Aufgabe 1: Superstar

Team: Pied Piper  
Team-ID: 00342

27. September 2018

# Inhalt

[Inhalt 1](#_Toc525843949)

[Lösungsidee 1](#_Toc525843950)

[Laufzeitoptimierung 1](#_Toc525843951)

[Umsetzung 2](#_Toc525843952)

[Beispiele 3](#_Toc525843953)

[Quellcode 3](#_Toc525843954)

# Lösungsidee

Die Aufgabe impliziert im Prinzip folgende zwei Bedingungen, die für ein Mitglied gelten muss, um als „Superstar“ gewertet zu werden:

1. Der potenzielle Superstar X folgt keinem Mitglied der Gruppe (im folgenden Bedingung 1)
2. Der potenzielle Superstar X, wird von jedem anderen Mitglied der Gruppe gefolgt (im folgenden Bedingung 2)

Diese zwei Bedingungen sind mit „Anfragen“ herauszufinden. Die Anfragenanzahl entspricht der benötigten Anzahl an Anfragen zum Erkennen eines Superstars. Eine Anfrage entspricht der Antwort, ob das Mitglied Y, dem Mitglied Z folgt. Grundlegend kann man einfach linear durch alle Namen durchgehen, und überprüfen ob die Bedingungen, für einen vorher bestimmten potenziellen Superstar X, zutreffen. Findet man ein Mitglied, für das beide Bedingungen zustimmen, wird das Mitglied als Superstar deklariert und die Suche wird abgebrochen, da es nur ein Mitglied geben kann, für das beide Bedingungen gelten.

## Laufzeitoptimierung

Die Reihenfolge der Abfrage der Bedingungen ist zur Minimierung der Anfragen, um die Überprüfung ggf. früher abzubrechen und somit keine unnötigen Anfragen zu stellen, nicht relevant, da bei beiden Bedingungen der potenzielle Superstar X mit jedem anderen Mitglied innerhalb der Gruppe in Verbindung gebracht werden muss. Wenn z.B. eine Gruppe Z mit den Mitgliedern A, B und C vorhanden ist, macht es keinen Unterschied ob zuerst überprüft wird ob der potenzielle Superstar A, den Mitgliedern B und C nicht folgt oder ob der potenzielle Superstar A von B und C gefolgt wird, es sind im Endeffekt gleich viele Anfragen. Ich habe mich für die Reihenfolge entschieden, bei der zuerst die Bedingung 2 und dann die Bedingung 1 überprüft wird, aus folgendem Grund entschieden. Denn relevant für die Anfragenanzahl sind die schon vorher gestellten Anfragen, sodass man dadurch folgende potenzielle Superstars ausschließen kann. Ist man dabei Bedingung 2 zu überprüfen, kann man durch diese schon getätigten Anfragen, Mitglieder ausschließen, wenn hier schon bekannt wird, dass dieses Mitglied einem anderen Mitglied folgt und somit direkt die erste Bedingung ausgeschlossen wird. Wenn wir wieder unsere Gruppe Z nehmen und schauen, ob Mitglied B der Superstar ist und durch aktuelles Überprüfen der zweiten Bedingung herausfinden, dass Mitglied C, dem Mitglied B folgt, braucht man gar nicht das Mitglied C überprüfen, da schon die erste Bedingung bei Mitglied C ausgeschlossen wurde. So kann man die zweite Bedingung vorschieben, da diese Anfragen schon nächste potenzielle Superstars ausschließen könnte, wenn mitten in der aktuellen Überprüfung klar wird, dass das momentan überprüfte Mitglied kein Superstar mehr sein kann. In der anderen Anordnung der Reihenfolge der Abfrage, hat man keinen Sinn aus den Anfragen mehr bekommen, wenn dort mitten in der Überprüfung, die Überprüfung abgebrochen wird. Mit dieser Anordnung der Bedingungen kann man dann jedes Mitglied nehmen und die Bedingungen, wie am Anfang schon erwähnt, überprüfen.

