**Ausarbeitung Weitere Programmiersprachen**

Dozent: Prof. Dr. rer. Nat. Michaela Huhn

Star-Voting

Felix Willrich Frederik Rieß

70452988 70453642

**Hochschule**

Ostfalia Hochschule für angewandte Wissenschaften  
Hochschule Braunschweig/Wolfenbüttel   
Salzdahlumer Straße 46/48  
38302 Wolfenbüttel



Inhalt

[1. Einleitung 1](#_Toc535837423)

[2. Star-Voting 1](#_Toc535837424)

[2.1. Kandidaten 1](#_Toc535837425)

[2.2. Bewertungswahl 1](#_Toc535837426)

[2.2.1. Bewertungswahl Auswertung 2](#_Toc535837427)

[2.2.2. Ungültige Stimmen 2](#_Toc535837428)

[2.3. Runoff 3](#_Toc535837429)

[2.3.1. Ergebnis 3](#_Toc535837430)

[2.4. Eigene Annahmen zur Wahlmethode 3](#_Toc535837431)

[3. Implementierung 4](#_Toc535837432)

[3.1. Imports 4](#_Toc535837433)

[3.2. Objektstruktur 4](#_Toc535837434)

[3.3. Votes überprüfen 5](#_Toc535837435)

[3.4. Daten einlesen- XML 6](#_Toc535837436)

[3.5. Bewertungswahl 7](#_Toc535837437)

[3.6. RunOff 8](#_Toc535837438)

[3.6.1. Kandidaten filtern 8](#_Toc535837439)

[3.6.2. Votes filtern 9](#_Toc535837440)

[3.6.3. Auswertung 10](#_Toc535837441)

[4. Fazit 11](#_Toc535837442)

# Einleitung

Das Projekt wurde im Rahmen der Lehrveranstaltung „Weitere Programmiersprachen“ für den Studiengang Informatik an der Ostfalia Hochschule für angewandte Wissenschaften konzipiert.

Das Programm umfasst eine in Haskell geschriebene Anwendung. Haskell ist eine funktionale Programmiersprache, die im Rahmen der Lehrveranstaltung als Beispiel diente den Studierenden funktionale Programmierung näher zu bringen.

Das Projektthema umfasst die Wahlmethode „Star-Voting“, welche im Folgenden erklärt wird.

# Star-Voting

Das Star-Voting wurde von dem amerikanischen Programmierer Mark Frohnmayer entwickelt. Das erste Konzept unter dem Namen „score plus top two“ oder „score runoff voting (SRV)“ wurde im Jahr 2014 vorgestellt. Der Name entstand dadurch, dass zwei Wahlmethoden zusammengeführt worden sind. Zum einen die Bewertungswahl (Punkteverteilung) und das Instant-Runoff-Voting. Erste Erfolge wurde mit dem Star-Voting im Jahr 2018 gefeiert, als es in Oregon (USA) für kleinere Wahlen benutzt worden ist.

## Kandidaten

Die Kandidaten dürfen sich beim Star-Voting, wie bei jeder anderen Wahlmethode anhand der Gegebenheiten aufstellen lassen. Die Namen werden in dem Zuge öffentlich gemacht und auf das Wahlformular gedruckt.

## Bewertungswahl

Die Bewertungswahl folgt dem Schema der Punkteverteilung. Beim Star-Voting gibt es die Möglichkeit die jeweiligen Kandidaten mit 0-5 Punkten zu bewerten. Null Punkte sind in dem Fall die niedrigste Punktevergabe, respektive sind 5 Punkte die höchste Punktevergabe.

Die Wählerschaft bekommt einen Zettel mit den jeweiligen Kandidaten und den jeweiligen Punkten, die angekreuzt werden müssen. Wahlweise ist auch ein Kästchen möglich, wo die Punktzahl eingetragen wird.

