Karlsruher Institut für Technologie

Programmieren – Wintersemester 2017/18

Architecture-driven Requirements Engineering (ARE)
`https://sdqweb.ipd.kit.edu/wiki/Programmieren`
Jun.-Prof. Dr.-Ing. Anne Koziolek

Abschlussaufgabe 2

Ausgabe: 23.02.2018 – 13:00 Uhr
Abgabe: 24.03.2018 – 06:00 Uhr

Bearbeitungshinweise

- Achten Sie darauf, nicht zu lange Zeilen, Methoden und Dateien zu erstellen.[1]
- Programmcode muss in englischer Sprache verfasst sein.
- Kommentieren Sie Ihren Code angemessen: So viel wie nötig, so wenig wie möglich. Die Kommentare sollen einheitlich in entweder englischer oder deutscher Sprache verfasst werden.
- Wählen Sie geeignete Sichtbarkeiten für Ihre Klassen, Methoden und Attribute.
- Verwenden Sie keine Klassen der Java-Bibliotheken ausgenommen Klassen der Pakete `java.lang`, `java.io` und `java.util`, es sei denn, die Aufgabenstellung erlaubt ausdrücklich weitere Pakete.[1]
- Achten Sie auf fehlerfrei kompilierenden Programmcode.[1]
- Halten Sie alle Whitespace-Regeln ein.[1]
- Halten Sie die Regeln zu Variablen-, Methoden- und Paketbenennung ein und wählen Sie aussagekräftige Namen.[1]
- Halten Sie die Regeln zur Javadoc-Dokumentation ein.[1]
- Nutzen Sie nicht das default-Package.[1]
- Halten Sie auch alle anderen Checkstyle-Regeln ein.
  Prinzipiell kann ein Nichteinhalten der Checkstyle-Regeln zu Punktabzug führen.
- `System.exit` und `Runtime.exit` dürfen nicht verwendet werden.[1]
- Achten Sie darauf, genau die vorgegebenen Ein- und Ausgabeformate einzuhalten.
- Allgemeiner Hinweis: Bei Regelverletzung wird der Praktomat Ihre Abgabe zurückweisen.

Abgabemodalitäten

Die Praktomat-Abgabe wird am **Freitag, den 09.03.2018, um 13:00 Uhr**, freigeschaltet. Laden Sie die `Terminal`-Klasse nicht mit hoch.

- Geben Sie Ihre Antworten zu Aufgabe A als `*.java`-Dateien ab.

[1] Der Praktomat wird die Abgabe zurückweisen, falls diese Regel verletzt ist.

Planen Sie für die Abgabe ausreichend Zeit ein, für den Fall, dass der Praktomat Ihre Abgabe wegen einer Regelverletzung ablehnen sollte. Hinweis: Beachten Sie, dass die öffentlichen Tests für eine erfolgreiche Abgabe erfüllt sein müssen.

Terminal-Klasse
Laden Sie für diese Aufgabe die Terminal-Klasse herunter und platzieren Sie diese unbedingt im Paket `edu.kit.informatik`. Die Methode `Terminal.readLine()` liest eine Benutzereingabe von der Konsole und ersetzt `System.in`. Die Methode `Terminal.printLine()` schreibt eine Ausgabe auf die Konsole und ersetzt `System.out`. Verwenden Sie für jegliche Konsoleneingabe oder Konsolenausgabe die Terminal-Klasse. Verwenden Sie in keinem Fall `System.in` oder `System.out`. Fehlermeldungen werden ausschließlich über die Terminal-Klasse ausgegeben und müssen aus technischen Gründen unbedingt mit `Error` beginnen. Laden Sie die Terminal-Klasse **niemals** zusammen mit Ihrer Abgabe hoch.

Fehlerbehandlung
Ihre Programme sollen auf ungültige Benutzereingaben mit einer aussagekräftigen Fehlermeldung reagieren. Aus technischen Gründen muss eine Fehlermeldung unbedingt mit `Error,` beginnen. Eine Fehlermeldung führt nicht dazu, dass das Programm beendet wird; es sei denn, die nachfolgende Aufgabenstellung verlangt dies ausdrücklich. Achten Sie insbesondere auch darauf, dass unbehandelte `RuntimeExceptions`, bzw. Subklassen davon – sogenannte *Unchecked Exceptions* – nicht zum Abbruch Ihres Programms führen sollen.

