttersprache, elektronisches Wörterbuch ohne Internetzugang („Flugmodus“), Papier, Stifte.

3. Notwendige Bestandteile:

   3.1. In **jeder** .java-Datei steht unmittelbar vor dem „Top-Level Element“ (Klasse, Annotation etc.) ein javadoc-Kommentar mit Ihrem Namen nach einem *@author*-tag.

   3.2. Ihre Entwurfsentscheidungen sind in javadoc-Kommentaren oder //-Kommentaren knapp und nachvollziehbar begründet. Das gilt auch für Testfälle.

   Meine Vorstellung dazu: Ein javadoc-Kommentar (s. o. unter 3.1) mit den Entwurfsentscheidungen für die Klasse. Pro Testmethode ein javadoc-Kommentar mit der Begründung für die gewählten Testfälle und die erwarteten Ergebnisse.

   3.3. Die Verwendung der Annotation *@SupressWarnings* ist nur zulässig, wenn Sie bei dieser Annotation in einem javadoc-Kommentar überzeugend darlegen, warum die jeweilige Warnung gefahrlos unterdrückt werden kann!

   3.4. Die Java-Codekonventionen müssen eingehalten werden!

1 Utility-Klasse schreiben

Schreiben Sie bitte im Paket *a01* eine *Utility-Klasse Collections02* mit folgenden Methoden:

1. Eine generische Methode, die eine *Collection* von *Collections* von T übergeben bekommt und die durchschnittliche Größe der *Collection*s zurückgibt!

2. Eine generische Methode *getBiggest*, die aus einer *Collection* von *Comparable*s das größte Element ermittelt und zurückgibt!

3. Eine Methode *sumExact01*, die eine Liste von ganzen Zahlen übergeben bekommt und die Summe der Zahlen zurückgibt! Tritt ein numerischer Überlauf auf, so werfen Sie bitte eine *ArithmeticException*!

4. Schreiben Sie analog zur vorstehenden Aufgabe eine Methode *sumExact02*, die für eine Liste ganzer Zahlen die korrekte Summe ermittelt, wenn das möglich ist und eine *ArithmeticException* wirft, wenn dies wegen Überlauf nicht möglich ist.

   Hier sollen Sie folgende Situation berücksichtigen: Es kann sein, dass es beim Aufsummieren der Einträge einer Liste von ganzen Zahlen zwischenzeitlich einen Überlauf gibt, am Ende aber ein gültiges Ergebnis herauskommt oder auch nicht.

   Hinweis: Die Aufgaben 3 und 4 dieses Teiles lassen sich auf einmal erledigen. Um denen eine weitere Chance zu geben, die nicht auf Anhieb auf die allgemeine Lösung kommen, habe ich dies aber in 4 präzisiert.

2 Generische Klasse schreiben

Schreiben Sie bitte im Paket *a02* eine generische Klasse *Stack*, die das vorgegebene Interface *IStack* implementiert!

Die Einträge im Stack verwalten Sie bitte über Objekte einer inneren Klasse, wie in Abb. 1 skizziert. Ein Objekt dieser Klasse enthält das jeweils „oberste“ Element des Stacks und eine

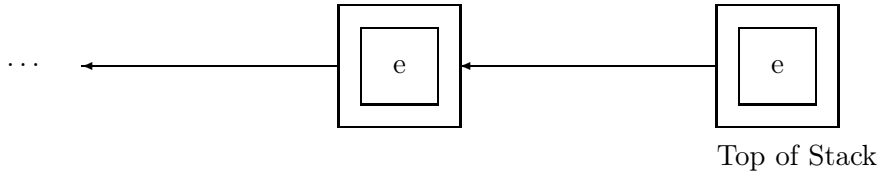

Abb. 1: Innere Klasse

Referenz auf das vorhergende Objekt der inneren Klasse, das das vorhergehende Element enthält usw.

3 Datum, Uhrzeit, Zeiträume

Schreiben Sie den Code für diese Aufgabe bitte im Paket *a03*!

Ihr Prof möchte für die am 27.06.2016 stattfindende Klausur am 01.07.2016 um 12:00 Klausureinsicht anbieten.

1. Schreiben Sie bitte *a03* Code, mit dem Werte wie die folgenden ermitteln werden können!

   - Wieviele Minuten hat er dann für die Durchsicht, wenn er am Dienstag, 28.06.2016 um 6:00 beginnt?
   - Am 04.06.2016 waren 54 Studierende angemeldet. Wenn 50 Studierende teilnehmen, wieviele Minuten hat er dann pro Klausur? Wieviele, wenn er 8 Stunden pro Tag Pause macht?
   - Wieviel Zeit hat er unter diesen Annahmen, wenn im gleichen Zeitraum auch noch 30 weitere Klausuren zu bewerten sind?

2. Schreiben Sie bitte aussagefähige Testfälle nicht nur, aber auch unter Verwendung der angegeben Beispieldaten!

4 Unbegrenzte Streams

Schreiben Sie den Code zu dieser Aufgabe bitte in Paket *a04*!

