Creating a Framework to Measure Voting Power: Exploring the Application of Mathematical Formulas in the World of Politics

Senior Honors Thesis By:

Cody Kovar

Presented to the Honors Committee

of McMurry University

in Partial Fulfillment of the Requirements for

Undergraduate Departmental Honors in:

Computer Science

Mathematics

Political Science

McMurry University

Spring 2019

Creating a Framework to Measure Voting Power: Exploring the Application of Mathematical Formulas in the World of Politics

APPROVED:

Thesis Director: Dr. Paul Fabrizio, Professor of Political Science

Committee Member: Dr. Mark Thornburg, Associate Professor of Mathematics

Committee Member: Mr. Richard Brozovic, Instructor of Computer Science

Honors Program Director: Dr. Philip LeMasters, Professor of Religion

ACKNOWLEDGEMENTS

McMurry University has prepared me for this very moment. I was incredibly prepared for the undertaking of such a large-scale project such as this. This interdepartmental thesis was the product of a high standard of education, dedication from the most incredible faculty, and the grace and understanding of all of those who bared with me through my goals and aspirations.

I would first like to thank Dr. Philip LeMasters for his direction of the overall Honors Department at McMurry University. Dr. LeMasters has worked with me very closely to ensure that I was able to complete all of the necessary requirements for this project and that I was able to even take on a project of this scale to begin with. His confidence in me to undertake this project as a whole is a large reason that it even exists.

Next, I would like to thank Dr. Mark Thornburg for his direction in the field of Mathematics. Dr. Thornburg taught me to take the math that we learn in the classroom and look deeper. His critical thinking challenges are what gave this project its initial footing. Without his ideas and suggestions, I might not ever have settled on one voting power calculation and would have had to call it quits early on.

I would also like to thank Mr. Richard Brozovic for his direction in the field of Computer Science. Mr. Brozovic, over the years, has given me all of the necessary tools to take on a programming challenge of this size from scratch. His teachings of coding standards that were instilled into me from the start of my freshman year stuck with me and helped make this project more easily digestible for those who read through it.

Finally, I would like to thank Dr. Paul Fabrizio for his direction in the field of Political Science. I had a very fundamental understanding of politics before I entered college, but because of Dr. Fabrizio, I was confident enough to undertake a project in which I introduce a sort of political theory into the world of politics, and I feel that speaks volumes into how much he has been able to teach his students.

I would also like to take a moment to thank those who were not directly involved in this thesis, but without them, it would not have been possible. Dr. Cynthia Martin and Dr. Kelly McCoun of the Department of Mathematics at McMurry University, Dr. Robert Watson and Dr. Alicia Wyatt of the Department of Computer Science at McMurry University, and Dr. Tina Bertrand and Dr. Bihter Tomen of the Department of Political Science at McMurry University. Dr. Joel Olufowote, Professor of Political Science at Indiana Wesleyan University, also gave me a significant amount of direction while I was a student of his during his time as a professor at McMurry University.

A final thank you to all whom I may not have mentioned personally but have believed in me and have stuck by me along the way. You are the reason this project is able to exist.

Abstract:

Seeking a way to create a framework to calculate voting power, this project design is intended to lay a foundation for future works in the political realm by assigning a quantifiable value to individual groups in a given population to show an estimate of how much voting power the group actually has. By utilizing mathematical power indices, we, in theory, should be able to identify one index that can work with the given data well enough to show a relevant correlation to provide a sufficient proof-of-concept. The implications of establishing a reliable measure of voting power in real-world political scenarios are far-reaching, however, one must be created first.

The Power Indices Explained

In determining whether or not voting power could even be feasibly calculated I turned to what the world of mathematics has already offered us. Now, there exists a myriad of prominent voting power formulas that exist today, but there are a couple in particular that stick out for this specific use case. The Banzhaf Power Index (BPI) and the Shapley-Shubik Power Index (SSPI) are both notable as they measure the voting power of coalitions. This is important as it seems the only way to determine the voting power of a specific individual would be to put them into as many subgroups of the population as possible and then map out the power of all of those subgroups. If we deem a subgroup as a coalition of voters, then these both become viable options. It is important to note, however, that these are mostly utilized in weighted voting systems, but we will be adapting it to our needs in the American standard of one-person-one-vote first-past-the-post voting system.

The first formula, the BPI, works by first mapping out all possible winning combinations of coalitions. The next step is determining a critical coalition by calculating in which combinations, and how many of those winning combinations, the coalition would be the swing voter, that is that their vote determines the success of the vote in its entirety. The formula has many different possible ways of being written, one way of writing it could be

$$[v\colon x_1, x_2 \ldots x_n]\ \{x, v \in \mathbf{Z}\ \&\ x \in \mathrm{S}\}$$

Where S is the set of all voting coalitions and v is the minimum required votes, also called the quota. The possible winning combinations would then be written as combinations of *x*.

The second formula, the SSPI, works by mapping out the list of voting groups and then mapping out every possible permutation of the voting groups, that is every possible combination of the group. Then for each permutation of voters, the pivot voter needs to be determined which is found by determining for which voting coalition is minimum required votes threshold met. The formula could be written identically to the BPI but with the addition that x_n is a pivot voter if and only if the sum of x_n and all previous element of a given permutation is equivalent to v:

$$[v\text{: } x_1, x_2 \dots x_n]\ \{x, v \in \mathbf{Z}\ \&\ x \in \mathrm{S}\}$$

Since the idea would be determining the voting power of a coalition to then reach the voting power of an individual it seems that the SSPI would be more valuable. I would like to note that any future work could utilize the addition of the BPI on top of the SSPI to show what subgroups could work together to accomplish an agenda or what demographics need to be reached in order to win a contested public office seat, but for the preliminary work here, the baseline seems to call for the SSPI alone to determine whether or not it would even be feasible.

Since we have chosen the Shapley-Shubik Power Index as the preferred method of determining voting power we can elaborate on it further with an example. The SSPI is usually utilized in the world of business in determining voting power amongst stockholders in a company, so let us use an example that fits that idea. Say we have a company called Company that has 3 main shareholders, *A*, *B*, and *C*. Shareholder *A* has 3 shares, *B* has 2 shares, and *C* has 1 share. The first step would be to determine *v*, the minimum required votes needed in order to pass a motion. This means *v* would be indicated by:

$$v = \frac{\sum_1^n (x_n)}{2} + 1$$

This shows that *v* is equivalent to 50% + 1, which is commonly the number of votes required to pass a motion in most voting systems, including companies. Simple majority (50% + 1) would dictate that our minimum shares be 4, as the sum of 3, 2, and 1 is 6, of which 50% is 3, and with the 1 added we have 4. Next, we must list all possible permutations of the given shareholders:

[4: *A* ***B*** *C*] = [6: 3 **2** 1]

[4: *A* ***C*** *B*] = [6: 3 **1** 2]

[4: *B* ***A*** *C*] = [6: 2 **3** 1]

[4: *B* *C* ***A***] = [6: 2 1 **3**]

[4: *C* ***A*** *B*] = [6: 1 **3** 2]

[4: *C* *B* ***A***] = [6: 1 2 **3**]

The shareholders that are in bold are those who are the pivot votes. This is determined by iterating through the shareholders individually from left to right and then marking which shareholders votes push the total number of votes past the minimum required. After this we run each shareholder through a formula similar to the one below:

$$\frac{Y_n}{S!}$$

Where Y_n is the number of times the individual shareholder, *n*, was the pivot voter and where *S* is the total number of shareholders, such that *S*! is the factorial of S. The number of permutations created showing all of the combinations of voters can be calculated by taking the factorial of the number of shareholders. This shows how complex these computations can be as 3! is equal to 6 but 8! is equal to 40,320. Using our data above we can determine that:

$$\boldsymbol{A} = \frac{4}{6} = 0.667 = \mathbf{66.67\%}$$

$$\boldsymbol{B} = \frac{1}{6} = 0.167 = \mathbf{16.67\%}$$

$$\boldsymbol{C} = \frac{1}{6} = 0.167 = \mathbf{16.67\%}$$

So, despite shareholder *B* having one more share and realistically one more vote than shareholder *C*, they both only are the pivot voter 16.67% of the time, while *A* is the pivot voter 50% of the time. For this given example we would say that A has 50% of the voting power in the contained system.

While this shows what some would call fair representation as every person involved in voting has at least some voting power according to the SSPI, there does exist other examples where this may not be the case. There are two more examples that we can explore to show that sometimes voting power becomes very skewed. Let us use the same variable names as above, but let us change the numbers around a little bit.

Let us create an instance in which there exists a dummy voter. Shareholder *A* has 3 shares, shareholder *B* has 3 shares as well, and shareholder *C* has 1 share. If we follow the same formula as we did before we will get the following 6 (3!) permutations:

$[5: A\ \boldsymbol{B}\ C] = [7: 3\ \mathbf{3}\ 1]$

$[5: A\ C\ \boldsymbol{B}] = [7: 3\ 1\ \mathbf{3}]$

$[5: B\ \boldsymbol{A}\ C] = [7: 3\ \mathbf{3}\ 1]$

[5: *B C **A***] = [7: 3 1 **3**]

[5: *C **A** B*] = [7: 1 **3** 3]

[5: *C **B** A*] = [7: 1 **3** 3]

From the data above, we see that *A* and *B* are the only two that are actually able to have any influence on the vote, as they are the only two pivot voters. If we use the same calculation as above, we will get:

$$\boldsymbol{A} = \frac{3}{6} = 0.500 = \mathbf{50.00\%}$$

$$\boldsymbol{B} = \frac{3}{6} = 0.500 = \mathbf{50.00\%}$$

$$\boldsymbol{C} = \frac{0}{6} = 0.000 = \mathbf{00.00\%}$$

As we can see from the calculations, *C* will never have a say on the matter. 50% of the total number of votes is 3.5, and when 1 is added (this is the "+ 1" in 50% + 1) we find 4.5 is the minimum number of votes needed to achieve a majority vote. Whether we choose to round up to 5 or to keep it at 4.5 is irrelevant, because no matter how many permutations of the shareholders we see, shareholder *C* can never add to *A* or *B* to make the total number of votes reach 4.5. Only *A*+*B* or *B*+*A* can yield a majority vote. Shareholder *C* in this specific instance is a dummy voter, with no real say in what happens in the company.