Objektorientierte Modellierung
Achten Sie darauf, dass Ihre Abgaben sowohl in Bezug auf objektorientierte Modellierung, als auch Funktionalität bewertet werden.

Öffentliche Tests
Bitte beachten Sie, dass das erfolgreiche Bestehen der öffentlichen Tests für eine erfolgreiche Abgabe nötig ist. Planen Sie entsprechend Zeit für Ihren ersten Abgabeversuch ein.

A Olympische Spiele

Traditionell werden seit 1896 die Olympischen Spiele (Beginn mit Spiele der I. Olympiade) im regelmäßigen Turnus ausgetragen. Dieses Jahr begannen die Olympischen Winterspiele am 09. Februar 2018 und finden im südkoreanischen Pyeongchang statt. Aus aktuellem Anlass soll daher vereinfacht die Grundfunktionalitäten eines Verwaltungs- und Archivierungssystems für die Olympischen Spiele in dieser Aufgabenstellung implementiert werden.

Sportstätte
Die Wettkämpfe der Olympischen Spiele werden in Sportstätten ausgetragen. Jede Sportstätte mit einem festen Sitz besitzt einen Namen und hat eine bestimmte Anzahl an Zuschauerplätzen. Alle Sportstätten zeichnen sich durch eine eindeutige `ID` aus.

Sportart und Sportdisziplin
Bei den Olympischen Winterspielen werden verschiedene Sportarten ausgetragen, die sich weiter in diverse Sportdisziplinen aufteilen lassen. So ist beispielsweise Eislauf eine der bei den Olympischen Winterspielen ausgetragenen Sportarten, die sich aus den Disziplinen Eiskunstlauf, Eisschnelllauf und Shorttrack zusammensetzt.

Wettkampf
In diesem Verwaltungs- und Archivierungssystem werden nur Einzelwettkämpfe hinterlegt. Mannschaftssportarten werden zunächst nicht berücksichtigt.

Teilnehmer - Athleten
Für alle stattfindenden Wettkämpfe sind Athleten als Teilnehmer der Olympischen Spiele gemeldet. Sie stellen eine Einzelperson dar, die u. a. unter einer eindeutigen `ID` und ihrem Herkunftsland (Nationalität) geführt wird.

Medaillenspiegel
Bei internationalen Sportveranstaltungen wie den Olympischen Winterspielen wird ein Medaillenspiegel in Form einer Auflistung der sportlichen Erfolge nach Nationalitäten erstellt. Ein sogenannten ewiger Medaillenspiegel gibt an, welches Land absolut gesehen bei sämtlichen Olympischen Winterspielen die meisten Medaillen erringen konnte.

Sortierung
Im Zuge der Aufgabenstellung werden alphabetische bzw. numerische Sortierungen für die Zeilen der Ausgaben verlangt. Eine genaue Beschreibung bzw. Vorgaben befindet sich immer in den Unterabschnitten der Befehle. Sie können im Folgenden davon ausgehen, dass alle `String`-Eingaben Kleinbuchstaben verwenden.

A.1 Interaktive Benutzerschnittstelle

Ihr Programm arbeitet ausschließlich mit Befehlen, die nach dem Programmstart über die Konsole mittels `Terminal.readLine()` eingelesen werden. Nach Abarbeitung eines Befehls wartet das Programm auf weitere Befehle, bis das Programm durch die Eingabe des `quit`-Befehls beendet wird. Bei Programmstart enthält Ihr Verwaltungs- und Archivierungssystems keine weiteren Angaben bzw. Daten.