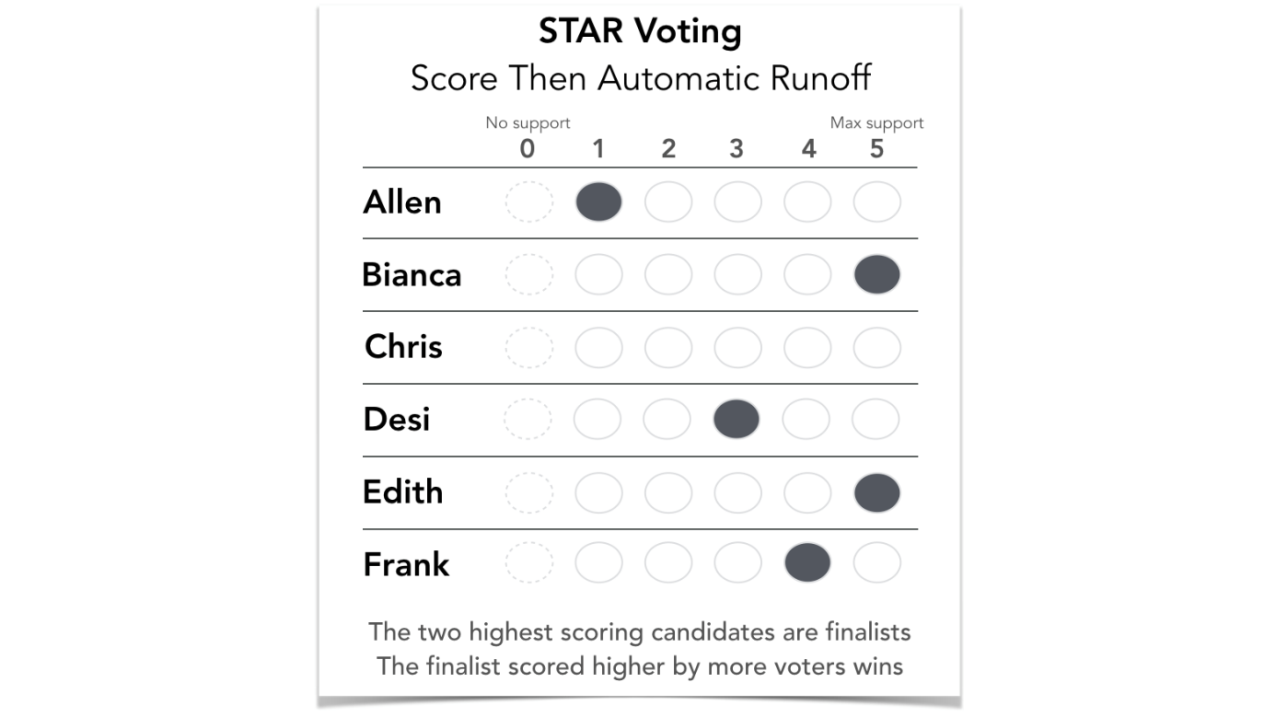


Abbildung 1: Beispiel eines Wahlzettels

Auf diesem Zettel darf der Wähler seine jeweilige Unterstützung in Form von Punkten notieren. Null Punkte bedeuten, dass der Wähler den Kandidaten mitnichten unterstützt und fünf Punkte, dass der Kandidat durch den Wähler voll unterstützt wird. Sollte keine Punktzahl angegeben werden, wird dies automatisch mit null Punkten bewertet.

Die Punktevergabe birgt den Vorteil, dass auch wenn der jeweilig höchstbewertete Kandidat nicht gewinnt, der Wähler einen großen Einfluss auf das Voting haben kann.

### Bewertungswahl Auswertung

Die Auswertung ist bei der Bewertungswahl simpel. Alle Punkte, die ein Kandidat bekommen hat, werden zusammengerechnet. Die Gesamtpunktzahl wird dann in einer Tabelle aufgelistet und die zwei Besten Kandidaten kommen in die nächste Runde, das sogenannte RunOff.