One final example of unfair voting power distribution is where there is only one entity which holds all of the power in the given system. One final time we will use the above variables to demonstrate how this happens. For this specific case, let shareholder *A* have 5 shares, let shareholder *B* have 2 shares, and let shareholder *C* only have 1 share. Besides shareholder *C* not having the greatest business sense, we see that both they and shareholder *B* have absolutely no say in terms of their voting power. We see this by once again showing all possible permutations:

[5: ***A*** *B C*] = [7: **5** 2 1]

[5: ***A*** *C B*] = [7: **5** 1 2]

[5: *B* ***A*** *C*] = [7: 2 **5** 1]

[5: *B C* ***A***] = [7: 2 1 **5**]

[5: *C* ***A*** *B*] = [7: 1 **5** 2]

[5: *C B* ***A***] = [7: 1 2 **5**]

In this case, we see that shareholder *A* is the only shareholder that truly has any say. The total number of votes in the system is 8 which means that the 50% + 1 vote, *v*, is 5. Since 5 is the minimum number of votes required to pass any motion in the company and shareholders *B* and *C* will never have enough votes on their own or combined to reach the 5 votes needed, they unfortunately will never have any significant impact on the vote.

It is incredibly important to note that the SSPI only gives us the percentage of times in which a given subgroup is the pivot voter, but for our intents and purposes we will determine that value as the voting power that each group has in the given political system. I would argue that this is a logical conclusion given that we are not claiming that individuals themselves are given X% voting power in a given system, just that the given subgroups of a population, if voting together, could conceivably make up a given portion of the available votes, more so than just a percentage of the population. If we were to leave the percentages of voting power as the percentage of votes a subgroup has compared to the entire population, our numbers are a bit different and potentially misleading. As in the United States, the first-past-the-post voting system dictates that one only needs 50% + 1 votes to win an election. As we saw in the example where only shareholder *A* had any votes, while the shareholder was only 62.5% of the entire population, the shareholder had full control of all votes, because of the number of shares they had.

The Custom Calculator

What good is all of this information without a way to calculate it? In order to have a reliable means of calculations, I built a Shapely-Shubik calculator with a Graphical User Interface (GUI), that is a user-friendly interface in which anyone should be able to pick up and use as opposed to having to enter in commands on the command line. All of the code and screenshots of the user interface can be found in the Appendix. Building one from scratch became important early on in this project as it quickly became apparent that customization will be the key to success.

As previously stated, maybe more times than one would have liked at this point, the SSPI uses permutations to calculate voting power. This can quickly become computationally expensive. The factorial growth curve is very steep, in that the growth can actually be quicker than exponential growth. When we get into nine or more subgroups we see a minimum of 362,880 permutations; the actual building of the subgroups is far more expensive computationally. However, I wanted to be able to calculate all groups of subgroups as well. When we calculate all 30 congressional districts as well, we bring our total number of groups to work with to 10,886,400. Each of these groups must be made and then iterated through, that is stepped through one by one, to calculate the pivot voter. As one can see, these calculations start to add up quickly.

In the interest of saving time, and my fingers, I also needed my SSPI calculator to read data in from a text file so that the data could be quickly entered and changed. For continuities sake, the ability to export the file was also added to be able to reference data in the future without having to recalculate every time.

All of these needs resulted in the program itself. The program is written in Java, contains 1190 lines of code spanning across 10 different code files, notated by the ".java" file extension. The project utilizes multiple different facets of common advanced programming tactics, including classes, comparators, and lists.

The program starts up on a splash screen where some title information is displayed as well as buttons that can continue to the next step of the program or show credits given to those who assisted in the project. The next step involves prompting users to choose whether or not they would like to enter the data manually or read in from a file. If they choose to do it manually, they will be taken to a new window where a blank text box lies. If the user needs help with formatting, there exists a button labeled "help" at the bottom. If they chose to read in from a file, a prompt appears to locate the file and then that data automatically populates the aforementioned window's text box as opposed to it being blank. The user is then able to submit the data for calculations.

The results screen is formatted similarly to the required input format, with voting power percentage in place of the populations. At this point in the program the user can either close the program, save the data locally in a text file, or start back over from the input screen to continue calculating data.

All code files in the Appendix are labeled with the title of the file as the heading, and the beginning of each code file has an explanation of what each of them does specifically.

The Data

Now that we have reviewed the two possible negative outcomes of the voting power calculations, have established a clear definition of what voting power actually is, and have a means of calculation, we can begin examining real-world data and understand how it is calculated. For the first real-world example, we will be examining data derived from the 2000 census on the race groups within Texas congressional districts. The race groups will be used as the given subgroups within the SSPI and these groups include White, Black, Native-American, Asian, Other Pacific Islander (OPI), and Two or more races. Since the census gives us the groups (congressional districts), subgroups (races), and populations of the subgroups, we can begin the process of utilizing the framework that the SSPI gives us for the calculation of voting power.

Table 1 located in the Appendix has the data derived from the census data. We see that there are 30 congressional districts to account for in 2000 along with the 6 race groups that belong to each district with their given populations.

When we look at *Table 1*, we see a standard set of data, population without anything too striking. There is a large percentage of almost every district's population that is made up of white people, but in the southern United States, this is not anything out of the ordinary. Looking at the output of the voting power calculation, we see things start to become a little less ordinary.

The section in the Appendix labeled *Output 1* in the appendix is where the output of the program lies, with the input being labeled *Input 1,* however that is just data from *Table 1* written in the format that the program can read it. As we can see, white people have all of the power in 28 of the 30 or 93% of the districts, and then they still have some power in the other 2 districts. It would not even be beneficial to make a graph or some sort of visual aid out of the data as it is incredibly lopsided. If every white person in districts 1 – 17, 19 – 27, and 29 – 30 voted together, they would have complete control over the person who represents the district.

This is almost astounding to see. It is shocking to see “White-100%” in the output, but even more shocking to see “Black-0%” in almost all of the outputs as well; not just the black voters but all other voters, other than white, have 0% voting power as well. An effective voting power of 0, as we saw above, means that unless one is white one would have no say in the choosing of their representative. This also assumes that everyone in the district will be voting together, which in reality will likely never be the case. People are unpredictable and do not necessarily vote where one thinks they would, given their generalized subgroup within the population. Ideally, a greater accuracy of the calculating of voting power could be created in future works, but that will be discussed in the Further Research and Practical Uses section.

We will come back to the data and interesting correlations in the next section, The Data Reviewed. For now let us look at the next data set: Age. *Table 2* shows us the entirety of the data again, with the same type of inputs (*Input 2*) and outputs (*Output 2*) as the first data set. This data set is much more diverse than the first section with no one group having a significant higher voting power than any other group, or any other anomaly like the last data set. It is composed of 9 blocks, all spaced out around 10 years, of voting age individuals.

The data categories themselves are interesting in that the census creates 10-year blocks and then switches to 5-year spans for the ages 54 – 59 and for 60 – 64 but then switches back to 10-year blocks immediately after. I am sure there exists a reason in which this is the case, however, I could not find a ready answer as to why.

The Data Reviewed

Taking the data derived from the census a step further and analyzing it, we see some very interesting correlations starting to form. This section will begin to detail possible extrapolations of the data collected by the calculations of the program along with the raw data itself while bringing in other works to assist in verifying any conjectures that are made in the interpretations of the data.

The obvious first choice to analyze would be the two districts from the first data set that did not have 100% white voting power, as the data breaks from the normal trend set for Texas Congressional Districts. In looking at the districts in the year 2000 for the 106th Congress, the 18th district and the 30th district are the only two districts in the state of Texas that have black congressional representatives. Whilst we do not know as to why this is the case, something we do know is that these districts are majority-minority districts, where a majority of the population is comprised of minorities. Majority-minority districts also make things a little bit more difficult to analyze as they sometimes are deemed to be unfair. In fact, one of the districts mentioned above, the 18th district, had legal ramifications. In 1995 the United States Supreme Court heard the case *Bush v. Vera* and in 1996 decided that "the Court noted that the proposed districts were highly irregular in shape, that their computerized design was significantly more sensitive to racial data, and that they lacked any semblance to pre-existing race-neutral districts" and that "the proposed districts violated the Voting Rights Act's 'results' test prohibiting activity that 'results in a denial or abridgment of the right of any citizen to vote on account of race or color'".[1]

[1] "*Bush v. Vera*." Oyez. Accessed April 1, 2019. https://www.oyez.org/cases/1995/94-805.

This decision was a 5 – 4 decision with the judges who lean conservatively comprising the majority.