Achten Sie darauf, dass durch Ausführung der folgenden Befehle die Spezifikationen nicht verletzt werden dürfen. Geben Sie im Fall der Eingabe eines Befehls, der direkt zu einer Verletzung der Spezifikation führt, eine aussagekräftige Fehlermeldung aus. Geben Sie auch eine Fehlermeldung aus, falls kein, ein oder mehrere falsche oder nicht korrekt zu interpretierende Parameter eingegeben wurden. Das heißt, wenn eine Eingabe nicht der hier vorgegebenen Spezifikation entspricht, wird immer nur eine Fehlermeldung ausgegeben und führt zu keiner Änderung des Datensatzes. Nach der Ausgabe einer Fehlermeldung soll das Programm regulär auf die nächste Eingabe warten.

A.2 Befehle

Ihre interaktive Benutzerschnittstelle muss die folgenden Befehle gemäß der Spezifikationen umsetzen. Beachten Sie, dass im Fehlerfall eine Fehlermeldung ausgegeben wird, ansonsten erfolgt die Ausgabe gemäß der Spezifikation des Ausgabeformats. Im Folgenden sind alle Befehle – außer `add-admin` und `login-admin` – nur dann erlaubt, wenn gerade ein Admin eingeloggt ist. Der Befehl `quit` beendet in jedem Fall das Programm.

Im Folgenden wird die Ausgabe für einen Erfolgsfall spezifiziert. Im Fehlerfall (z.B. bei falsch spezifizierten Eingaben) muss eine aussagekräftige Fehlermeldung beginnend mit `Error,` ausgegeben werden.

Hinweise zu den Beispielen

Beachten Sie, dass bei den folgenden Beispielen die Eingabezeilen mit dem >-Zeichen, gefolgt von einem Leerzeichen, eingeleitet werden. Diese beiden Zeichen sind ausdrücklich kein Bestandteil des eingegebenen Befehls, sondern dienen nur der Unterscheidung zwischen Ein- und Ausgabe.

A.2.1 `add-admin`-Befehl

Der `add-admin`-Befehl fügt dem System einen neuen Nutzer hinzu, der Einträge verwalten und archivieren kann. Diese Operation kann nur durchgeführt werden, wenn kein Nutzer mit Hilfe des `login-admin`-Befehls angemeldet ist.

Eingabeformat:

```
add-admin <Vorname>;<Nachname>;<Benutzername>;<Passwort>
```

`<Vorname>` und `<Nachname>` beschreiben einen Verwaltungs- und Archivierungsnutzer und sind jeweils ein `String` ohne Zeilenumbruch und ohne Semikolon. `<Benutzername>` ist ein `String` ohne Zeilenumbruch und ohne Semikolon mit mindestens 4 und höchstens 8 Stellen zur eindeutigen Identifizierung eines Nutzers. `<Passwort>` ist ein `String` ohne Zeilenumbruch und ohne Semikolon mit mindestens 8 und höchstens 12 Stellen.

Ausgabeformat:

```
OK
```

Vorausgesetzt, dass ein neuer Nutzer erfolgreich dem System hinzugefügt wurde, wird `OK` ausgegeben. Im Fehlerfall (z.B. wenn bereits ein Nutzer mit dem angegebenen `<Benutzername>` existiert) wird eine aussagekräftige Fehlermeldung beginnend mit `Error,` ausgegeben.

A.2.2 `login-admin`-Befehl

Der `login-admin`-Befehl authentifiziert einen Nutzer des Verwaltungs- und Archivierungssystems und räumt ihm nach den hinterlegten Rechten die Möglichkeiten ein, bestimmte Befehle aufzurufen. Es kann immer nur ein Nutzer gleichzeitig angemeldet sein. Bevor sich ein neuer Nutzer erfolgreich mit dem `login-admin`-Befehl authentifiziert, muss sich der zuvor angemeldete Nutzer mittels `logout-admin`-Befehl abmelden.

Eingabeformat:

```
login-admin <Benutzername>;<Passwort>
```

`<Benutzername>` ist ein `String`, der den persönlichen Nutzernamen repräsentiert. `<Passwort>` ist ein `String`, der das persönliche Passwort des Nutzers repräsentiert. Wie beim Befehl `add-admin` sind die zuvor eingepflegten und eindeutigen `<Benutzername>` und `<Passwort>` Zeichenketten ohne Zeilenumbruch und ohne Semikolon.