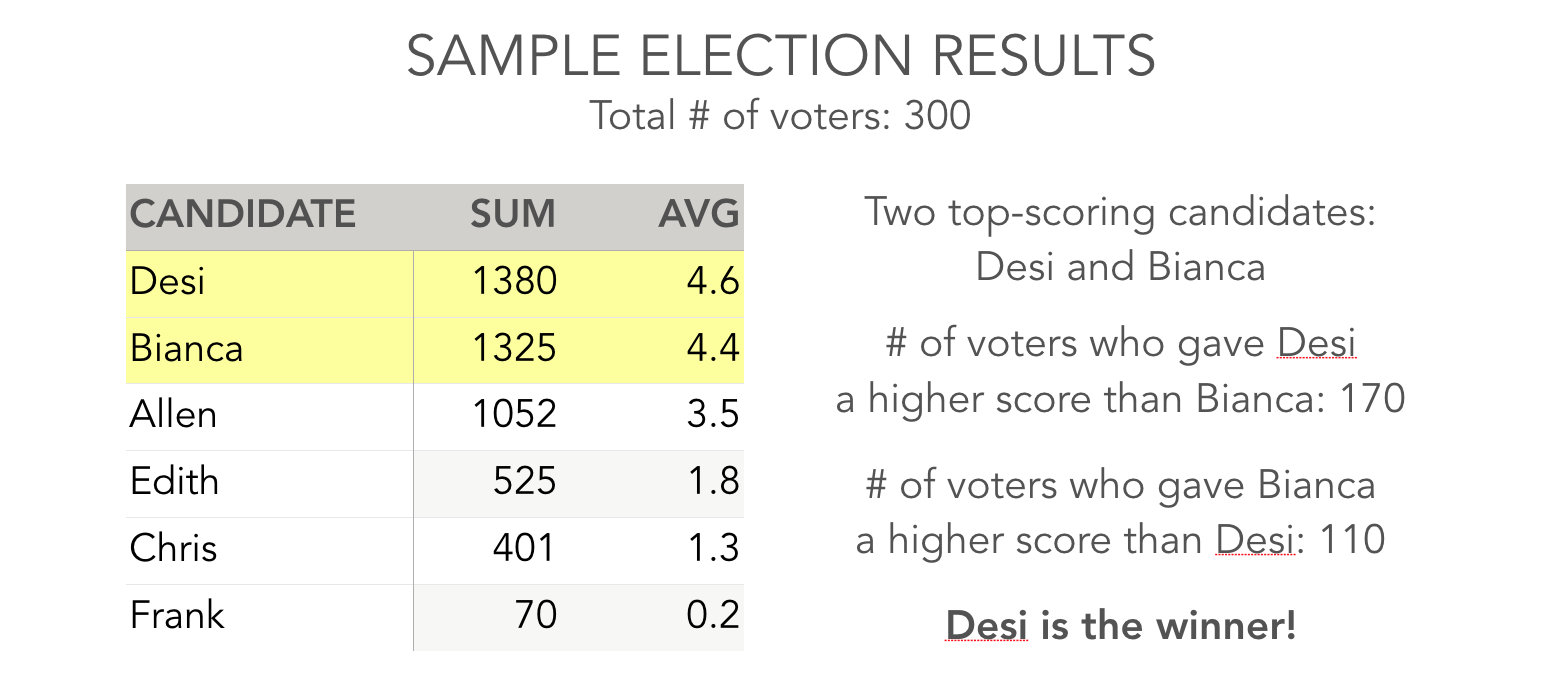


Abbildung 2: Tabelle zur Auswertung

Abbildung 3: Diagramm nach der Auswertung

### Ungültige Stimmen

Jede Stimme, die eins der folgenden Kriterien erfüllt, wird als ungültig angesehen und somit nicht in die Bewertung reingezogen (diese Kriterien wurden auch im Programm umgesetzt):

* Punktzahl kleiner 0 oder größer 5
* Kandidat muss zur Auswahl stehen

## Runoff

Das Runoff wird im Anschluss an die Bewertungswahl mit den zwei besten Kandidaten durchgeführt.