Majority-minority districts seem to have stemmed from the Voting Rights Act of 1965. The Acts purpose as defined by the United States Department of Justice is "adopted at a time when African Americans were substantially disfranchised in many Southern states, the Act employed measures to restore the right to vote that intruded in matters previously reserved to the individual states."[2] Majority-minority districts were a solution to the disfranchised voters the tow which the Act was referring. By giving a large number of minorities a significant advantage in the deciding of who their representative should be, they should feasibly get a portion of the overall representation that the state has, at the national level as they are US congressional representatives. This could be the reason in which Districts 18 and 30 do have minorities as their representatives. Critics of the Voting Rights Act of 1965 say that the Act does not actually go far enough. David Epstein and Sharyn O'Halloran write in the American Journal of Political Science that "The key phrase in voting rights litigations is that minorities should have an 'equal opportunity to elect the candidate of their choice,' but this is bridled with problems in implementations" and that "'candidates of choice' have never been fully defined" which leaves much room to be decided by the interpretation of the courts and those who make the districts themselves.[3]

[2] "Introduction To Federal Voting Rights Laws." The United States Department of Justice. August 06, 2015. Accessed April 11, 2019. https://www.justice.gov/crt/introduction-federal-voting-rights-laws-1.
[3] Epstein, David, and Sharyn O'Halloran. "Measuring the Electoral and Policy Impact of Majority-Minority Voting Districts." *American Journal of Political Science*43, no. 3 (April 1999): 367-95. Accessed April 11, 2019. doi:10.2307/2991799.

Before we move on to discuss other data, there is one more interesting correlation that exists outside the data that could call for further investigation: location. Texas Congressional Districts 18 and 30 are in Houston and Dallas, respectively. These, as I am sure most if not all of this paper's readers know, are very large cities. These cities are so large, in fact, that they are both in the top 10 largest cities in the United States as of 2015 when ranked by population, with Houston being the fourth largest and Dallas being the ninth largest city.[4] This seems relatively unimportant to the project as a whole, as it is outside information that does not seem to directly correlate to anything, that is until we look at voting trends amongst cities versus rural areas. The Pew Research Center conducted a study which showed that conservative individuals tend to have a stronger desire to live in more rural areas while their liberal counterparts preferred cities. The numbers are a bit higher than one might expect, as "46% of consistent liberals said they'd prefer to live in a city, versus just 4% of consistent conservatives" and "liberals also are about twice as likely as conservatives to live in urban areas, while conservatives are more concentrated in rural areas."[5]

This gives us a second correlation to observe about Congressional Districts 18 and 30. The data from the Pew Research Center tells us that these large cities should be more liberal in nature, and that data supports reality. Besides being black congresswomen, both Sheila Jackson Lee of District 18 and Eddie Bernice Johnson of District 30 identify as members of the Democratic Party. It is important to note, however, that just because an area tends to be more urban as opposed to rural does not directly imply that it will be more liberal instead of

[4] "The 50 Largest Cities in the United States." PolitiFact. Accessed April 19, 2019. https://www.politifact.com/largestcities/.

[5] DeSilver, Drew. "Chart of the Week: The Most Liberal and Conservative Big Cities." Pew Research Center. August 08, 2014. Accessed April 19, 2019. https://www.pewresearch.org/fact-tank/2014/08/08/chart-of-the-week-the-most-liberal-and-conservative-big-cities/.

conservative, as it is just a noted correlation and not a causal relationship. Also, as we have lightly touched on, the way that maps are drawn have a significant impact on how the district as a whole will tend to be, similar to what we saw above with the majority minority districts.

Next, we have the age data to analyze. As previously stated, nothing immediately jumped out as a correlation to explore further, so we had to dig a little deeper. Interesting correlations can be seen once one considers who represented the given district during the time that the data was collected, like the case of race above. It has been long conjectured that young people tend to be more liberal in nature and as a result will vote for the Democratic candidate. The Pew Research Center did a study in 2018 and found evidence for this long-standing conjecture. The research center notes that over time "there has been an increase in the share of Americans expressing consistently liberal or mostly liberal views, while the share holding a mix of liberal and conservative views has declined", which leads us to believe that newer generations, once they come of voting age, will lean left ideologically.[6]

With that knowledge in mind, let us look at the correlations. During 2000, the districts that had Republican congressional representatives were districts 3 with Sam Johnson, 5 with Pete Sessions, 6 with Joe Barton, 7 with Bill Archer, 8 with Kevin Brady, 12 with Kay Granger, 13 with Mac Thornberry, 14 with Ron Paul, 19 with Larry Combest, 21 with Lamar S. Smith, 22 with Tom DeLay, 23 with Henry Bonilla, and 26 with Dick Armey.[7] The districts that had Democratic congressional representatives were districts 1 with Max Sandlin, 2 with Jim Turner, 4 with Ralph M. Hall, 9 with Nick Lampson, 10 with Lloyd Doggett, 11 with Chet Edwards, 15

[6] "The Generation Gap in American Politics." Pew Research Center for the People and the Press. March 1, 2018. Accessed April 12, 2019. https://www.people-press.org/2018/03/01/the-generation-gap-in-american-politics/.
[7] "Congress Profiles | US House of Representatives: History, Art & Archives." Congressional Profiles | US House of Representatives: History, Art & Archives. Accessed March 20, 2019. https://history.house.gov/Congressional-Overview/Profiles/106th/.

with Ruben Hinojosa, 16 with Silvestre Reyes, 17 with Charles Stenholm, 18 with Sheila Jackson Lee, 20 with Charlie Gonzalez, 24 with Martin Frost, 25 with Kenneth E. Bentsen Jr., 27 with Solomon P. Ortiz, 28 with Ciro D. Rodriguez, 29 with Gene Green, and 30 with Eddie Bernice Johnson.[8] The distinction in the difference of the political parties becomes important once the data in the Appendix, *Output 2* is analyzed.

If we take the voting power of each subgroup of age and average them for each district in Texas, we see a fitting correlation. The districts that had a congressional representative who was a member of the Democratic Party had a higher voting power amongst those aged 20 – 24 and 25 – 34. While this is not the case for every single district in a side-by-side comparison, the correlation exists when the average is taken. Conversely, the districts that had a congressional representative who was a member of the Republican Party had a higher voting power amongst those aged 35 – 44 and 45 – 54. For the other age groups (55+), they all had a higher voting power for Democratic congressional representatives, but the voting power difference seems too negligible to have a real impact one way or the other.

We know that young people tend to sway towards the left and that older voters tend to sway towards the right, however, if we look at the data about the urban versus rural regions given to us by the Pew Research Center and combine it with the data about the age demographics, can we see any correlations there? The short answer is yes, but the long answer seems to be a tad more nuanced. Using the Summary File 1 provided by the US Census, we have access to how many people in each given district live in urban areas and how many people live in rural areas. If we take that data and derive the percentage of people that live in urban areas versus rural areas for each political party, we start seeing correlations that do not necessarily

[8] Ibid.

agree with other research on the topic. The derived data shows that Congressional Districts that have a representative who identifies with the Democratic party actually have a lower percentage of constituents living in urban areas than their Republican represented counterparts. This also implies that Congressional Districts that have a representative that identifies with the Democratic party have a higher percentage of constituents living in rural areas than their Republican represented counterparts.

This statistic is incredibly shocking, and it is difficult to add commentary to, as it quite literally goes against the literature on the matter. This, however, could be explained multiple ways. The data suggesting the shifts in young voters being more liberal and the data suggesting that more liberal people prefer living in urban areas are at least a decade newer than the census data, which could imply that these correlations found by the Pew Research Center are newer trends that are more relevant to the population of today and not 2000. Another explanation could be that the difference is negligible, and that the urban versus rural differences do not actually have a significant impact on the electing of a representative, as the difference between the urban and rural population percentage of Democratic represented districts and Republican represented districts is .0121%.

Using this data, we were also able to find yet another interesting statistic about the Congressional Districts 18 and 30. Districts 18 and 30 have the lowest percentage of people living in rural areas by far – not very surprising since they are based in two of the most heavily populated areas in the United States – which means that the two districts also fit the correlations set by the Pew Research Center. However, the percentage of people who live in rural areas is astoundingly low, at just 0.0007% in district 18 and 0.0049% in district 30. It appears that these two districts fit most of the correlative data that exists based on their demographics alone.

Now that we have the data laid out and explained it is time to answer some very important questions. Why does this matter exactly? Why go through the trouble of showing the voting power of race groups in congressional districts in Texas in 2000? This project was a proof-of-concept to be built further upon in future works by providing these potential works with a framework of calculations. The correlations found by the data are arguably sufficient enough to make it viable as an expandable idea to be implemented and tested further. In the next section, we will discuss what works that utilize this data and these correlations could look like along with what real-world applications could be.

I would like to make one very important note here at the end of the data reviewed section that this research is not intended to take a stance on any issue or imply one way or the other in what action should be taken from the data derivations. Not enough has been done to determine root causes of why populations and voting power are the way that they are, but enough evidence has been presented to show that correlations do exist, whilst no causation research has been executed.

Further Research and Practical Uses

Further research could expand upon this greatly. Breaking down the six race groups further by age, socioeconomic status, religious preferences, or other categories that people identify with could strengthen this heavily. Another point of data that would be valuable to look at would be the data for the Hispanic population of each district. Because Hispanic identification does not rely on any racial ties, it gets a bit trickier to work with. Any person is either Hispanic, or they are not, and they can belong to any racial group or any other subgroup of a population for that matter. For this reason, the implementer of this idea in future works must take minor nuances such as that in consideration.

The implications of the data are very exciting though. If we were able to obtain similar to the information above about all given members of a district – of any kind, not strictly congressional – politicians could see what prominent groups they need to focus on reaching the most in order to increase their odds of winning. If politicians knew what groups had the most power, that could change how campaigning works, or if they knew the contents of their constituency they could tailor their legislation to them in order to make the largest impact for their community.