# Umsetzung

Die Lösungsidee wurde in Java implementiert. Mithilfe der java.nio Bibliothek wird die Datei mit den Mitgliedern in ein Array gespeichert und die Teilnehmeranzahl ermittelt, sowie die einzelnen Teilnehmernamen. Diese werden dann gespeichert und bilden somit den Grundbaustein des Vorgangs (im Programm die splitFile Methode).

Um überhaupt die Bedingungen überprüfen zu können, müssen entsprechende Anfragen gestellt werden, sprich ob Mitglied A dem Mitglied B folgt. Dies wird dadurch bewerkstelligt, dass die zu überprüfenden Mitglieder als Paramater an eine Methode, sowie das Array, das die Textdatei enthält, an eine Methode übergeben werde. Zu allererst wird in der Methode eine globale Variable um 1 erhöht, die somit anzeigt, wie viele Anfragen bisher gestellt wurden. Anschließend geht die Methode dann einfach mithilfe einer for-Schleife durch das Array und schaut in jeder Zeile ob die Zeile dem String: Mitglied A plus einem Leerzeichen plus Mitglied B gleicht. Falls durch die ganze Datei gegangen ist, und keine solche Zeile gefunden worden ist, wird *false* zurückgegeben, das Mitglied A folgt also nicht dem Mitglied B. Falls eine Zeile gefunden wurde, dass dem angegebenen String gleicht, wird *true* zurückgegeben, das Mitglied A folgt also dem Mitglied B.

Insgesamt sind 3 verschiedene for-Schleifen für das Finden des Superstars verbaut. Eine zeigt das aktuell zu überprüfende Mitglied an, während die anderen beiden jeweils die Bedingungen für das in der ersten for-Schleife ausgewählte Mitglied überprüfen. Die Reihenfolge des Vorgangs ist von der vorhandenen Liste aller Mitglieder abhängig.

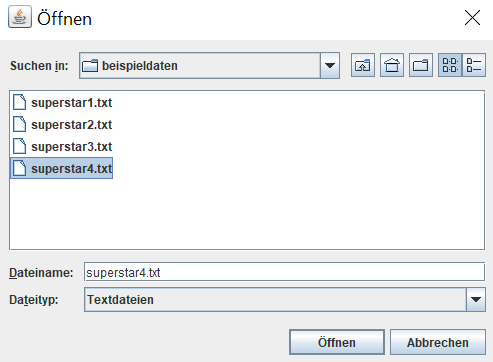
Ist man nun in der zweiten for-Schleife angelangt wird zuerst die Bedingung 2 überprüft. Diese Schleife läuft solange bis entweder rauskommt, dass ein Mitglied nicht dem potenziellen Superstar folgt, also die Anfrage ob Mitglied Y dem potenziellen Superstar X folgt ist *false*, oder der Vorgang läuft bis zum Ende und die Variable die Aussagt. Ist bekannt, dass ein Mitglied, X nicht wird der Vorgang vorzeitig abgebrochen, die Variable *getsFollowed*, die Aussagt ob X von allen Mitgliedern gefolgt wird, wird auf *false* gesetzt und der nächste potenzielle Superstar X wird von der ersten for-Schleife ausgewählt. Läuft der Vorgang erfolgreich bis zum Ende, ohne abzubrechen, sprich die Bedingung wurde mit jedem Mitglied und X überprüft, bleibt *getsFollowed* auf *true*. Außerdem wird nach jedem erfolgreichen Durchlauf dieser Schleife, das dem potenziellen Superstar folgende Mitglied auf einer Liste gespeichert, falls es nicht schon auf der Liste ist, um folgende Iterationen zu überspringen. Ist *getsFollowed* nach der Schleife *false* wird sofort der nächste potenzielle Superstar von der ersten for-Schleife gewählt. Sonst wird die nächste Bedingung, die Bedingung 1, mit der dritten for-Schleife überprüft.