Das Prinzip vom Runoff ist ein direkter Vergleich. Alle abgegebenen Stimmen von den Wählern werden ein zweites Mal durchsucht und die direkte Punktzahl der beiden Anwärter wird vergleichen.

Dabei gibt es 3 Möglichkeiten, die eintreten können:

* Kandidat 1 hat eine höhere Punktzahl als Kandidat 2 = Kandidat 1 kriegt einen Punkt
* Kandidat 2 hat eine höhere Punktzahl als Kandidat 1 = Kandidat 2 kriegt einen Punkt
* Die Punktzahl ist gleich = Beide bekommen keinen Punkt

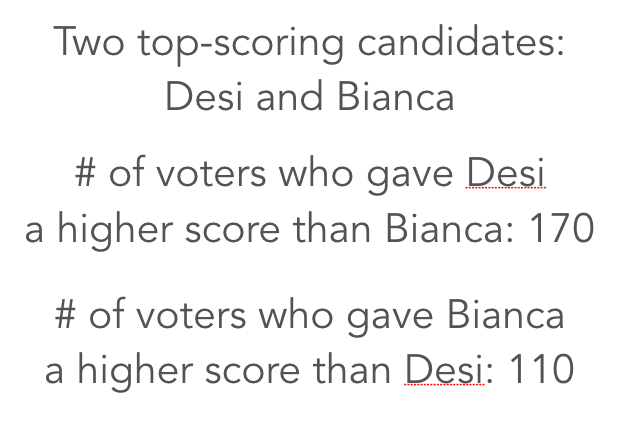


Abbildung 4: Auswertung Runoff

Da es in diesem Schritt nur noch gültige Stimmen gibt, aufgrund der zuvor gelaufenen Filterung, werden alle übriggebliebenen Stimmen berechnet.

### Ergebnis

Nachdem alle Stimmen im Runoff ein zweites Mal ausgewertet worden sind, kann der Sieger der Wahl bestimmt werden.

In unserem Beispiel aus den Bildern hatte Desi 170 Mal eine höhere Bewertung als Bianca. Wohingegen Bianca nur 110 Mal höher gewertet worden ist. Daraus folgt, dass Desi die Wahl gewonnen hat.

Sollte der Fall eintreten, dass beide Kandidaten, die gleiche Punktzahl am Ende des Runoffs haben, würde dies als Unentschieden gewertet werden und beide hätten gewonnen.

## Eigene Annahmen zur Wahlmethode

Da das Runoff an manchen Stellen kleinere Lücken offenlässt, haben wir in Rücksprache mit der Professorin Huhn eigene Annahmen zu den jeweiligen Fällen getroffen, um eine reibungslose Implementierung zu garantieren.

Das erste Problem tritt direkt zum Start der Wahlmethode ein. Wenn drei oder mehr Kandidaten in die Wahl geschickt werden, gibt es keine Probleme. Sobald es nur zwei Kandidaten sind, wird der Schritt mit der Bewertungswahl überflüssig und das Runoff wird direkt ausgeführt.

Weiterhin birgt die Wahlmethode größere Probleme, wenn nach der Bewertungswahl drei oder mehr Kandidaten übrigbleiben. Dies könnte passieren, wenn alle Aspiranten die gleiche Punktzahl haben oder der zweite Platz geteilt wird. In diesem Fall wird eine Neuwahl zwischen den übrig gebliebenen Kandidaten angesetzt.

# Implementierung

Wie in der Einleitung erklärt, wurde das Programm für die Veranstaltung „Weitere Programmiersprachen“ in der Programmiersprache Haskell entwickelt. Da dies in Gruppenarbeit geschehen ist, wurde ein eigenes Github Repository angelegt, um einen Austausch zu ermöglichen <https://github.com/Frede9/Star-Voting>.