Another implication could be to see the change in voting power over time. If we are able to find this data for districts spanning a few decades and we calculated the power of all the given groups over time, could we see a change in the amount of power given communities have? We briefly discussed earlier the ideas of majority-minority districts and how they are allowed as long as there does not seem to be foul play involved and with accordance to the Voting Rights Act of 1965, as long as there are no "cracked" districts where certain groups are intentionally separated across district lines in order to minimalize their voting power. With that in mind, this framework could potentially use the calculations to see whether or not any districts did crack certain groups up in order to limit their voting power. If, for example, we use the data from the 2000 census and saw that districts 18 and 30 all of a sudden had 100% white voting power in 2010, along with the other districts, this could signal foul play, as they were the only two with non-white voters having the most power in the entire state of Texas.

Something that would be interesting to see is if we were to compare two states with similarly sized districts and see if there exists any differences in voting power amongst the given subgroups of the populations. While states in the south might tend towards catering to one political ideology and towards one race, do different groups yield more power up north? Could we see a day when people calculate how much voting power they would have somewhere before they chose to live there?

Appendix

Table 1:

	Total	White	Black	NativeAm	Asian	OPI	Two+
CD 1	622475	480028	105689	3529	2480	216	7900
CD 2	669591	519631	103685	3320	3052	225	8691
CD 3	835040	609159	80202	3948	66581	462	20860
CD 4	707329	602008	54135	5158	4318	245	11792
CD 5	657495	440493	114310	3914	9884	316	14410
CD 6	759418	621128	60952	3739	28398	1070	16077
CD 7	772147	558184	57948	3079	55965	456	23421
CD 8	776623	665282	40965	3039	19676	428	13862
CD 9	636960	428523	138273	2672	17326	261	12194
CD 10	791117	533878	75111	4630	36137	555	22861
CD 11	663275	466207	108807	4143	10038	1649	17951
CD 12	661753	492436	56919	4595	13989	1068	16279
CD 13	597401	443655	51020	4930	8257	291	13858
CD 14	688604	521030	62817	3664	5164	309	14465
CD 15	780310	599649	13809	3671	4605	295	17339
CD 16	620847	461298	18847	5126	6199	592	19934
CD 17	618958	520668	24896	3787	3328	227	11708
CD 18	606441	235344	244852	2562	18520	250	15737
CD 19	607535	492025	18479	4006	5956	202	12330
CD 20	624384	413828	36642	5985	8995	670	26149
CD 21	801078	692475	25810	3882	12945	607	16845
CD 22	784759	492840	115079	2930	78803	344	22168
CD 23	762627	587045	20918	4737	7635	406	20628
CD 24	680808	375452	144065	4559	17711	680	19742
CD 25	662264	354622	157362	2713	35805	459	19262
CD 26	845541	673943	50457	4063	39479	368	18930
CD 27	664428	506855	15201	3531	5351	349	17920
CD 28	646161	430656	52686	4983	4832	537	20972
CD 29	672591	341283	105836	4170	15071	446	23979
CD 30	633860	239880	248794	3297	15819	451	16236

Input Text 1:

```
Congressional District 1: White-480028 Black-105689 NativeAm-3529 Asian-2480 OPI-216 Two+-7900
Congressional District 2: White-519631 Black-103685 NativeAm-3320 Asian-3052 OPI-225 Two+-8691
Congressional District 3: White-609159 Black-80202 NativeAm-3948 Asian-66581 OPI-462 Two+-20860
Congressional District 4: White-602008 Black-54135 NativeAm-5158 Asian-4318 OPI-245 Two+-11792
Congressional District 5: White-440493 Black-114310 NativeAm-3914 Asian-9884 OPI-316 Two+-14410
Congressional District 6: White-621128 Black-60952 NativeAm-3739 Asian-28398 OPI-1070 Two+-16077
Congressional District 7: White-558184 Black-57948 NativeAm-3079 Asian-55965 OPI-456 Two+-23421
Congressional District 8: White-665282 Black-40965 NativeAm-3039 Asian-19676 OPI-428 Two+-13862
Congressional District 9: White-428523 Black-138273 NativeAm-2672 Asian-17326 OPI-261 Two+-12194
Congressional District 10: White-533878 Black-75111 NativeAm-4630 Asian-36137 OPI-555 Two+-22861
Congressional District 11: White-466207 Black-108807 NativeAm-4143 Asian-10038 OPI-1649 Two+-17951
Congressional District 12: White-492436 Black-56919 NativeAm-4595 Asian-13989 OPI-1068 Two+-16279
Congressional District 13: White-443655 Black-51020 NativeAm-4930 Asian-8257 OPI-291 Two+-13858
Congressional District 14: White-521030 Black-62817 NativeAm-3664 Asian-5164 OPI-309 Two+-14465
Congressional District 15: White-599649 Black-13809 NativeAm-3671 Asian-4605 OPI-295 Two+-17339
Congressional District 16: White-461298 Black-18847 NativeAm-5126 Asian-6199 OPI-592 Two+-19934
Congressional District 17: White-520668 Black-24896 NativeAm-3787 Asian-3328 OPI-227 Two+-11708
Congressional District 18: White-235344 Black-244852 NativeAm-2562 Asian-18520 OPI-250 Two+-15737
Congressional District 19: White-492025 Black-18479 NativeAm-4006 Asian-5956 OPI-202 Two+-12330
Congressional District 20: White-413828 Black-36642 NativeAm-5985 Asian-8995 OPI-670 Two+-26194
Congressional District 21: White-692475 Black-25810 NativeAm-3882 Asian-12945 OPI-607 Two+-16845
Congressional District 22: White-492840 Black-115079 NativeAm-2930 Asian-78803 OPI-344 Two+-22168
Congressional District 23: White-587045 Black-20918 NativeAm-4737 Asian-7635 OPI-406 Two+-20628
Congressional District 24: White-375452 Black-144065 NativeAm-4559 Asian-17711 OPI-680 Two+-19742
Congressional District 25: White-354622 Black-157362 NativeAm-2713 Asian-35805 OPI-459 Two+-19262
Congressional District 26: White-673943 Black-50457 NativeAm-4063 Asian-39479 OPI-368 Two+-18930
Congressional District 27: White-506855 Black-15201 NativeAm-3531 Asian-5351 OPI-349 Two+-17920
Congressional District 28: White-430656 Black-52686 NativeAm-4983 Asian-4832 OPI-537 Two+-20972
Congressional District 29: White-341283 Black-105836 NativeAm-4170 Asian-15071 OPI-446 Two+-23979
Congressional District 30: White-239880 Black-248794 NativeAm-3297 Asian-15819 OPI-451 Two+-16236
```

Output 1:

```
Congressional District 1:      White-100.0000%     Black-000.0000%  NativeAm-000.0000%
        Asian-000.0000%       OPI-000.0000%      Two+-000.0000%
Congressional District 2:      White-100.0000%     Black-000.0000%  NativeAm-000.0000%
        Asian-000.0000%       OPI-000.0000%      Two+-000.0000%
Congressional District 3:      White-100.0000%     Black-000.0000%  NativeAm-000.0000%
        Asian-000.0000%       OPI-000.0000%      Two+-000.0000%
Congressional District 4:      White-100.0000%     Black-000.0000%  NativeAm-000.0000%
        Asian-000.0000%       OPI-000.0000%      Two+-000.0000%
Congressional District 5:      White-100.0000%     Black-000.0000%  NativeAm-000.0000%
        Asian-000.0000%       OPI-000.0000%      Two+-000.0000%
Congressional District 6:      White-100.0000%     Black-000.0000%  NativeAm-000.0000%
        Asian-000.0000%       OPI-000.0000%      Two+-000.0000%
Congressional District 7:      White-100.0000%     Black-000.0000%  NativeAm-000.0000%
        Asian-000.0000%       OPI-000.0000%      Two+-000.0000%
Congressional District 8:      White-100.0000%     Black-000.0000%  NativeAm-000.0000%
        Asian-000.0000%       OPI-000.0000%      Two+-000.0000%
Congressional District 9:      White-100.0000%     Black-000.0000%  NativeAm-000.0000%
        Asian-000.0000%       OPI-000.0000%      Two+-000.0000%
Congressional District 10:      White-100.0000%     Black-000.0000%  NativeAm-000.0000%
        Asian-000.0000%       OPI-000.0000%      Two+-000.0000%
Congressional District 11:      White-100.0000%     Black-000.0000%  NativeAm-000.0000%
        Asian-000.0000%       OPI-000.0000%      Two+-000.0000%
Congressional District 12:      White-100.0000%     Black-000.0000%  NativeAm-000.0000%
        Asian-000.0000%       OPI-000.0000%      Two+-000.0000%
Congressional District 13:      White-100.0000%     Black-000.0000%  NativeAm-000.0000%
        Asian-000.0000%       OPI-000.0000%      Two+-000.0000%
Congressional District 14:      White-100.0000%     Black-000.0000%  NativeAm-000.0000%
        Asian-000.0000%       OPI-000.0000%      Two+-000.0000%
Congressional District 15:      White-100.0000%     Black-000.0000%  NativeAm-000.0000%
        Asian-000.0000%       OPI-000.0000%      Two+-000.0000%
Congressional District 16:      White-100.0000%     Black-000.0000%  NativeAm-000.0000%
        Asian-000.0000%       OPI-000.0000%      Two+-000.0000%
Congressional District 17:      White-100.0000%     Black-000.0000%  NativeAm-000.0000%
        Asian-000.0000%       OPI-000.0000%      Two+-000.0000%
Congressional District 18:      White-016.6667%     Black-050.0000%  NativeAm-000.0000%
        Asian-016.6667%       OPI-000.0000%      Two+-016.6667%
Congressional District 19:      White-100.0000%     Black-000.0000%  NativeAm-000.0000%
        Asian-000.0000%       OPI-000.0000%      Two+-000.0000%
Congressional District 20:      White-100.0000%     Black-000.0000%  NativeAm-000.0000%
        Asian-000.0000%       OPI-000.0000%      Two+-000.0000%
Congressional District 21:      White-100.0000%     Black-000.0000%  NativeAm-000.0000%
        Asian-000.0000%       OPI-000.0000%      Two+-000.0000%
Congressional District 22:      White-100.0000%     Black-000.0000%  NativeAm-000.0000%
        Asian-000.0000%       OPI-000.0000%      Two+-000.0000%
Congressional District 23:      White-100.0000%     Black-000.0000%  NativeAm-000.0000%
        Asian-000.0000%       OPI-000.0000%      Two+-000.0000%
Congressional District 24:      White-100.0000%     Black-000.0000%  NativeAm-000.0000%
        Asian-000.0000%       OPI-000.0000%      Two+-000.0000%
Congressional District 25:      White-100.0000%     Black-000.0000%  NativeAm-000.0000%
        Asian-000.0000%       OPI-000.0000%      Two+-000.0000%
Congressional District 26:      White-100.0000%     Black-000.0000%  NativeAm-000.0000%
        Asian-000.0000%       OPI-000.0000%      Two+-000.0000%
Congressional District 27:      White-100.0000%     Black-000.0000%  NativeAm-000.0000%
        Asian-000.0000%       OPI-000.0000%      Two+-000.0000%
Congressional District 28:      White-100.0000%     Black-000.0000%  NativeAm-000.0000%
        Asian-000.0000%       OPI-000.0000%      Two+-000.0000%
Congressional District 29:      White-100.0000%     Black-000.0000%  NativeAm-000.0000%
        Asian-000.0000%       OPI-000.0000%      Two+-000.0000%
Congressional District 30:      White-016.6667%     Black-050.0000%  NativeAm-000.0000%
        Asian-016.6667%       OPI-000.0000%      Two+-016.6667%
```

