1 Introduction

In Project 6, we wrote a `Voter` class, which models a single voter in an election; the voter has a list of candidates that they prefer. The `vote()` method of the `Voter` class takes a list of candidates as a parameter, and we compare the candidates to the voter's preferences; it returns the index of the candidate chosen.

In this project, we'll be adding to that code. First, we'll implement a simple election - meaning we'll write a method which asks all of the voters to vote, and then collects the vote totals and immediately announces a winner.

Then, we'll implement the Alternative Vote - where, if no one candidate got more than 50% of the vote, we'll remove one candidate from the list (the one which got the fewest votes), and run the election again, over and over.

Please review the Project 6 spec to remind yourself what was required for that project. You can also review the video about the Alternative Vote here: [https://www.youtube.com/watch?v=3Y3jE3B8HsE](https://www.youtube.com/watch?v=3Y3jE3B8HsE)

1.1 Pseudocode Changes

After talking with the SLs, I have decided to focus the pseudocode requirement for this project. Instead of writing pseudocode for the entire project, you will instead write pseudocode only for a few key algorithms. The intent of this change is to use pseudocode to focus on the most interesting (and most challenging) parts of the project.

Here's the tradeoff: in the past, the SLs have been relatively lenient with pseudocode, and so some students have been turning in very poor, broken code. Starting with Project 7, the SLs will now grade your pseudocode a lot more like your Java code - it needs to work properly.

You'll have less pseudocode to write, but I hope that you will be more careful as you write it!

2 Overview

In this Project, you will turn in two classes. The `Voter` class hasn't changed since Project 6 (except for one little detail about `vote()`, which we'll detail below). You will also turn in a `Proj07_Voting` class (NOTE: it's a different name than Project 6!). This class will be an expanded version of the `Proj06_Voting` class from Project 6.
In Project 6, you had to read candidate and voter information from a file; you printed out all of this information, and then had each voter vote in the election (though you didn’t count the votes).

In Project 7, you will write several new methods (detailed below). One will implement a simple election; one will implement an Alternative Vote election. Both of these methods will call the doVoting() method, which will do one round of voting. They will also have another helper method, named arrayDel(), which will delete a single element from an array (this is used in the Alternative Vote).

2.1 Output

Your SL will be using a script to do the testing on your code. For this reason (like in some previous projects), it’s important that you make your output look exactly like the example output that I provide.

3 Requirements

3.1 Pseudocode

Write pseudocode for the following methods. All methods are detailed below.

- doVoting()
- doSimpleElection()
- doAlternativeVoteElection()

Remember: you will be graded on the correctness of your pseudocode this time!

3.2 Java

Your program must do the following (see later sections for details):

- Implement all of the Project 6 requirements.
- Update the vote() method in the Voter class to return -1 (and print out the appropriate message) if none of the voter’s preferences are in the current candidate list.
- Add a doVoting() method to the Proj07_Voting class.
- Add a doSimpleElection() method to the Proj07_Voting class. This method must call doVoting() to run its election.
- Add a doAlternativeVoteElection() method to the Proj07_Voting class. This method must call doVoting() to run each of its elections. It must also call arrayDel() to remove candidates from the array.
• Add a `arrayDel()` method to the `Proj07.Voting` class.

3.3 Output

The TAs will be using scripts to help them grade this project. So your output should look exactly like the example output files that I’ve provided. Since that output is longer than in Project 6, I won’t include it in the spec - please go look at the example files I’ve posted online.

3.3.1 Downloading the Inputs

Use the same input files as from Project 6. However, the output files are now longer, since you are actually running some elections. You can find the inputs and outputs here:

http://www.cs.arizona.edu/classes/cs127a/spring16/projects/proj07_files

4 The Voter Class

The `Voter` class works exactly like in Project 6, except that we now need to add a detail.

In Project 6, I didn’t say exactly what `vote()` must return if none of the preferences match the candidates. In Project 7, we need to implement this. You must print out a message which looks like this:

Gorilla chooses not to vote.

The method must then return -1.

5 The Proj07.Voting Class

`Proj07.Voting` must implement the same requirements as `Proj06.Voting`, but additionally must also implement the following methods. (main() should call `doSimpleElection()` and `doAlternativeVoteElection()`.)

5.1 The doVoting() Method

This method takes two parameters: an array of voters, and an array of candidate names. It returns an array of `int`, representing the number of votes earned by each candidate.