## Imports

Damit das Programm reibungslos funktionieren kann, wird das XML Package „HXT“(http://hackage.haskell.org/package/hxt) benötigt. Hierzu muss in die Konsole „cabal install hxt“ eingegeben werden und automatisch werden alle Pakete installiert und die XML-Dateien können verarbeitet werden.

HXT funktioniert reibungslos unter Linux, sollte ein Windows System vorliegen, braucht es ein Interpretationsprogramm zwischen Linux und Windows Befehlen.

Damit die XML Dateien verarbeitet werden, muss der Pfad innerhalb des Programms mit dem Dateipfad übereinstimmen.

## Objektstruktur

Zum Anfang des Projekts haben wir uns eine Struktur ausgedacht, um die jeweiligen Daten besser verwalten zu können.

Um die jeweiligen Kandidaten speichern zu können, haben wir eine Data-Struktur verwendet, welche nur einen String enthält. Dies hilft uns im Nachhinein bei den Results, bzw. bei jedem einzelnen Vote den Aspiranten zu hinterlegen.



Abbildung 5: Candidate-Objekt

Das Candidate-Objekt wird in den Results genutzt. Die Results sind dazu gedacht, um das Ergebnis der Bewertungswahl und des Runoffs zu speichern. Dazu wurde eine Liste mit Tupeln angelegt. Der Vorteil liegt darin, dass jedes einzelne Tupel in der Liste vergleichen werden kann. Das Tupel selbst beinhaltet den Kandidaten bzw. den Namen und einen Double Wert für das Ergebnis. Das Objekt wird für beide Wahlmethoden als Auswertungsmedium genutzt.

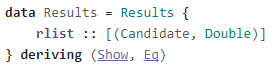


Abbildung 6: Result-Objekt

Das Candidate-Objekt wird dazu auch im Vote genutzt. Das Vote-Objekt wurde konzipiert, um die Daten vom Wahlzettel speichern zu können. Ein Vote ist eine Stimme für einen Kandidaten. Bei 5 Anwärtern bräuchte man 5 Vote-Objekte, um den Wahlzettel komplett speichern zu können. Damit dies übersichtlich und einfach verarbeitbar bleibt, wurde ein Voter-Objekt angelegt, worin die einzelnen Votes gespeichert werden. Somit ist ein Eintrag in der Liste von dem Voter-Objekt ein Wahlzettel eines Wählers.

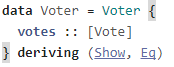
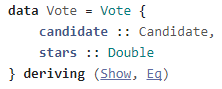


Abbildung 7: Voter-Objekt

Abbildung 8: Vote-Objekt

## Votes überprüfen

Gültige Votes sind solche, die die passende Anzahl an Kandidaten mit den jeweils richtigen Namen besitzen. Zudem sollten die einzelnen Bewertungen mit der Maximal- (5) und Minimal- anzahl (0) der zu vergebenen Sterne übereinstimmen. Falls der Fall eintritt, dass ein Vote eines Voters nicht gültig ist, so sind auch die übrigen Votes ungültig und werden nicht mit in die Bewertung aufgenommen.

In der implementierten Funktion wird daher eine Liste mit Votes sowie eine Liste mit den richtigen Kandidaten als Parameter übergeben. Zurückgeliefert wird anschließend ein Bool. Da die Funktion so arbeitet, dass am Ende zwei leere Listen übergeben werden um True wiederzugeben, werden durch Pattern Matching zunächst die verschiedenen Fälle abgefangen. Danach werden die Sterne des aktuellen Vote überprüft. Falls dies schon nicht zutrifft, wird sofort False returned. Ist dann der Kandidat in dem Vote auch auf der korrekten Liste der Kandidaten zu finden, so ist der Vote korrekt und es wird rekursiv die Funktion mit den restlichen Votes sowie der Kandidatenliste ohne den gefundenen Kandidaten aufgerufen. Dadurch wird auch der Fall abgefangen, wenn in den Votes zweimal der gleiche Kandidat zu finden ist. Falls der Kandidat von einem Vote nicht Teil der richtigen Kandidaten ist, so wird False returned.