Table 2:

	20 - 24	25 - 34	35 - 44	45 - 54	54 - 59	60 - 64	65 - 74	74 - 84	85+
CD 1	41700	75078	90390	81448	33203	28500	49169	32608	12612
CD 2	46368	87667	101434	85819	34043	29695	50894	29787	10392
CD 3	51489	146730	158349	112187	35143	22933	29291	15390	4715
CD 4	42207	91233	112284	93762	36137	29632	48615	31033	11711
CD 5	47611	111668	106047	78192	27871	23112	39794	25519	9554
CD 6	45041	119102	141403	110289	35388	23806	32997	18025	5424
CD 7	48350	135572	141313	108376	32855	21367	30400	17087	5666
CD 8	65695	102674	127934	110526	37875	25452	36200	19790	6729
CD 9	41976	89599	105924	87536	29662	23250	39486	24653	7872
CD 10	89498	159373	131395	95956	27053	18784	27950	17248	6346
CD 11	61151	96998	96404	74224	27412	22831	39845	26840	10655
CD 12	48031	102555	106342	80275	28449	22347	36771	24492	8285
CD 13	50072	78837	86194	69251	26456	23149	42150	27593	10868
CD 14	50118	84239	104927	90635	32415	27271	47068	29919	11142
CD 15	60725	114572	104008	80960	28110	24613	45496	27454	8385
CD 16	45778	88410	90753	71237	24442	21174	37310	20195	6055
CD 17	42371	75566	91896	77293	30549	27431	49147	31961	12757
CD 18	48674	102971	95214	73683	25011	20298	32995	18699	6169
CD 19	50567	78773	91704	76031	26408	22479	39273	23800	8102
CD 20	56011	103463	91923	67352	23061	18937	33557	21391	7426
CD 21	41973	106106	134022	111971	39814	32437	56264	36988	12453
CD 22	45161	115721	146176	114167	34465	23691	32980	16486	5175
CD 23	50151	107288	113911	91889	31658	24896	40254	22666	7167
CD 24	53817	109878	105048	78145	26357	19645	30162	18057	6074
CD 25	52711	114822	105518	81981	25291	17895	27949	16973	5370
CD 26	56852	156092	159095	111512	35276	23582	31808	19735	6851
CD 27	47414	90531	94010	78929	27155	23058	41626	24792	7701
CD 28	42879	88349	94125	76755	27724	23214	39408	23546	7315
CD 29	59741	114118	98797	71286	22720	17236	25481	13387	3674
CD 30	55272	114098	95698	69470	24518	18954	28268	15870	5295

Input 2:

```
Congressional District 1: 20to24-41700 25to34-75078 35to44-90390 45to54-81448 55to59-33203 60to64-28500 65to74-49169 74to84-32608
85+-12612
Congressional District 2: 20to24-46368 25to34-87667 35to44-101434 45to54-85819 55to59-34043 60to64-29695 65to74-50894 74to84-29787
85+-10392
Congressional District 3: 20to24-51489 25to34-146730 35to44-158349 45to54-112187 55to59-35143 60to64-22933 65to74-29291 74to84-
15390 85+-4715
Congressional District 4: 20to24-42207 25to34-91233 35to44-112284 45to54-93762 55to59-36137 60to64-29632 65to74-48615 74to84-31033
85+-11711
Congressional District 5: 20to24-47611 25to34-111668 35to44-106047 45to54-78192 55to59-27871 60to64-23112 65to74-39794 74to84-
25519 85+-9554
Congressional District 6: 20to24-45041 25to34-119102 35to44-141403 45to54-110289 55to59-35388 60to64-23806 65to74-32997 74to84-
18025 85+-5424
Congressional District 7: 20to24-48350 25to34-135572 35to44-141313 45to54-108376 55to59-32855 60to64-21367 65to74-30400 74to84-
17087 85+-5666
Congressional District 8: 20to24-65695 25to34-102674 35to44-127934 45to54-110526 55to59-37875 60to64-25452 65to74-36200 74to84-
19790 85+-6729
Congressional District 9: 20to24-41976 25to34-89599 35to44-105924 45to54-87536 55to59-29662 60to64-23250 65to74-39486 74to84-24653
85+-7872
Congressional District 10: 20to24-89498 25to34-159373 35to44-131395 45to54-95956 55to59-27053 60to64-18784 65to74-27950 74to84-
17248 85+-6346
Congressional District 11: 20to24-61151 25to34-96998 35to44-96404 45to54-74224 55to59-27412 60to64-22831 65to74-39845 74to84-26840
85+-10655
Congressional District 12: 20to24-48031 25to34-102555 35to44-106342 45to54-80275 55to59-28449 60to64-22347 65to74-36771 74to84-
24492 85+-8285
Congressional District 13: 20to24-50072 25to34-78837 35to44-86194 45to54-69251 55to59-26456 60to64-23149 65to74-42150 74to84-27593
85+-10868
Congressional District 14: 20to24-50118 25to34-84239 35to44-104927 45to54-90635 55to59-32415 60to64-27271 65to74-47068 74to84-
29919 85+-11142
Congressional District 15: 20to24-60725 25to34-114572 35to44-104008 45to54-80960 55to59-28110 60to64-24613 65to74-45496 74to84-
27454 85+-8385
Congressional District 16: 20to24-45778 25to34-88410 35to44-90753 45to54-71237 55to59-24442 60to64-21174 65to74-37310 74to84-20195
85+-6055
Congressional District 17: 20to24-42371 25to34-75566 35to44-91896 45to54-77293 55to59-30549 60to64-27431 65to74-49147 74to84-31961
85+-12757
Congressional District 18: 20to24-48674 25to34-102971 35to44-95214 45to54-73683 55to59-25011 60to64-20298 65to74-32995 74to84-
18699 85+-6169
Congressional District 19: 20to24-50567 25to34-78773 35to44-91704 45to54-76031 55to59-26408 60to64-22479 65to74-39273 74to84-23800
85+-8102
Congressional District 20: 20to24-56011 25to34-103463 35to44-91923 45to54-67352 55to59-23061 60to64-18937 65to74-33557 74to84-
21391 85+-7426
```

Congressional District 21: 20to24-41973 25to34-106106 35to44-134022 45to54-111971 55to59-39814 60to64-32437 65to74-56264 74to84-36988 85+-12453
Congressional District 22: 20to24-45161 25to34-115721 35to44-146176 45to54-114167 55to59-34465 60to64-23691 65to74-32980 74to84-16486 85+-5175
Congressional District 23: 20to24-50151 25to34-107288 35to44-113911 45to54-91889 55to59-31658 60to64-24896 65to74-40254 74to84-22666 85+-7167
Congressional District 24: 20to24-53817 25to34-109878 35to44-105048 45to54-78145 55to59-26357 60to64-19645 65to74-30162 74to84-18057 85+-6074
Congressional District 25: 20to24-52711 25to34-114822 35to44-105518 45to54-81981 55to59-25291 60to64-17895 65to74-27949 74to84-16973 85+-5370
Congressional District 26: 20to24-56852 25to34-156092 35to44-159095 45to54-111512 55to59-35276 60to64-23582 65to74-31808 74to84-19735 85+-6851
Congressional District 27: 20to24-47414 25to34-90531 35to44-94010 45to54-78929 55to59-27155 60to64-23058 65to74-41626 74to84-24792 85+-7701
Congressional District 28: 20to24-42879 25to34-88349 35to44-94125 45to54-76755 55to59-27724 60to64-23214 65to74-39408 74to84-23546 85+-7315
Congressional District 29: 20to24-59741 25to34-114118 35to44-98797 45to54-71286 55to59-22720 60to64-17236 65to74-25481 74to84-13387 85+-3674
Congressional District 30: 20to24-55272 25to34-114098 35to44-95698 45to54-69470 55to59-24518 60to64-18954 65to74-28268 74to84-15870 85+-5295