Ausgabeformat:

```
OK
```

Im Fall einer erfolgreichen Authentifizierung eines registrierten Nutzers, wird `OK` ausgegeben. Im Fehlerfall (z.B. wenn ein Nutzer noch nicht im System registriert ist oder ein Nutzer bereits im System angemeldet ist) wird eine aussagekräftige Fehlermeldung beginnend mit `Error,` ausgegeben.

A.2.3 `logout-admin`-Befehl

Der `logout-admin`-Befehl meldet einen zuvor mit dem `login-in`-Befehl authentifizieren Nutzer des Verwaltungs- und Archivierungssystems ab und entzieht ihm bis zu seiner erneuten Authentifizierung die Möglichkeiten entsprechende Befehle, die im Nachfolgenden beschrieben werden, aufzurufen.

Eingabeformat:

```
logout-admin
```

Ausgabeformat:

```
OK
```

Falls ein zuvor authentifizierter Nutzer erfolgreich abgemeldet wurde, wird `OK` ausgegeben. Im Fehlerfall (z.B. wenn der abzumeldende Nutzer nicht vorher authentifiziert wurde mittels `login-admin`) wird eine aussagekräftige Fehlermeldung beginnend mit `Error,` ausgegeben.

A.2.4 `add-sports-venue`-Befehl

Dieser Befehl fügt dem System eine neue Sportstätte hinzu, die als Austragungsort für die Wettkämpfe sowie als Veranstaltungsort der Eröffnungs- und Abschlussfeier der Olympischen Spiele genutzt werden kann.

Eingabeformat:

```
add-sports-venue <ID>;<Ländername>;<Ort>;<Name>;<Eröffnungsjahr>;<Anzahl_Sitzplätze>
```

Die Sportstätte wird mithilfe einer eindeutigen dreistelligen `<ID>` in das Verwaltungs- und Archivierungssystem eingepflegt mit `<ID>` $\in \{001, 002, ..., 999\}$. Der Standort wird über den `<Ort>` und den `<Ländername>`, der einem IOC-Kürzel zugeordnet ist, bestimmt. Darüber hinaus besitzt jede Sportstätte einen Namen `<Name>` sowie eine vierstellige Angabe zum `<Eröffnungsjahr>` und wird durch die Zuschauerplätze `<Anzahl_Sitzplätze>` charakterisiert.

Ausgabeformat:

```
OK
```

`OK` lautet die Ausgabe zu diesem Befehl, wenn dem System erfolgreich eine Sportstätte hinzugefügt wurde. Im Fehlerfall (z.B. wenn die Sportstätte bereits mit der eingegeben `<ID>` existiert) wird eine aussagekräftige Fehlermeldung beginnend mit `Error,` ausgegeben.

A.2.5 `list-sports-venues`-Befehl

Für eine adäquate Planung der Olympischen Spiele ist es wichtig die Sportstätten im jeweiligen Land nach ihren Ressourcen zu sortieren.

Eingabeformat:

```
list-sports-venues <Ländername>
```

Ausgabeformat:

```
(<Platzierung>␣<ID>␣<Ort>␣<Anzahl_Sitzplätze>)
```

Die Ausgabe zu dem `list-sports-venues`-Befehl erfolgt nach einer Platzierung beginnend mit 1, indem nach der `<Anzahl_Sitzplätze>` einer Sportstätte aufsteigend sortiert wird. Besitzen die zu sortierenden Sportstätten eine gleiche Anzahl an Zuschauerplätzen, so wird aufsteigend nach ihrer `<ID>` sortiert. Bitte beachten Sie, dass die einzelnen Ausgaben in einer Zeile jeweils durch exakt ein Leerzeichen separiert sind. Im Fehlerfall (z.B. für den eingegebenen `<Ländername>` ist im System noch kein IOC-Kürzel festgelegt) muss eine aussagekräftige Fehlermeldung beginnend mit `Error,` ausgegeben werden.