## Daten einlesen- XML

Um große Datensätze nicht manuell eingeben zu müssen, werden diese zuvor mit einem Python-Skript als XML-Datei erstellt. Anschließend erfolgt das Einlesen der Datei über einen XML-Parser. Die Haskell XML Toolbox (HXT) dient dazu XML in Haskell darzustellen. Bei HXT werden das Arrow-Konzept und die Kombination von Filtern angewandt.

Die Funktion parseXML wird für das initiale Parsen der XML-Datei verwendet.

Durch den deep-Filter werden alle unterliegenden Strukturen rekursiv mit einem bestimmten Tag durchlaufen. Die verschiedenen Funktionen (getVote, getVotes und getCandidates) suchen so jeweils nach dem passenden Tag. In diesen Methoden sind viele Ähnlichkeiten mit Monaden zu erkennen. Die sogenannte Proc-Syntax verhält sich wie ein Lambda-Ausdruck. Es werden hier jedoch keine Funktionen sondern Arrows konstruiert. Die Variable x dient dabei als „Input“ des Arrows. -< stellt den Beginn eines Arrows dar:

... **<-** ... **<<<** ... **<<<** ... **-<** ...

## Bewertungswahl

|  |
| --- |
|  |
|  |

|  |
| --- |
| Nach dem Parsen der XML-Dateien folgt nun die erste Auswertung der Votes. |
|  |

|  |
| --- |
| In der Auswertung werden die gesamten Votes der Voter für jeden einzelnen Kandidaten summiert, sodass am Ende eine Liste entsteht, die alle Kandidaten und deren Gesamtbewertung enthält. Dafür werden die Votes am Anfang gefiltert und anschließend ausgewertet. |
|  |

|  |
| --- |
| 3.5.1 Filtern der Votes |
|  |

|  |
| --- |
| In der Funktion getListOfVoters werden, unter anderem für das spätere Runoff, zunächst die Voter mit ihren Votes gefiltert, die nur reguläre Votes abgegeben haben. Dafür sind die Liste der Voter und der Kandidaten als Parameter übergeben, um dann die gefilterte Liste der Voter zurückzugeben. Durch die Funktion checkVotes werden die Votes eines Voters überprüft und entsprechend je nach Ergebnis an die Ausgabeliste drangehangen. |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| 3.5.2 Auswerten der Votes |
|  |

|  |
| --- |
| Nach dem Filtern der Votes müssen diese nun ausgewertet werden. Eine Map verwaltet die einzelnen Kandidaten und deren Ergebnisse. Die Methode getResults besitzt also einen Parameter mit den einzelnen Votern und einer Map. Als Ergebnis entsteht eine Liste von Tupeln mit den Kandidaten und deren Ergebnis. |
|  |

|  |
| --- |
| Über die Funktion insertVotes werden die Votes eines Voters der Map hinzugefügt. Durch die Methode Map.insertWith (+) werden die Kandidaten als Key geupdatet und die entsprechende Bewertung hinzugerechnet. Wenn alle Votes eines Votes durchgegangen worden, werden in der Funktion getResults die nächsten Votes eines Voters übergeben. |
|  |

Abschließend, falls die Liste der Voter leer ist, wird die Map in eine Liste umgewandelt.

## RunOff

Das Runoff besteht im Code aus mehreren Schritten. Zuerst werden die Kandidaten und die Votes vorbereitet. Im Anschluss wird das Runoff durchgeführt.

Nachdem die Daten per XML eingelesen und verrechnet worden sind, gibt es zwei Datensätze mit denen weitergearbeitet wird. Zum einen gibt es ein Result-Objekt mit den Kandidaten zuzüglich der zusammengerechneten Votes. Auf der anderen Seite gibt es eine Liste der Voter, die alle korrekten Stimmen beinhaltet.