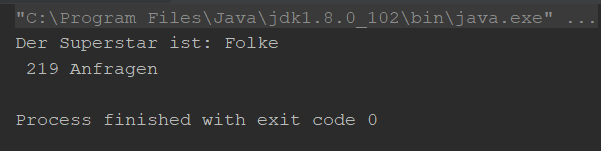
Diese Bedingung wird fast genauso wie in der ersten Schleife geprüft. Die Schleife läuft solange, bis erkannt wird, dass X einem Mitglied folgt, also bis die Anfrage *true* ist oder bis die Bedingung mit jedem Mitglied und X erfolgreich überprüft worden ist. Folgt X einem Mitglied, wird der Vorgang abgebrochen und die Variable, die aussagt, ob X keinem Mitglied folgt *doesntFollow* wird auf *false* gesetzt. Wurde erfolgreich überprüft, bleibt die Variable auf *true*.

Wurden die beiden Variablen bzw. Bedingungen nicht verändert, sind sie beide auf *true*. Somit müssen beide Bedingungen bei dem aktuellen potenziellen Superstar gelten. Die Suche wird erfolgreich abgebrochen, da es nur einen Superstar geben kann. Der Superstar wird dann an die main-Methode zurückgegeben, danach wird der Superstar sowie die benötigten Anfragen ausgegeben und das Programm wird beendet. Wurde die Suche erfolglos abgebrochen, werden die beiden Variablen wieder zurück auf *true* gesetzt und die Schleife beginnt wieder mit dem nächsten potenziellen Superstar.

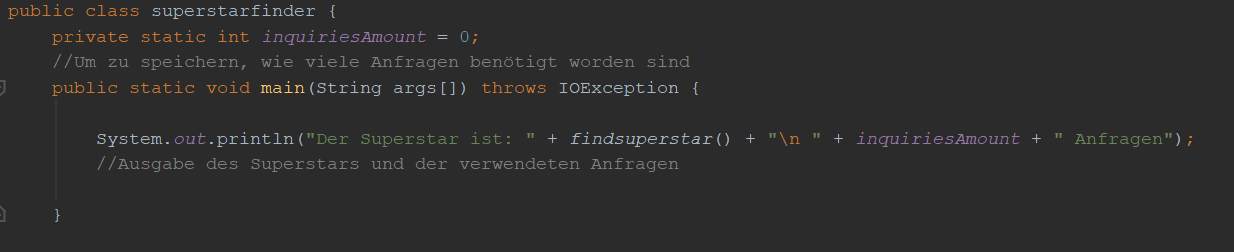
Wird am Anfang einer Schleife erkannt, dass der aktuelle potenzielle Superstar ein Mitglied der Liste ist, die vorher schon durch das Überprüfen der Bedingung 2 in der zweiten for-Schleife, ausgeschlossen wurden, wird die komplette Iteration übersprungen und der nächste potenzielle Superstar ausgewählt.

# Beispiele

Das Programm wird aufgerufen und die zu überprüfende Datei wird ausgewählt.