A.2.6 `add-olympic-sport`-Befehl

Mit dem `add-olympic-sport`-Befehl werden durch den angemeldeten Nutzer Sportarten und Sportdisziplinen, an denen die Athleten in Wettkämpfen teilnehmen können, angelegt.

Eingabeformat:

```
add-olympic-sport <Sportart>;<Sportdisziplin>
```

Hierbei sind `<Sportart>` und `<Sportdisziplin>` jeweils eine Zeichenkette ohne Zeilenumbruch und ohne Semikolon. Eine Sportart kann sich aus mehreren Sportdisziplinen zusammensetzen. Es können auch Mannschaftssportarten bzw. die dazugehörigen Sportdisziplinen eingepflegt werden.

Ausgabeformat:

```
OK
```

Bei einem erfolgreichen Hinzufügen wird `OK` zu diesem Befehl ausgegeben. Im Fehlerfall (z.B. wenn die Zeichenketten `<Sportart>` und `<Sportdisziplin>` bereits hinzugefügt wurden) wird eine aussagekräftige Fehlermeldung beginnend mit `Error,` ausgegeben.

Beispielablauf mit sinnvollen Eingaben

```
> add-olympic-sport eishockey;eishockey
OK
> add-olympic-sport biathlon;biathlon
OK
> add-olympic-sport bobsport;bob
OK
> add-olympic-sport bobsport;skeleton
OK
> add-olympic-sport curling;curling
OK
> add-olympic-sport eislauf;eiskunstlauf
OK
> add-olympic-sport eislauf;eisschnelllauf
OK
```

A.2.7 `list-olympic-sports`-Befehl

Der `list-olympic-sports`-Befehl listet die Sportarten und -disziplinen alphabetisch auf.

Eingabeformat:

```
list-olympic-sports
```

Ausgabeformat:

```
<Sportart>␣<Sportdisziplin>
```

Die Ausgabe wird alphabetisch aufsteigend in erster Instanz nach der Sportart und bei Gleichheit in zweiter Instanz nach der Sportdisziplin sortiert. Bitte beachten Sie, dass die einzelnen Ausgaben in einer Zeile jeweils durch exakt ein Leerzeichen separiert sind.

Beispielablauf

```
> list-olympic-sports
biathlon biathlon
bobsport bob
bobsport skeleton
curling curling
eishockey eishockey
eislauf eiskunstlauf
eislauf eisschnelllauf
```

A.2.8 `add-ioc-code`-Befehl

Mit diesem Befehl wird ein IOC-Länder-Code (engl. International Olympic Committee IOC) angelegt. Die Eingabe `<IOC_Code>` stellt ein dreistelliges und eindeutiges Kleinbuchstabenländerkürzel dar. Die Eingabe `<Ländername>` repräsentiert den Namen des Landes als `String` ohne Zeilenumbruch und ohne Semikolon sowie `<Festlegungsjahr>` eine vierstellige Ziffer als positive `Integer`-Zahl, die das Jahr zur Festlegung der Kürzels angibt. Alle Einträge werden durch eine dreistellige `<IOC_ID>` mit `<IOC_ID>` $\in \{001, 002, ..., 999\}$ eindeutig gekennzeichnet. Gehen Sie davon aus, dass ein angelegter IOC-Code nicht mehr geändert werden kann bzw. muss. Gehen Sie davon aus, dass ein Land nur einen `IOC_Code` erhalten kann.
Eingabeformat:

```
add-ioc-code <IOC_ID>;<IOC_Code>;<Ländername>;<Festlegungsjahr>
```

Ausgabeformat:

```
OK
```

`OK` lautet die Ausgabe zu diesem Befehl. Im Fehlerfall (z.B. wenn die `<IOC_ID>` bzw. der `<IOC_Code>` bereits existieren) wird eine aussagekräftige Fehlermeldung beginnend mit `Error,` ausgegeben.