Output 2:

```
Congressional District 1:     20to24-009.2857%    25to34-017.6190%    35to44-021.1905%    45to54-018.3333%    55to59-007.1429%
                              60to64-007.1429%    65to74-010.0000%    74to84-007.1429%      85+-002.1429%
Congressional District 2:     20to24-008.8889%    25to34-018.4127%    35to44-023.1746%    45to54-018.4127%    55to59-006.0317%
                              60to64-006.0317%    65to74-011.2698%    74to84-006.0317%      85+-001.7460%
Congressional District 3:     20to24-004.2063%    25to34-028.9683%    35to44-031.5873%    45to54-023.0159%    55to59-004.2063%
                              60to64-002.3016%    65to74-004.2063%    74to84-001.1111%      85+-000.3968%
Congressional District 4:     20to24-007.3016%    25to34-018.9683%    35to44-023.7302%    45to54-019.6825%    55to59-007.3016%
                              60to64-005.1587%    65to74-009.6825%    74to84-005.8730%      85+-002.3016%
Congressional District 5:     20to24-008.4921%    25to34-026.5873%    35to44-023.9683%    45to54-017.0635%    55to59-005.8730%
                              60to64-005.1587%    65to74-006.5873%    74to84-005.1587%      85+-001.1111%
Congressional District 6:     20to24-005.3968%    25to34-024.2063%    35to44-030.3968%    45to54-023.0159%    55to59-004.2063%
                              60to64-003.4921%    65to74-004.2063%    74to84-003.4921%      85+-001.5873%
Congressional District 7:     20to24-003.8889%    25to34-029.3651%    35to44-031.2698%    45to54-023.4127%    55to59-003.8889%
                              60to64-002.6984%    65to74-003.8889%    74to84-000.7937%      85+-000.7937%
Congressional District 8:     20to24-009.6825%    25to34-020.6349%    35to44-026.5873%    45to54-021.3492%    55to59-007.7778%
                              60to64-003.2540%    65to74-007.7778%    74to84-002.5397%      85+-000.3968%
Congressional District 9:     20to24-007.9365%    25to34-021.7460%    35to44-026.2698%    45to54-019.8413%    55to59-006.0317%
                              60to64-004.1270%    65to74-007.2222%    74to84-004.8413%      85+-001.9841%
Congressional District 10:    20to24-011.3492%    25to34-033.7302%    35to44-024.4444%    45to54-013.4921%    55to59-004.6825%
                              60to64-003.2540%    65to74-004.6825%    74to84-003.2540%      85+-001.1111%
Congressional District 11:    20to24-012.6984%    25to34-022.4603%    35to44-022.4603%    45to54-016.9841%    55to59-004.8413%
                              60to64-004.1270%    65to74-008.8889%    74to84-004.8413%      85+-002.6984%
Congressional District 12:    20to24-008.9683%    25to34-025.1587%    35to44-025.8730%    45to54-018.2540%    55to59-005.1587%
                              60to64-005.1587%    65to74-005.8730%    74to84-005.1587%      85+-000.3968%
Congressional District 13:    20to24-011.5079%    25to34-019.8413%    35to44-021.2698%    45to54-017.2222%    55to59-006.0317%
                              60to64-005.3175%    65to74-009.6032%    74to84-006.0317%      85+-003.1746%
Congressional District 14:    20to24-010.5556%    25to34-017.6984%    35to44-024.3651%    45to54-019.6032%    55to59-006.0317%
                              60to64-005.3175%    65to74-009.3651%    74to84-005.3175%      85+-001.7460%
Congressional District 15:    20to24-011.1111%    25to34-025.6349%    35to44-020.1587%    45to54-016.8254%    55to59-005.1587%
                              60to64-004.4444%    65to74-009.2063%    74to84-005.1587%      85+-002.3016%
Congressional District 16:    20to24-009.7619%    25to34-023.5714%    35to44-024.2857%    45to54-019.0476%    55to59-005.9524%
                              60to64-004.0476%    65to74-007.3810%    74to84-004.0476%      85+-001.9048%
Congressional District 17:    20to24-009.6825%    25to34-018.0159%    35to44-021.5873%    45to54-018.0159%    55to59-006.8254%
                              60to64-006.8254%    65to74-009.6825%    74to84-006.8254%      85+-002.5397%
Congressional District 18:    20to24-009.2857%    25to34-026.1905%    35to44-024.7619%    45to54-017.8571%    55to59-005.4762%
                              60to64-004.7619%    65to74-006.1905%    74to84-004.7619%      85+-000.7143%
Congressional District 19:    20to24-011.2698%    25to34-019.6032%    35to44-023.6508%    45to54-018.4127%    55to59-005.3175%
                              60to64-005.3175%    65to74-009.3651%    74to84-005.3175%      85+-001.7460%
Congressional District 20:    20to24-011.4286%    25to34-027.6190%    35to44-023.3333%    45to54-015.7143%    55to59-004.7619%
                              60to64-004.0476%    65to74-008.3333%    74to84-004.0476%      85+-000.7143%
```

```
Congressional District 21:   20to24-006.8254%   25to34-019.4444%   35to44-024.6825%   45to54-021.3492%   55to59-005.6349%
                             60to64-004.9206%   65to74-008.7302%   74to84-005.6349%      85+-002.7778%
Congressional District 22:   20to24-005.3968%   25to34-024.2063%   35to44-030.3968%   45to54-023.0159%   55to59-004.2063%
                             60to64-003.4921%   65to74-004.2063%   74to84-003.4921%      85+-001.5873%
Congressional District 23:   20to24-008.6508%   25to34-024.3651%   35to44-025.5556%   45to54-019.8413%   55to59-004.8413%
                             60to64-004.8413%   65to74-007.4603%   74to84-003.6508%      85+-000.7937%
Congressional District 24:   20to24-009.2857%   25to34-026.1905%   35to44-024.7619%   45to54-017.8571%   55to59-005.4762%
                             60to64-004.7619%   65to74-006.1905%   74to84-004.7619%      85+-000.7143%
Congressional District 25:   20to24-008.1746%   25to34-028.1746%   35to44-024.8413%   45to54-018.6508%   55to59-005.0794%
                             60to64-003.6508%   65to74-006.2698%   74to84-003.6508%      85+-001.5079%
Congressional District 26:   20to24-005.3175%   25to34-028.8889%   35to44-030.7937%   45to54-021.0317%   55to59-005.3175%
                             60to64-002.2222%   65to74-004.1270%   74to84-001.5079%      85+-000.7937%
Congressional District 27:   20to24-010.0000%   25to34-021.1905%   35to44-023.0952%   45to54-019.2857%   55to59-006.1905%
                             60to64-004.7619%   65to74-008.8095%   74to84-004.7619%      85+-001.9048%
Congressional District 28:   20to24-009.0476%   25to34-022.3810%   35to44-023.5714%   45to54-019.0476%   55to59-005.2381%
                             60to64-005.2381%   65to74-007.8571%   74to84-005.2381%      85+-002.3810%
Congressional District 29:   20to24-011.3492%   25to34-029.4444%   35to44-023.0159%   45to54-014.9206%   55to59-005.3968%
                             60to64-004.6825%   65to74-006.1111%   74to84-003.2540%      85+-001.8254%
Congressional District 30:   20to24-010.9524%   25to34-029.0476%   35to44-023.3333%   45to54-015.2381%   55to59-005.0000%
                             60to64-004.2857%   65to74-006.4286%   74to84-003.5714%      85+-002.1429%
```