Ein sogenanntes compostelanisches *Heiliges Jahr* findet statt, wenn der *25.07.* auf einen Sonntag fällt. Das erste Heilige Jahr wurde im Jahr 1126 begangen. Definieren Sie bitte einen unbegrenzten Stream, der die Heiligen Jahre liefert! Ermitteln Sie bitte aus diesem Stream die Heiligen Jahre zwischen dem Jahr 2000 und dem Jahr 2100, jeweils einschließlich! Geben Sie das Ergebnis bitte als *List* zurück.

Welche Abstände gibt es zwischen diesen Jahren?

5 Datei einlesen und verarbeiten

Schreiben Sie den Code zu dieser Aufgabe bitte in Paket *a05*!

Das Verzeichnis *data* enthält in der Textdatei *picasso.txt* ein Gedicht von Robert Gernhardt. Der Dichter wurde mit einem Doppelreim „beschenkt“:

> Werd ich nicht nach Tarif bezahlt,
> wird ab sofort naiv gemalt.

Da Doppelreime „doppelt“ schwierig sind, „baute“ er erst einige Jahre später um diesen Doppelreim herum dieses 1997 in „Lichte Gedichte“ veröffentlichte Gedicht aus Doppelreimen.

Dies zur Motivation der folgenden Aufgaben, die Sie bitte in Java lösen und *nicht* durch manuelles Abzählen oder Ähnliches!

1. Lesen Sie bitte die Datei *picasso.txt* in eine geeignete Datenstruktur ein!

2. Der wievielte Doppelreim ist derjenige, der den Anstoß zu dem Gedicht gab?

3. Um Reimworte zu finden, gibt es z. B. das Reimlexikon [1]. Dort finden Sie Worte zu Endsilben. Zu „-at“(mit langem a) finden Sie dort ca. 200 Einträge, aus denen Nonsense-Gedichte erstellt werden können, wie dieses:

   > Der Advokat
   > aß grad Salat
   > als ihm ein Schrat
   > die Saat zertrat

   Da ich Ihnen (noch) keine Bibliothek zur Zerlegung von deutschen Wörten in Silben liefern kann, beschränkt sich diese Aufgabe auf die letzen Worte jeder Zeile und die letzten darin zusammenhängend vorkommenden Konsonanten sowie den direkt davorstehenden Vokal.

   Ordnen Sie bitte die Zeilen des Gedichts nach diesem Kriterium. Zum Ausprobieren Ihres Codes können Sie obiges Nonsense-Gedicht verwenden.

   Speichern Sie bitte zu den Zeilenenden im oben angegebenen Sinn (z. B. „at“) die jeweils letzten Worte der Zeilen aus dem Gedicht in *picasso.txt*. Geben Sie aus dieser Struktur zu jeder Endung die Worte aus, die sich im obigen Sinne „reimen“!

6 Annotationen und Reflection

Bitte schreiben Sie den Code für diese Aufgabe im Paket *a06*!

1. Schreiben Sie bitte eine wiederholbare Annotation *Schedule*, mit der Methoden annotiert werden können! In der Annotation *Schedule* kann angegeben werden, dass die entsprechende Methode

- an einem Tag der Woche, z. B. am Sonntag,
- an einem Tag im Monat, z. B. dem ersten oder letzten Tag (default: letzter Tag des Monats),
- zu einer bestimmten Uhrzeit, z. B. um 12:00

ausgeführt werden soll. Verwenden Sie ggf. vorhandene geeignete Enums!

2. Schreiben Sie bitte eine Klasse mit je einer Methode, die

   - an jedem Montag, Mittwoch, Freitag bzw.
   - am 27. Tag eines Monats, bzw.
   - an jedem Tag um 11:11

   ausgeführt werden sollen und annotieren Sie sie geeignet! Die Methoden sollen nur ihren Namen ausgeben (reflexiv ermitteln!) sowie Datum und Uhrzeit der Ausführung.

   Hinweis: Es gibt eine Methode *getEnclosingMethod* in *Class*, deren API-Dokumentation Sie genau lesen sollten. Das sollte Sie auf eine Idee bringen, um die Information für die Ausgabe zu bekommen.

3. Schreiben Sie bitte Code, der für alle Klassen in diesem Verzeichnis (*a06*) die Methoden findet, die mit *@Schedule* annotiert sind und sie nach folgenden Regeln ausführt:

   - Soll die Methode an einem bestimmten Wochentag ausgeführt werden, falls dies gerade der aktuelle Wochentag ist.
   - Soll die Methode an einem bestimmten Tag des Monats ausgeführt werden, falls dies gerade dieser Tag des Monats ist.
   - Soll die Methode zu einer bestimmten Uhrzeit ausgeführt werden, wenn diese erreicht oder höchstens um drei Stunden überschritten wurde.

Literatur

[1] Willy Steputat. *Reimlexikon. Neu bearbeitet von Angelika Fabig.* Philipp Reclam jun., Stuttgart, 1997, 398 Seiten.