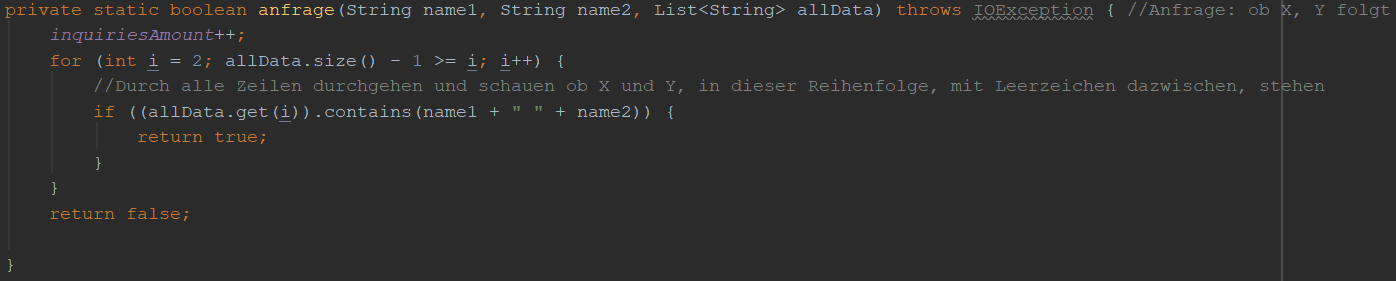
Anschließend wird in der Konsole das Ergebnis präsentiert.

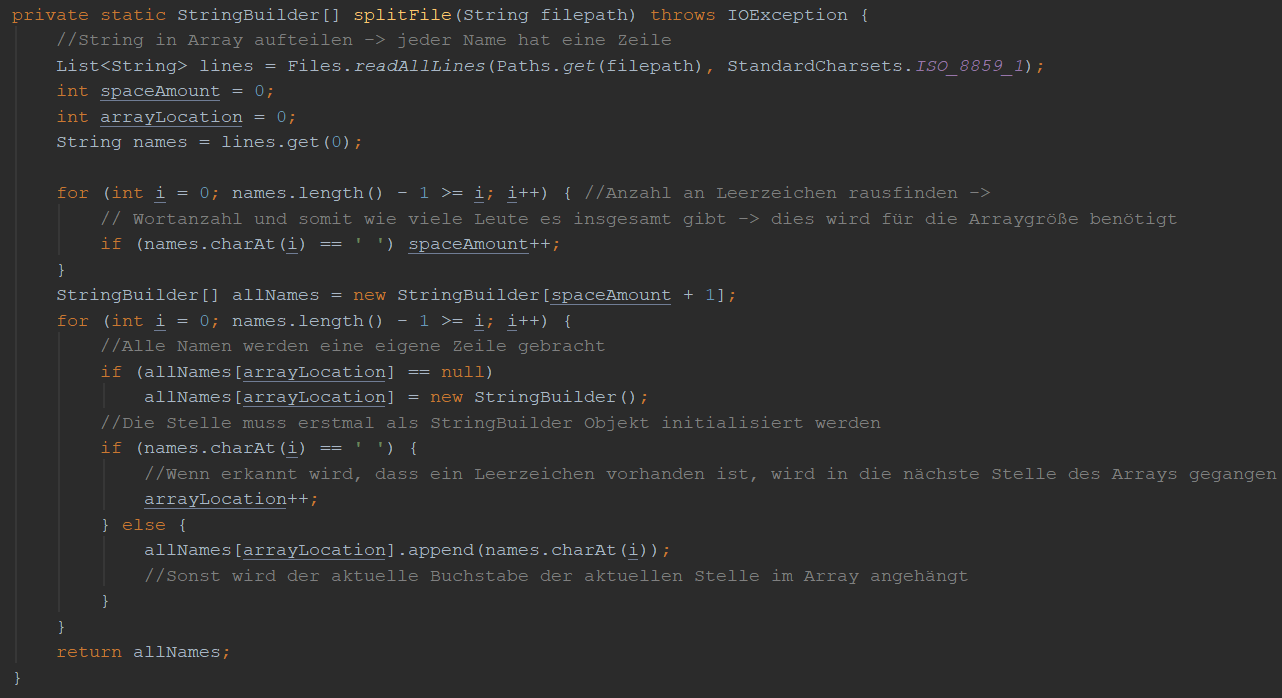
# Quellcode

Die globale Variable und die main-Methode

Die Methode zum Finden des Superstars

Die Methode zum Anfragen stellen



Die Methode zum Entnehmen aller Mitglieder

Die Methode zum Auswählen der Datei