### Kandidaten filtern

Im ersten Schritt des Runoffs werden die beiden Aspiranten mit der höchsten Punktzahl aus dem Result-Objekt gefiltert.

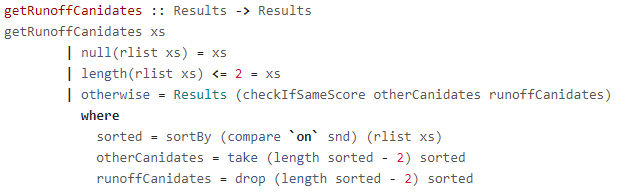


Abbildung 9: Runoff-Kandidiaten filtern

Dazu wir die Liste genommen und zuerst als insgesamtes betrachtet. Sollte die Liste zwei oder weniger Einträge enthalten, wird die Liste unverändert zurückgegeben. Sobald die Liste mehr als zwei Einträge enthält wird eine Filterung vorgenommen. Die Liste wird zuerst sortiert und danach aufgeteilt in die zwei höchsten Kandidaten und den Rest. Diese beiden Paramater werden an die nächste Funktion weitergegeben, um zu überprüfen ob der Zweitplatzierte die gleiche Punktzahl besitzt wie Einträge in der „otherCanidates“ Liste.

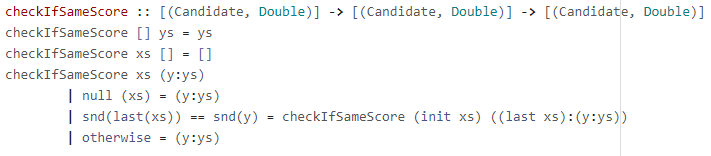


Abbildung 10: Überprüfen der Kandidaten auf gleiche Ergebnisse

Auch in dieser Methode gibt es drei Fälle, die unterschieden werden. Zuerst darf die Liste der restlichen Kandidaten nicht leer sein, sonst würden die Runoff-Kandidaten direkt zurückgegeben werden. Dies wird nur eintreten, wenn zuvor weitere Kandidaten zur Runoff-Liste hinzugefügt worden sind.

Sollte der Fall eintreten, dass der erste Eintrag der Runoff-Kandidaten und der letzte Eintrag der restlichen Kandidaten die gleiche Punktzahl nach dem Auswerten besitzen, wird dieser zur Liste hinzugefügt und die Methode wird neu aufgerufen.

Tritt keiner dieser Fälle ein, ist die Überprüfung abgeschlossen und das Ergebnis wird zurückgegeben.

Bei mehr als zwei Kandidaten wird eine Neuwahl angesetzt.

### Votes filtern

Um das Runoff einfacher und performanter zu gestalten, gibt es den Zwischenschritt, dass die Liste der Votes gefiltert wird. Hierzu werden alle überflüssigen Kandidaten aus der Liste gelöscht.

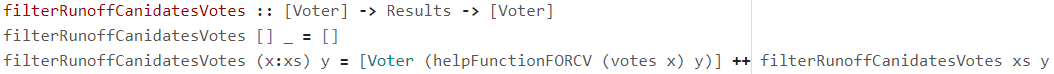


Abbildung 11: Votes filtern

Die Funktion „filterRunoffCanidatesVotes“ ist eine reine Transferfunktion. Es soll die Liste der Votes aufteilen und an die Helferfunktion weiterleiten. Dazu werden die Kandidaten eingelesen.

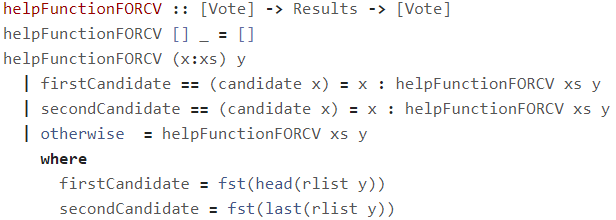


Abbildung 12: Helferfunktion für das Filtern der Votes

Die Helferfunktion filtertet jeden einzelnen Wahlzettel bzw. jede einzelne Bewertung auf die mitgelieferten Kandidaten. Sollte der Name übereinstimmen, wird die Stimme in eine Liste gepackt und am Ende zurückgegeben. Somit ist sichergestellt, dass nur die relavanten Daten übrig bleiben.

