

Dokumentation zur Aufgabe „Superstar“

Die Lösungsidee:

Ein Superstar darf niemanden aus einer Gruppe Folgen und muss zeitgleich von allen anderen Mitgliedern dieser Gruppe gefolgt werden, um den Titel Superstar zu erhalten. Daraus folgt folgende Tatsache: Wenn eine Anfrage gestellt wird, um zu gucken ob eine Person X einer Person Y folgt, so kann man stets für eine Person sagen, ob sie ein Superstar sein kann oder nicht. Folgt Person X Person Y, so kann X kein Superstar sein, da X einer anderen Person in der Gruppe folgt. Folgt Person X hingegen nicht Person Y, so kann Person Y kein Superstar sein, weil nicht alle anderen Mitglieder der Gruppe Person Y folgen. Auf diesen Weg kann man mit einer Anzahl von *Gruppenmitglieder - 1* Anfragen heraus finden, welche Person als einzige ein Superstar sein könnte. Da aber in einer Gruppe nicht immer ein Superstar sein muss, muss man anschließend kontrollieren, ob die zuvor übrig gebliebene Person niemandem folgt und von allen anderen in der Gruppe gefolgt wird. Dies kostet folglich bis zu $2 * (\text{Gruppenmitglieder} - 1)$ Anfragen. Das „bis zu“ kommt daher, dass zwar für alle übrigen Mitglieder einer Gruppe zwei Anfragen gestellt werden müssen - also jeweils: folgt Person X dem möglichen Superstar und folgt der mögliche Superstar Person X -, manche dieser Anfragen aber bereits im vorherigen Ausschließungsverfahren gestellt worden sein können, weshalb diese Anfragen nicht erneut gestellt werden müssen.

Die Umsetzung mit Quelltext:

Die Umsetzung der Lösungsidee erfolgt in meiner Bearbeitung in der Programmiersprache Ruby. Das Programm beziehungsweise Script („superstar.rb“) liegt im Ordner „Aufgabe 1“ und stellt zeitgleich Quelltext und ausführbare Datei dar. Ich habe das Script mit der Ruby Version 2.5.3-1-x64 in der CMD von Windows 10 getestet.

```
puts "Gebe den Namen der .txt-Datei einer Gruppe an (z.B. superstar4.txt)"
file = gets.chomp
$list = File.open(file).read.force_encoding("UTF-8").split("\n")
```

Als erstes wird eine eingegebene Gruppen-Liste eingelesen und in den globalen Array \$List so aufgeteilt, dass in diesem jede Zeile einzeln liegt. Dies ist notwendig um mit der folgenden Methode auf einfachem Weg herausfinden zu können wer wem folgt:

```
def isYfollowingZ(y, z)
  i = 1
  follows = false
  while i < $list.length do
    firstName = $list[i].split(" ")[0]
    secondName = $list[i].split(" ")[1]
    if firstName == y and secondName == z
      follows = true
    end
    i += 1
  end
  return follows
end
```

Die Methode geht jeden Eintrag in der Gruppen-Liste durch und gibt abhängig vom Ergebnis true oder false zurück.

```
requests = 0
membersOfGroup = Array.new
membersOfGroup = $list[0].split(" ")
puts "Die Gruppe besteht aus #{membersOfGroup.length} Mitgliedern.\n\n"
possibleSuperstars = Array.new
possibleSuperstars = membersOfGroup.dup
followingList = Array.new(membersOfGroup.length)
i = 0
while i < membersOfGroup.length
  followingList[i] = Array.new(membersOfGroup.length, nil)
  i += 1
end
```

Im nächsten Schritt wird ausgegeben, wie viel Mitglieder die Gruppe hat. Zudem werden zwei Arrays erstellt: *possibleSuperstars* enthält nach jeder Anfrage später die noch verbleibenden möglichen Superstars. Der Array *followingList* beinhaltet am Ende sämtliche Ergebnisse der gestellten Anfragen: es handelt sich um einen zweidimensionalen Array, welcher vollständig mit nichts - also *nil* - gefüllt ist.

