Zur Leitseite Brecht
Zur Leitseite dieser Veranstaltung

Verteilte Systeme

Übung 1

Abgabe bis spätestens 11.11.14
Beachten Sie bitte die Hinweise zu den Testaten. Sie sind Bestandteil der Aufgabenstellung.

Arbeiten mit Java-Threads (Nebenläufigkeit)

Grobstruktur

Ein Telefonverzeichnis soll nebenläufig abgefragt werden können. Anschaulich ist dieses Verzeichnis eine zweispaltige Tabelle der folgenden Form:

```
Name     Tel.-Nummer
====     ==========
Meier    4711
Schmitt  0815
Müller   4711
Meier    0816
```

Die Feldwerte sind keineswegs eindeutig. Das heißt, dass zu einem Namen mehrere Telefonnummern und zu einer Telefonnummer mehrere Namen gehören können. Das Abfragen des Telefonverzeichnisses ist so zu gestalten, dass der Benutzer
* entweder nur einen Namen
* oder nur eine Nummer
* oder einen Namen und eine Nummer zusammen eingeben kann.

Gibt der Benutzer nur einen Namen aber keine Nummer ein, wie zum Beispiel "Meier", dann erhält er alle dazu passenden Zeilen, also im Beispiel
Meier 4711
Meier 0816

Das Gleiche geschieht, wenn er nur eine Nummer und keinen Namen eingibt. Sucht er z.B. nach "4711", dann bekommt er folgende Zeilen geliefert:
Meier 4711
Müller 4711

Gibt er jedoch einen Namen und eine Telefonnummer (zusammen) ein, dann wird nebenläufig, also unabhängig voneinander, sowohl nach dem Namen als auch nach der Telefonnummer gesucht. Sucht der Benutzer beispielsweise nach der Kombination "Meier" und "4711", dann erhält er im obigen Beispiel folgende Trefferzeilen, wobei wegen der Nebenläufigkeit die Reihenfolge nicht vorhersagbar ist:
Meier 4711 (Aus der Suche nach "Meier")
Meier 0816 (Aus der Suche nach "Meier")
Meier 4711 (Aus der Suche nach "4711")
Müller 4711 (Aus der Suche nach "4711")

Aufgabe

Es ist ein Java-Programm zu erstellen, das einen Telefonserver realisiert, der die in der Grobstruktur beschriebenen Dialoge ermöglicht. Der Server bietet dem Benutzer in einer Endloschleife (es ist kein Einmal-Server!) an,

* das Programm zu beenden oder
* einen Namen oder
* eine Telefonnummer oder
* einen Namen und eine Telefonnummer zusammen anzugeben.

Gibt der Benutzer nur einen Namen an, erzeugt der Server neben dem main-Thread einen (und nur einen) zusätzlichen Thread, der in dem Verzeichnis nach dem Namen sucht. Gibt er nur eine Nummer an, wird neben dem main-Thread genau ein Thread erzeugt, der nach der Nummer sucht. Werden schließlich sowohl ein Name als auch eine Nummer eingegeben, werden zwei zusätzliche Such-Threads erzeugt, von denen nebenläufig einer nach dem Namen und der andere nach der Nummer sucht.

Konkretisierung

1. Wie Sie das Telefonverzeichnis realisieren, bleibt Ihnen überlassen. Sie können eine externe Lösung mit einer Datei vorsehen oder eine ausschließlich interne, zum Beispiel mit einem Stringarray.
2. Lassen Sie bei den Namen Umlaute wie ä und ü zu und beachten Sie, dass es Namen wie "von Reibach" mit Wortlücken geben kann. Für das Testat müssen im Telefonverzeichnis wenigstens ein Name und eine Nummer mehrfach vorkommen. Wenigstens ein Name muss einen Umlaut enthalten und wenigstens einer eine Wortlücke. Achten Sie auf die korrekte Ausgabe der Umlaute.
3. Es gehört nicht zur Aufgabenstellung, das Telefonverzeichnis per Programm aufzubauen und zu pflegen. Beschränken Sie sich beim Programmieren auf das reine Abfragen.
4. Die Suchthreads sind so zu gestalten, dass sie jeweils nur in ihrem Bereich suchen. Das heißt, dass eine Suche mit beispielsweise der Telefonnummer "Meier" scheitern muss.
5. Geben Sie im Falle des Scheiterns einer Suche immer eine Negativmeldung im Sinne von ***Die Suche nach xxx war erfolglos*** aus.
6. Es genügt, die Benutzerschnittstelle einfach zu halten. Wenn Sie wollen, können Sie eine grafische Schnittstelle erstellen, notwendig ist es nicht. Es reicht, wenn der Benutzer mit einem Text abgefragt werden kann.
7. Weisen Sie leere Eingaben und reine Blank- oder Tabfolgen sofort mit einer entsprechenden Meldung zurück. Gehen Sie damit nicht durch das Telefonverzeichnis.
8. Bei der nebenläufigen Suche kann eine Zeile der Tabelle von beiden Threads gefunden werden. Sie wird dann zweimal ausgegeben, was im Testat auch so erwartet und überprüft wird.

Programmierhinweise

Die Nebenläufigkeit Ihres Programms muss klar erkennbar sein. Betreiben Sie keine *verschleiernde Programmierung*. Ihr Programm muss, und das wird beim Testat geprüft, eine Stelle folgender Art enthalten:

```
suchThr1.start();
suchThr2.start();
suchThr1.join();
suchThr2.join();
```

Lassen Sie bitte nicht die beiden Such-Threads, sondern den main-Thread, die Suchergebnisse ausgeben. Das darf jedoch erst geschehen, wenn beide Such-Threads beendet sind. Lassen Sie die Such-Threads je eine dynamische Datenstruktur, z.B. einen String oder einen Stack aus dem java.util-Paket, füllen und main diese Strukturen auslesen und ausgeben.