subGroup.java – Code File 1

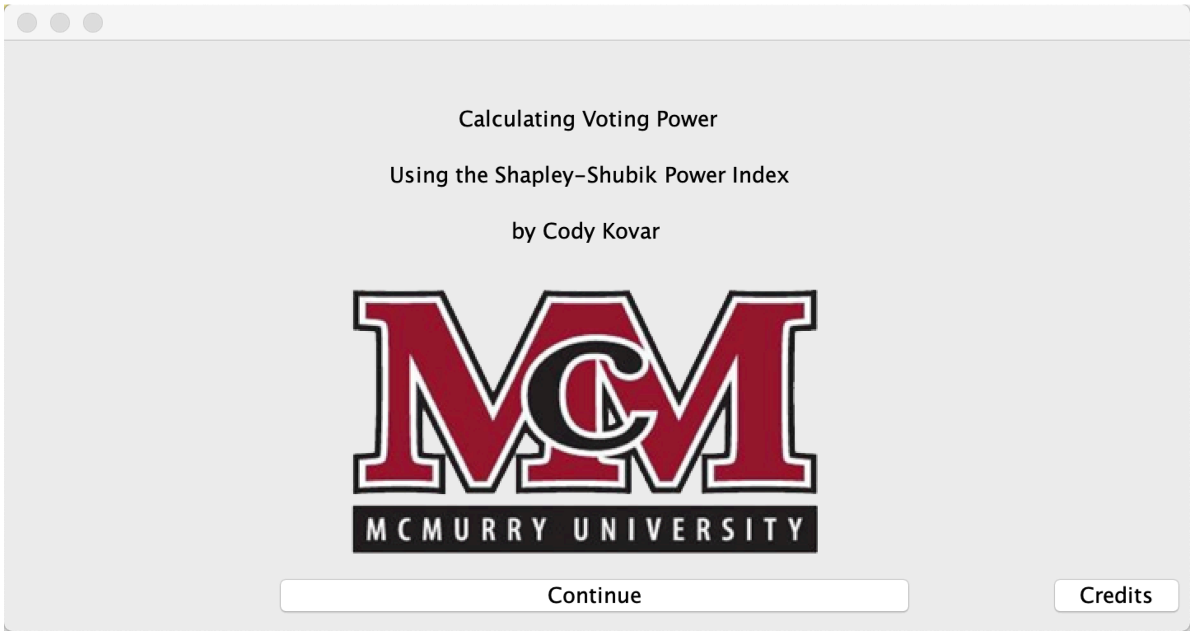
```
/*
Author: Cody Kovar
IDE: NetBeans IDE 8.2
School: McMurry University
Senior Honors Thesis
Creating a Framework for Voting Power

subGroup.java
This file creates the subGroup class. This is needed so that we can store data other than
integers in the list of permutations. The subGroup is complete with all necessary variable
declarations and with all necessary accessors and mutators
*/

package sspicalculator;

public class subGroup
{
    String group_name;
    String region_name;
    double effective_voting_power;
    int population;
    int pivot_count;
    int og_position;

    public subGroup(String gn, int pop, String rn, int o)
    {
        group_name = gn;
        population = pop;
        region_name = rn;
        pivot_count = 0;
        effective_voting_power = 0.0;
        og_position = o;
    }

    subGroup isPivotVoter()
    {
        pivot_count++;
        return this;
    }

    int getPopulation()
    {
        return population;
    }

    String getGroupName()
    {
        return group_name;
    }

    int getPivotCount()
    {
        return pivot_count;
    }

    String getRegionName()
    {
        return region_name;
    }

    void setEffectiveVotingPower(double evp)
    {
        effective_voting_power = evp;
    }

    double getEffectiveVotingPower()
    {
        return effective_voting_power;
    }

    int getOGPosition()
    {
        return og_position;
    }

}
```

subGroupComparator.java – Code File 2

```
/*
Author: Cody Kovar
IDE: NetBeans IDE 8.2
School: McMurry University
Senior Honors Thesis
Creating a Framework for Voting Power

subGroupComparator.java
This file creates the subGroupComparator. This exists solely to sort the subgGroups by the
position in which they were first entered so that the output is sorted in a way that the
user recognizes.
*/

package sspicalculator;
import java.util.Comparator;

public class subGroupComparator implements Comparator<subGroup>
{
    @Override
    public int compare(subGroup sg1, subGroup sg2)
    {
        return sg1.getOGPosition() > sg2.getOGPosition() ? 1 : -1;
    }

}
```

splashScreen.java – Code File 3

```
/*
Author: Cody Kovar
IDE: NetBeans IDE 8.2
School: McMurry University
Senior Honors Thesis
Creating a Framework for Voting Power

SplashScreen.java
This file is just the splash screen per the name. It includes a "credits" button to lead to a
window where I give credit to those who helped make this project possible
*/

package sspicalculator;

public class SplashScreen extends javax.swing.JFrame
{

    public SplashScreen()
    {
        initComponents();
    }

    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN:initComponents
    private void initComponents()
    {

        continueButton = new javax.swing.JButton();
        title1Label = new javax.swing.JLabel();
        title2Label = new javax.swing.JLabel();
        nameLabel = new javax.swing.JLabel();
        creditsButton = new javax.swing.JButton();
        jLabel2 = new javax.swing.JLabel();

        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
        setAutoRequestFocus(false);

        continueButton.setText("Continue");
        continueButton.addActionListener(new java.awt.event.ActionListener()
        {
            public void actionPerformed(java.awt.event.ActionEvent evt)
            {
                continueButtonActionPerformed(evt);
            }
        });

        title1Label.setText("Calculating Voting Power");

        title2Label.setText("Using the Shapley-Shubik Power Index");

        nameLabel.setText("by Cody Kovar");

        creditsButton.setText("Credits");
        creditsButton.addActionListener(new java.awt.event.ActionListener()
        {
            public void actionPerformed(java.awt.event.ActionEvent evt)
            {
                creditsButtonActionPerformed(evt);
            }
        });

        jLabel2.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/sspicalculator/McMLogo.png"))); // NOI18N

        javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(161, 161, 161)
                .addComponent(continueButton, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                .addGap(75, 75, 75)
                .addComponent(creditsButton))
            .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addGroup(layout.createSequentialGroup()
                        .addGap(276, 276, 276)
                        .addComponent(title1Label))
                    .addGroup(layout.createSequentialGroup()
                        .addGap(234, 234, 234)
                        .addComponent(title2Label)))
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
            .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addGroup(layout.createSequentialGroup()
                        .addGap(308, 308, 308)
                        .addComponent(nameLabel))
                    .addGroup(layout.createSequentialGroup()
                        .addGap(197, 197, 197)
                        .addComponent(jLabel2)))
                .addGap(0, 210, Short.MAX_VALUE))
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
                .addGap(39, 39, 39)
                .addComponent(title1Label)
                .addGap(18, 18, 18)
                .addComponent(title2Label)
                .addGap(18, 18, 18)
                .addComponent(nameLabel)
                .addGap(18, 18, 18)
                .addComponent(jLabel2)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(continueButton)
                    .addComponent(creditsButton))
                .addContainerGap())
        );

        pack();
    }// </editor-fold>//GEN-END:initComponents
    private void creditsButtonActionPerformed(java.awt.event.ActionEvent evt)
    {//GEN-FIRST:event_creditsButtonActionPerformed
        creditsScreen letsGo = new creditsScreen();
        letsGo.setTitle("Credits Screen");
        letsGo.creditsTextArea.append("Credits\n\n");
        letsGo.creditsTextArea.append("Dr. Paul Fabrizio, McMurry University\n");
        letsGo.creditsTextArea.append("Professor of Political Science\n\n");
        letsGo.creditsTextArea.append("Dr. Mark Thornburg, McMurry University\n");
        letsGo.creditsTextArea.append("Professor of Mathematics\n\n");
        letsGo.creditsTextArea.append("Mr. Rich Brozovic, McMurry University\n");
        letsGo.creditsTextArea.append("Professor of Computer Science\n\n");
        letsGo.creditsTextArea.append("Dr. Tina Bertrand, McMurry University\n");
        letsGo.creditsTextArea.append("Professor of Political Science\n\n");
        letsGo.creditsTextArea.setEditable(false);
        letsGo.setVisible(true);
    }//GEN-LAST:event_creditsButtonActionPerformed

    private void continueButtonActionPerformed(java.awt.event.ActionEvent evt)
    {//GEN-FIRST:event_continueButtonActionPerformed
        entryTypeScreen letsGo = new entryTypeScreen(null, true);
        letsGo.setTitle("Entry Type Screen");
        dispose();
        letsGo.setVisible(true);
    }//GEN-LAST:event_continueButtonActionPerformed

    public static void main(String args[])
    {
        java.awt.EventQueue.invokeLater(new Runnable()
        {
            public void run()
            {
                new SplashScreen().setVisible(true);
            }
        });
    }

    // Variables declaration - do not modify//GEN-BEGIN:variables
    private javax.swing.JButton continueButton;
    private javax.swing.JButton creditsButton;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel nameLabel;
    private javax.swing.JLabel title1Label;
    private javax.swing.JLabel title2Label;
    // End of variables declaration//GEN-END:variables
}
```