Im folgenden Abschnitt wird dieser Array wie folgt befüllt: Folgt Person X der Person Y so wird an der Position [Position von Person X in der Gruppe][Position von Person Y in der Gruppe] der Wert des Arrays auf true gesetzt. Folgt Person X der Person Y nicht wird der Wert des Arrays an dieser Position auf false gesetzt.

```

while possibleSuperstars.length > 1 do
  if isYfollowingZ(possibleSuperstars[0], possibleSuperstars[1])
    followingList[membersOfGroup.index(possibleSuperstars[0])]

  [membersOfGroup.index(possibleSuperstars[1])] = true
    puts "Anfrage: Folgt #{possibleSuperstars[0]} #{possibleSuperstars[1]}?\\nAntwort: True\\n\\n"
    possibleSuperstars.delete_at(0)
    possibleSuperstars.push(possibleSuperstars.shift)

  else
    followingList[membersOfGroup.index(possibleSuperstars[0])]

  [membersOfGroup.index(possibleSuperstars[1])] = false
    puts "Anfrage: Folgt #{possibleSuperstars[0]} #{possibleSuperstars[1]}?\\nAntwort: False\\n\\n"
    possibleSuperstars.push(possibleSuperstars.shift)
    possibleSuperstars.delete_at(0)

  end
  requests += 1
end

```

Hier findet das erste Ausschließungsverfahren statt: Solange es mehr als einen möglichen Superstar gibt, werden die zwei ersten Namen aus der Liste der möglichen Superstars genommen. Es wird geguckt, ob die erste dieser beiden Personen der zweiten folgt. Ist dies der Fall wird die erste Person aus der Liste entfernt, da diese einer anderen Person der Gruppe folgt und somit kein Superstar sein kann. Folgt die erste Person der zweiten nicht, so wird die zweite Person aus der Liste entfernt, weil sie nicht von allen anderen Gruppenmitgliedern gefolgt wird und folglich kein Superstar sein kann. Die übrig gebliebene Person wird nach hinten wieder in die Liste angehängt, um nicht mehrere Anfragen für eine Person hintereinander zu stellen. Ohne diesen Vorgang könnte es vorkommen, dass für eine Person, die letztlich gar kein Superstar ist, eine große Anzahl an Anfragen verschickt werden.

```

puts "Lediglich #{possibleSuperstars[0]} könnte der Superstar der Gruppe sein. Bisher wurden
#{requests} Anfragen gestellt. Es ist möglich, dass die Gruppe keinen Superstar hat. Soll
kontrolliert werden, ob #{possibleSuperstars[0]} wirklich ein Superstar ist? (ja / beliebige andere
Eingabe)"
x = gets.chomp.downcase
if x == "ja"
  indexOfPossibleStar = membersOfGroup.index(possibleSuperstars[0])

```

Da es denkbar ist, dass die Werbetreibenden aus der Aufgabenstellung für ihre Zwecke nicht kontrollieren müssen, ob der mögliche Superstar wirklich einer ist, wird hier gefragt, ob man kontrollieren möchte, ob der mögliche Superstar wirklich ein Superstar ist.
 Zum Beispiel wäre es schließlich denkbar, dass Werbetreibende einem Superstar keine Werbung zeigen wollen, wobei es nicht sonderlich schlimm wäre, wenn eine einzelne Person in einer Gruppe keine Werbung bekommt, obwohl diese kein Superstar ist. Die Ersparnisse bei der Findung des (falschen) Superstars würden hier schließlich den Verlust durch nicht gezeigte Werbung ausgleichen.

```

canBeAStar = true
i = 0
while i < membersOfGroup.length and canBeAStar do
  if not membersOfGroup[i] == possibleSuperstars[0]
    if followingList[indexOfPossibleStar][i] == nil
      if isYfollowingZ(possibleSuperstars[0], membersOfGroup[i])
        canBeAStar = false
        puts "Anfrage: Folgt #{possibleSuperstars[0]} #{membersOfGroup[i]}?\nAntwort:
True\n\n"
      else
        puts "Anfrage: Folgt #{possibleSuperstars[0]} #{membersOfGroup[i]}?\nAntwort:
False\n\n"
    end
    requests += 1
  end
  if followingList[i][indexOfPossibleStar] == nil
    if not isYfollowingZ(membersOfGroup[i], possibleSuperstars[0])
      canBeAStar = false
      puts "Anfrage: Folgt #{membersOfGroup[i]} #{possibleSuperstars[0]}?\nAntwort:
False\n\n"
    else
      puts "Anfrage: Folgt #{membersOfGroup[i]} #{possibleSuperstars[0]}?\nAntwort:
True\n\n"
    end
    requests += 1
  end
  i += 1
end

```

In diesem Codeabschnitt wird folglich nun kontrolliert, ob der übrig gebliebene, mögliche Superstar tatsächlich ein Superstar ist. Um dies zu erreichen, wird für jede Person in der jeweiligen Gruppe geguckt, ob diese dem möglichen Superstar folgt und ob diese vom möglichen Superstar gefolgt wird. Dabei werden Anfragen, die bereits gestellt wurden ausgelassen. Dies geschieht dadurch, dass mit dem Array followingList kontrolliert wird, ob eine gleiche Anfrage bereits im vorherigen Ausschlussverfahren stattgefunden hat. In einem solchen Fall wird die Anfrage übersprungen.