This method must take into account that some voters might return -1 from their `vote()` method (meaning that they choose not to vote). Such votes should not be recorded in the array that is returned (effectively, those voters just stayed home and didn’t vote on Election Day).
5.2 The doSimpleElection() Method

Write this method first, it’s easier than the Alternative Vote!

This method implements a simple (that is, First Past the Post) election. It has
two parameters: an array of voters, and an array of candidate names. It must
return void.

Call the doVoting() method to actually do the voting process; that method
will return an array of int, which represent the vote counts. Then scan through
the results, and report which candidate won the election.

If two candidates have the same number of votes, report the first one in the
array - sorry, no ties in this election!

5.3 The doAlternativeVoteElection() Method

This method implements an Alternative Vote election. It has the same parame-
ters and has the same return type as doSimpleElection(), and it likewise calls
doVoting() to actually hold each election. However, it has a loop, where it may
hold many elections.

Each pass of the loop must perform the following steps:

• Call doVoting() to hold the election.

• Count the total # of votes, and find the candidates with the most and
least votes.

• If the candidate with the most had more than half the votes, then report
them as the winner and return from the method.

• Otherwise, use the arrayDel method to remove the candidate with the
least votes from the array, and try again.

If there are any ties (for most votes or least votes), then always use the first
in the array.

Note, however, that there will never be a tie in the last pass of the loop. If
a candidate gets exactly 50% of the vote, keep looping around until somebody
gets more than 50%.

5.4 The arrayDel() Method

This method takes two parameters: an array of String(), and an int. It
returns a duplicate of the array, but with that one element removed.

You are not required to include any error checking in this method; you may
assume that the parameters are reasonable.

5.5 Additional Methods?

You may add as many additional methods (to either class) if you would like.
However, none are required (other than the ones I’ve already described above.)
6 Pseudocode

Download the latest pseudocode requirements from:
http://www.cs.arizona.edu/classes/cs127a/spring16/projects/
Project 7 will be graded based on v1.3 of the requirements.

7 Java code

After you complete the pseudocode (and turn it in!), start working on translating it to Java. (For a list of steps for converting the pseudocode to Java, see the Project 1 spec.)

Your classes must have exactly the names given above: Proj06_Voting and Voter. Make sure that your Java files are named to match.

7.1 Header Comment

Every Java file that you submit must have a Header Comment - which is a simple comment, at the top of your file, which gives basic information about it. A header comment must include:

- The name of the Java class
- The name of this class (“CSc 127A Spring 16”)
- The assignment name
- Your name
- Your Section Leader’s name (or letter)
- A quick description of the Java class.

Here’s my suggested header comment for Program 6:

/* class ScanString
 * CSc 127A Spring 16, Project 07
 * 
 * Author: TODO: your name here
 * SL Name: TODO: the name of your SL here
 * 
 * ---
 * 
 * TODO: put a quick (2-3 sentence) description of the class here
 */
Late Day Clarifications

Pseudocode

The syllabus says that you can use Late Days for late projects - however, the syllabus does not explicitly say how this works with pseudocode. The SLs and I talked it over, and we decided on the following policy:

- Late days cannot be spent on pseudocode. Turn it in on time, or you’ll just miss that part of the project grade.
- You must turn in your pseudocode using `turnin`; email will no longer be accepted.

(We were flexible about both of these points in the first two projects - but starting with Project 3, we’re going to enforce this.)

Java code

Starting with Project 3, if you email us your code (instead of using `turnin`), it will count as a Late Day, even if you do it before the deadline. **Please use `turnin`.** And remember: if you are having trouble connecting from home, it’s always possible to come to the 228 lab, and upload your file using a thumbdrive.

8 Turning in Your Programs

See the class website for information about how to upload your files to lectura, and how to use the ‘turnin’ utility.

8.1 Pseudocode

Your pseudocode file must be either TXT or PDF files (one per program), but you may name them whatever you want. You must turn them in using the assignment name

`cs127a_s16_sX_proj07_pseudocode`

(replace the X with your section letter). **REMEMBER:** The pseudocode is due two days before the Java code!

8.2 Java Code

You must turn in file(s) named exactly this:

```
Proj07_Voting.java
Voter.java
```

You must turn in the Java file(s) using the assignment name

`cs127a_s16_sX_proj07`

(replace the X with your section letter).