Beispielablauf

```
> add-ioc-code 111;arg;argentinien;1920
OK
> add-ioc-code 112;bhu;bhutan;1984
OK
> add-ioc-code 113;bul;bulgarien;1896
OK
> add-ioc-code 114;chi;chile;1896
OK
> add-ioc-code 115;cze;tschechien;1992
OK
> add-ioc-code 116;ecu;ecuador;1924
OK
> add-ioc-code 117;esp;spanien;1900
OK
> add-ioc-code 118;ger;deutschland;1992
OK
> add-ioc-code 119;can;kanada;1900
OK
```

A.2.9 `list-ioc-codes`-Befehl

Der Befehl `list-ioc-codes` gibt u. a. alle Länder mit Ihren Ländercodes aus.

Eingabeformat:

```
list-ioc-codes
```

Ausgabeformat:

```
<Festlegungsjahr>␣<IOC_ID>␣<IOC_Code>␣<Ländername>
```

Die ausgegebene Liste wird aufsteigend nach dem `<Festlegungsjahr>` sortiert. Für den Fall, dass zwei oder mehrere Länder dasselbe Festlegungsjahr besitzen, wird aufsteigend nach ihrer `<IOC_ID>` sortiert.
Bitte beachten Sie, dass die einzelnen Ausgaben in einer Zeile jeweils durch exakt ein Leerzeichen separiert sind.

Beispielablauf

```
> list-ioc-codes
1896 113 bul bulgarien
1896 114 chi chile
1900 117 esp spanien
1900 119 can kanada
1920 111 arg argentinien
1924 116 ecu ecuador
1984 112 bhu bhutan
1992 115 cze tschechien
1992 118 ger deutschland
```

A.2.10 `add-athlete`-Befehl

Der `add-athlete`-Befehl fügt dem Verwaltungs- und Archivierungssystem ein Profil eines Athleten hinzu.

Eingabeformat:

```
add-athlete <ID>;<Vorname>;<Nachname>;<Ländername>;<Sportart>;<Sportdisziplin>
```

Hierbei wird für jeden Athleten eine eindeutige vierstellige `<ID>` mit `<ID>` $\in \{0001, 0002, ..., 9999\}$ angelegt. Der Athlet selbst ist durch seinen `<Vorname>` sowie `<Nachname>` und sein Herkunftsland mit `<Ländername>` als Zeichenketten ohne Zeilenumbruch und ohne Semikolon charakterisiert. Der `<Ländername>` muss einem IOC-Kürzel zugehörig sein. Weiterhin ist die `<Sportart>` und die dazugehörige `<Sportdisziplin>`, mit der er an den Olympischen Spielen teilnimmt, Bestandteil seines Profils. Prinzipiell kann ein Athlet auch an mehreren Sportarten und Sportdisziplinen teilnehmen.

Ausgabeformat:

```
OK
```

Beim erfolgreichen Hinzufügen eines Athletenprofils wird `OK` ausgegeben. Im Fehlerfall (z.B. wenn die `<Sportart>` bzw. `<Sportdisziplin>` im System noch nicht existiert) wird eine aussagekräftige Fehlermeldung beginnend mit `Error,` ausgegeben.

Beispielablauf

```
> add-athlete 0001;max;mustermann;deutschland;bobsport;bob
OK
> add-athlete 0001;max;mustermann;deutschland;eislauf;eisschnelllauf
OK
> add-athlete 0002;jane;doe;kanada;eislauf;eiskunstlauf
OK
> add-athlete 0002;jane;doe;kanada;eislauf;eisschnelllauf
OK
```

A.2.11 `summary-athletes`-Befehl

Der Befehl `summary-athletes` listet die Teilnehmer der Olympischen Winterspiele anhand ihrer sportlichen Erfolge in der jeweiligen `<Sportart>` und `<Sportdisziplin>` auf.

Eingabeformat:

```
summary-athletes <Sportart>;<Sportdisziplin>
```

Ausgabeformat:

```
<ID>␣<Vorname>␣<Nachname>␣<Anzahl_Medaillen>
```

Die Athleten werden anhand ihrer gewonnen Anzahl an Medaillen als `Integer`-Zahl m mit $m \geq 0$ absteigend sortiert. Jede Medaille ist dabei gleich viel wert. Besitzen zwei oder mehrere Athleten die gleiche Anzahl an Medaillen, so wird die Ausgabe anhand ihrer vierstelligen `<ID>` aufsteigend sortiert. Bitte beachten Sie, dass die einzelnen Ausgaben in einer Zeile jeweils durch exakt ein Leerzeichen separiert sind.