Im Falle das festgestellt wird, dass der mögliche Superstar kein Superstar ist, wird die Kontrolle (Schleife) vorzeitig abgebrochen.

```

if canBeAStar
  puts "#{possibleSuperstars[0]} ist mit Gewissheit ein Superstar! Es wurden insgesamt
#{requests} Anfragen benötigt, um dies herauszufinden."
else
  puts "#{possibleSuperstars[0]} ist kein Superstar. Die Gruppe scheint keinen Superstar zu
haben. Es wurden #{requests} Anfragen gestellt."
end
else
  puts "Superstar Suche beendet ..."
end

```

Zum Schluss folgt dann die jeweilige Ausgabe der Ergebnisse, zu denen das Programm gekommen ist.

Beispiel (Demonstration):

```
cmd: Eingabeaufforderung
C:\Users\HP-2018\Superstar>superstar
Gebe den Namen der .txt-Datei einer Gruppe an (z.B. superstar4.txt)
superstar1.txt
Die Gruppe besteht aus 3 Mitgliedern.
Anfrage: Folgt Selena Justin?
Antwort: True
Anfrage: Folgt Hailey Justin?
Antwort: True
Lediglich Justin könnte der Superstar der Gruppe sein. Bisher wurden 2 Anfragen gestellt. Es ist möglich, dass die Gruppe keinen Superstar hat. Soll kontrolliert werden, ob Justin wirklich ein Superstar ist? (ja / beliebige andere Eingabe)
ja
Anfrage: Folgt Justin Selena?
Antwort: False
Anfrage: Folgt Justin Hailey?
Antwort: False
Justin ist mit Gewissheit ein Superstar! Es wurden insgesamt 4 Anfragen benötigt, um dies herauszufinden.

cmd: Eingabeaufforderung
C:\Users\HP-2018\Superstar>superstar
Gebe den Namen der .txt-Datei einer Gruppe an (z.B. superstar4.txt)
superstar2.txt
Die Gruppe besteht aus 5 Mitgliedern.
Anfrage: Folgt Turing Hoare?
Antwort: True
Anfrage: Folgt Dijsktra Knuth?
Antwort: False
Anfrage: Folgt Codd Hoare?
Antwort: False
Anfrage: Folgt Dijkstra Codd?
Antwort: False
Lediglich Dijkstra könnte der Superstar der Gruppe sein. Bisher wurden 4 Anfragen gestellt. Es ist möglich, dass die Gruppe keinen Superstar hat. Soll kontrolliert werden, ob Dijkstra wirklich ein Superstar ist? (ja / beliebige andere Eingabe)
ja
Anfrage: Folgt Dijkstra Turing?
Antwort: False
Anfrage: Folgt Turing Hoare?
Antwort: True
Anfrage: Folgt Hoare Dijkstra?
Antwort: True
Anfrage: Folgt Knuth Dijkstra?
Antwort: True
Anfrage: Folgt Codd Dijkstra?
Antwort: True
Dijkstra ist mit Gewissheit ein Superstar! Es wurden insgesamt 10 Anfragen benötigt, um dies herauszufinden.

cmd: Eingabeaufforderung
C:\Users\HP-2018\Superstar>superstar
Gebe den Namen der .txt-Datei einer Gruppe an (z.B. superstar4.txt)
superstar3.txt
Die Gruppe besteht aus 8 Mitgliedern.
Anfrage: Folgt Edsger Jitse?
Antwort: True
Anfrage: Folgt Tonrit Peter?
Antwort: False
Anfrage: Folgt Pia Rineke?
Antwort: False
Anfrage: Folgt Rinus Sjoukje?
Antwort: False
Anfrage: Folgt Jitse Tonrit?
Antwort: False
Anfrage: Folgt Pia Rinus?
Antwort: False
Anfrage: Folgt Jitse Pia?
Antwort: False
Lediglich Jitse könnte der Superstar der Gruppe sein. Bisher wurden 2 Anfragen gestellt. Es ist möglich, dass die Gruppe keinen Superstar hat. Soll kontrolliert werden, ob Jitse wirklich ein Superstar ist? (ja / beliebige andere Eingabe)
ja
Anfrage: Folgt Jitse Edsger?
Antwort: True
Jitse ist kein Superstar. Die Gruppe scheint keinen Superstar zu haben. Es wurden 8 Anfragen gestellt.
```

Zu sehen sind die Ausgaben zu den ersten drei Beispiel Gruppen.