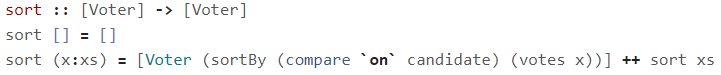


Abbildung 13: Sortierung Wahlzettel

Der letzte Schritt stellt sicher, dass ein einfaches vergleichen im Runoff selbst möglich ist. Die neu erstellte Liste der relevanten Stimmen wird sortiert. Dies geschieht in dem die Namen der einzelnen Kandidaten verglichen wird. Zurück kommt eine sortiere Liste alle Voter.

### Auswertung

Damit der letzte Schritt des Runoffs durchlaufen kann, müssen die Daten der vorrangegangen Filterfunktionen in diese Methode eingespeist werden.

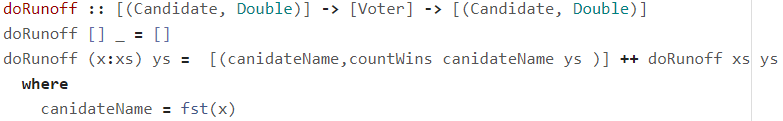


Abbildung 14: Hauptmethode Runoff

Die Funktion „doRunoff“ ist auch eine Transferfunktion. Primär dient sie dazu, die Kandidaten aufzuteilen und die Daten an die Hilfsfunktion weiterzuleiten. Als Ergebnis wird ein Tupel zurückgegeben, welches in ein Result-Objekt geschrieben wird.

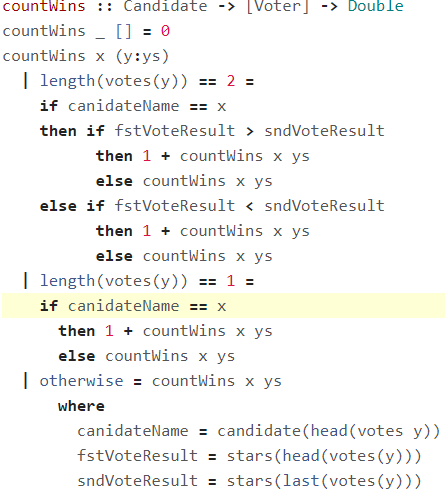


Abbildung 15: Unterscheidungen für die jeweiligen Fälle im Runoff

Das Zählen der Auswertung geschieht über mehrere If-Abfragen. Zuerst wird überprüft, ob die Länge der Liste zwei oder weniger ist.

Sollten beide Kandidaten vorhanden sein, wird überprüft, ob der erste Name in der Liste übereinstimmt mit dem übergebenen Kandidaten. Ist diese Aussage wahr, wird kontrolliert, ob die Punktzahl höher ist. Wenn auch diese Aussage wahr ist, wird ein Punkt draufgerechnet und die Methode wieder aufgerufen.

Sollte der Name nicht übereinstimmen, wird angenommen, dass der zweite Kandidat der übergebene Aspirant ist. Diese Annahme können wir auf aufgrund der Filterung der Votes treffen. In diesem Fall wird überprüft, ob der zweite Eintrag mehr Punkte bekommen hat. Sollte dies geschehen wird ein Punkt draufgerechnet und die Methode erneut aufgerufen.

Wenn die Länge der Liste genau eins ist, muss nur überprüft werden, ob der Kandidat dem Eintrag in der Liste gleicht. Tritt dies ein, bekommt der Aspirant einen Punkt hinzugerechnet.

Sollte keiner dieser Fälle eintreten, wird die Methode solange aufgerufen bis keine Votes mehr übrig sind.

# Fazit