A.2.12 `add-competition`-Befehl

Der `add-competition`-Befehl fügt dem System das Ergebnis eines Wettkampfes eines teilnehmenden Sportlers hinzu.

Eingabeformat:

```
add-competition <ID>;<Jahr>;<Ländername>;<Sportart>;<Sportdisziplin>;<Gold>;<Silber>;<Bronze>
```

Die erste Eingabe repräsentiert die eindeutige vierstellige `ID` (siehe `add-athlete`-Befehl) des Olympiateilnehmers. Das vierstellige `<Jahr>` steht für das Jahr, indem der Sportler mit einem Herkunftsland als `<Ländername>` an seiner `<Sportart>` und `<Sportdisziplin>` teilgenommen hat. Die gültige Eingabe des Jahres sind 1926-2018 im vierjährigen Turnus. Abschließend wird der Leistungserfolg des betrachteten Sportlers dokumentiert. Für den Fall, dass er eine Goldmedaille in dem zuvor spezifizierten Wettkampf gewonnen hat, wird `1;0;0` eingegeben. Eine Silbermedaille wird durch `0;1;0` repräsentiert. Die Eingabe `0;0;1` stellt eine gewonnene Bronzemedaille dar. Gewinnt der Athlet keine Medaille wird `0;0;0` eingegeben. Bemerkung zu den Eingaben: Ein Sportler kann nicht im gleichen Jahr in derselben Sportart/-disziplin mehrere Medaillen gewinnen. Zwei oder mehrere Sportler können in der gleichen Sportart/-disziplin im gleichen Jahr die gleiche Platzierung haben. Damit sind in diesem spezifizierten Fall Mehrfach-Gold/-Silber/-Bronze-Medaillen erlaubt.

Ausgabeformat:

```
OK
```

`OK` lautet die Ausgabe zu diesem Befehl. Im Fehlerfall (z.B. bei falsch spezifizierten Eingaben, ungültigen Eingaben bezüglich der Platzierung, die eingegebene `<Sportart>` und `<Sportdisziplin>` existiert noch nicht im System usw.) wird eine aussagekräftige Fehlermeldung beginnend mit `Error,` ausgegeben.

A.2.13 `olympic-medal-table`-Befehl

Für sämtliche Wettkämpfe der Olympischen Winterspielen wird eine Rangliste in Form eines ewigen Medaillenspiegels erstellt.
Eingabeformat:

```
olympic-medal-table
```

Ausgabeformat:

```
(<Platzierung>,<IOC_ID>,<IOC_Code>,<Ländername>,<Gold>,<Silber>,<Bronze>,<Gesamtanzahl_Medaillen>)
```

Die Ausgabe enthält eine `<Platzierung>` als `Integer`-Zahl beginnend mit 1. Sortiert wird die Ausgabe absteigend nach der Anzahl der Goldmedaillen, dann der Silbermedaillen und zuletzt nach der Anzahl der Bronzemedaillen. Besitzen mehrere IOC-Länder exakt die gleiche Medaillenverteilung, so werden die Länder nach ihrer `IOC_ID` aufsteigend sortiert.

A.2.14 `reset`-Befehl

Der Befehl `reset` initialisiert die Olympischen Spiele neu. Hinweis: Nach einem `reset`-Befehl bleiben alle registrierten Nutzer erhalten. Der aktuell angemeldete Nutzer wird dadurch nicht abgemeldet.
Eingabeformat:

```
reset
```

Ausgabeformat:

```
OK
```

`OK` lautet die Ausgabe zu diesem Befehl.

Beispielablauf

```
> reset
OK
```

A.2.15 `quit` -Befehl

Dieser Befehl beendet das Programm. Dabei findet keine Konsolenausgabe statt.
Hinweis: Denken Sie daran, dass hierfür keine Methoden wie `System.exit` oder `Runtime.exit` verwendet werden dürfen.

Beispielablauf

```
> quit
```