Calculating Voting Power

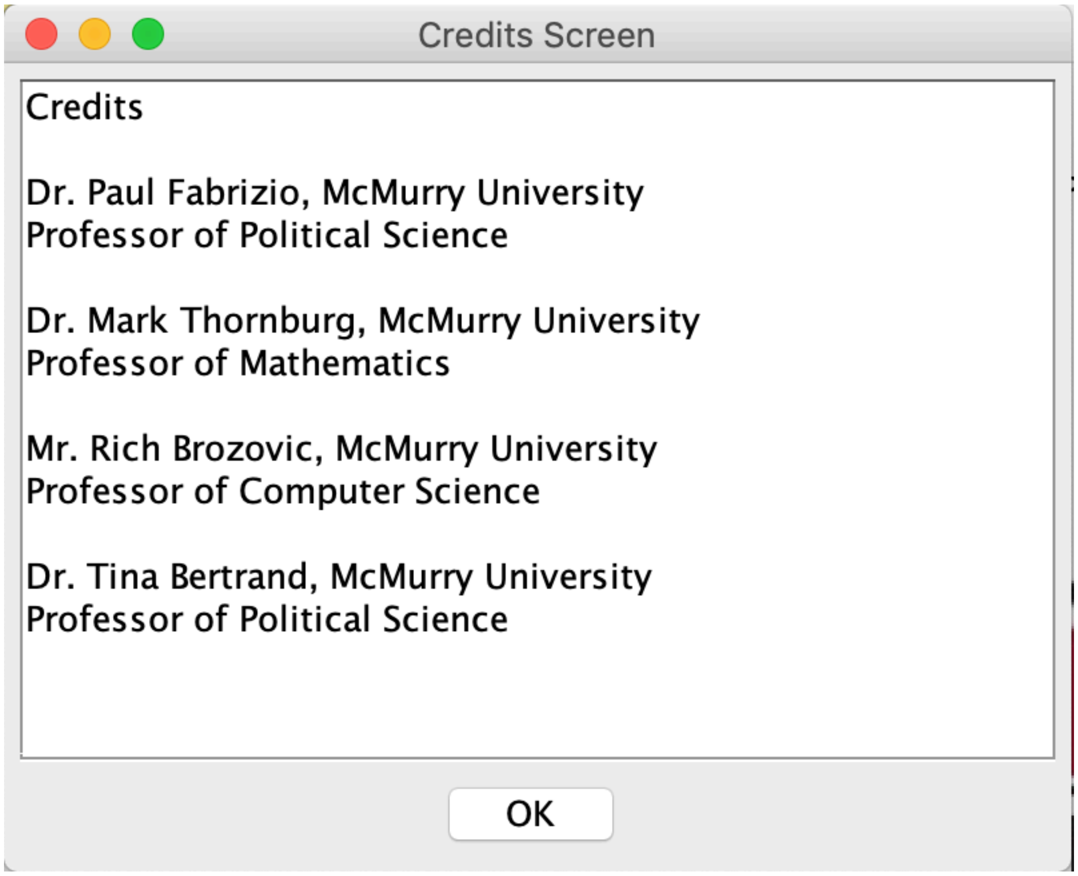
Using the Shapley-Shubik Power Index

by Cody Kovar

creditsScreen.java – Code File 4

```
/*
Author: Cody Kovar
IDE: NetBeans IDE 8.2
School: McMurry University
Senior Honors Thesis
Creating a Framework for Voting Power

creditsScreen.java
This file creates the creditsScreen window where credit is given to those who helped make
this project possible.
*/

package sspicalculator;

public class creditsScreen extends javax.swing.JFrame
{

    public creditsScreen()
    {
        initComponents();
    }

    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN:initComponents
    private void initComponents()
    {

        jScrollPane1 = new javax.swing.JScrollPane();
        creditsTextArea = new javax.swing.JTextArea();
        okButton = new javax.swing.JButton();

        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);

        creditsTextArea.setColumns(20);
        creditsTextArea.setRows(5);
        jScrollPane1.setViewportView(creditsTextArea);

        okButton.setText("OK");
        okButton.addActionListener(new java.awt.event.ActionListener()
        {
            public void actionPerformed(java.awt.event.ActionEvent evt)
            {
                okButtonActionPerformed(evt);
            }
        });

        javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jScrollPane1)
                .addContainerGap())
            .addGroup(layout.createSequentialGroup()
                .addGap(160, 160, 160)
                .addComponent(okButton, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
```

```
                .addGap(165, 165, 165))
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED_SIZE, 256,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(okButton)
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
        );

        pack();
    }// </editor-fold>//GEN-END:initComponents

    private void okButtonActionPerformed(java.awt.event.ActionEvent evt)
    {//GEN-FIRST:event_okButtonActionPerformed
        dispose();
    }//GEN-LAST:event_okButtonActionPerformed

    public static void main(String args[])
    {
        java.awt.EventQueue.invokeLater(new Runnable()
        {
            public void run()
            {
                new creditsScreen().setVisible(true);
            }
        });
    }

    // Variables declaration - do not modify//GEN-BEGIN:variables
    public javax.swing.JTextArea creditsTextArea;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JButton okButton;
    // End of variables declaration//GEN-END:variables
} /*
Author: Cody Kovar
IDE: NetBeans IDE 8.2
School: McMurry University
Senior Honors Thesis
Creating a Framework for Voting Power

creditsScreen.java
This file creates the creditsScreen window where credit is given to those who helped make
this project possible.
*/

package sspicalculator;

public class creditsScreen extends javax.swing.JFrame
{

    public creditsScreen()
    {
        initComponents();
    }

    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN:initComponents
    private void initComponents()
    {

        jScrollPane1 = new javax.swing.JScrollPane();
        creditsTextArea = new javax.swing.JTextArea();
        okButton = new javax.swing.JButton();

        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);

        creditsTextArea.setColumns(20);
        creditsTextArea.setRows(5);
        jScrollPane1.setViewportView(creditsTextArea);

        okButton.setText("OK");
        okButton.addActionListener(new java.awt.event.ActionListener()
        {
            public void actionPerformed(java.awt.event.ActionEvent evt)
            {
                okButtonActionPerformed(evt);
            }
        });

        javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jScrollPane1)
                .addContainerGap())
            .addGroup(layout.createSequentialGroup()
                .addGap(160, 160, 160)
                .addComponent(okButton, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                .addGap(165, 165, 165))
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED_SIZE, 256,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(okButton)
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
        );

        pack();
    }// </editor-fold>//GEN-END:initComponents

    private void okButtonActionPerformed(java.awt.event.ActionEvent evt)
    {//GEN-FIRST:event_okButtonActionPerformed
        dispose();
    }//GEN-LAST:event_okButtonActionPerformed

    public static void main(String args[])
    {
        java.awt.EventQueue.invokeLater(new Runnable()
        {
            public void run()
            {
                new creditsScreen().setVisible(true);
            }
        });
    }

    // Variables declaration - do not modify//GEN-BEGIN:variables
    public javax.swing.JTextArea creditsTextArea;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JButton okButton;
    // End of variables declaration//GEN-END:variables
}
```



Credits

Dr. Paul Fabrizio, McMurry University
Professor of Political Science

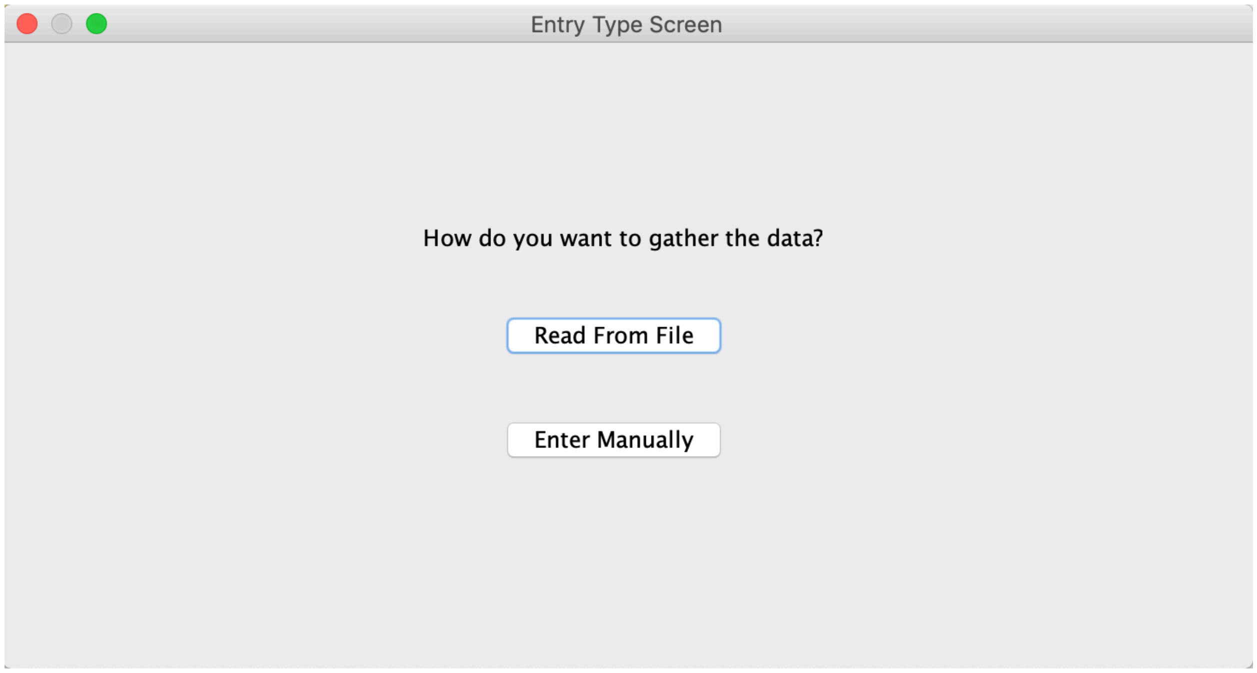
Dr. Mark Thornburg, McMurry University
Professor of Mathematics

Mr. Rich Brozovic, McMurry University
Professor of Computer Science

Dr. Tina Bertrand, McMurry University
Professor of Political Science

entryTypeScreen.java – Code File 5

```
/*
Author: Cody Kovar
IDE: NetBeans IDE 8.2
School: McMurry University
Senior Honors Thesis
Creating a Framework for Voting Power

entryTypeScreen.java
This file creates the entryTypeScreen window where the user can choose whether to
enter the data manually or to read in from a text file, given that it has the correct format
*/

package sspicalculator;

public class entryTypeScreen extends javax.swing.JDialog
{

    public entryTypeScreen(java.awt.Frame parent, boolean modal)
    {
        super(parent, modal);
        initComponents();
    }

    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN:initComponents
    private void initComponents()
    {

        readFromFileButton = new javax.swing.JButton();
        enterManuallyButton = new javax.swing.JButton();
        titleLabel = new javax.swing.JLabel();

        setDefaultCloseOperation(javax.swing.WindowConstants.DISPOSE_ON_CLOSE);

        readFromFileButton.setText("Read From File");
        readFromFileButton.addActionListener(new java.awt.event.ActionListener()
        {
            public void actionPerformed(java.awt.event.ActionEvent evt)
            {
                readFromFileButtonActionPerformed(evt);
            }
        });

        enterManuallyButton.setText("Enter Manually");
        enterManuallyButton.addActionListener(new java.awt.event.ActionListener()
        {
            public void actionPerformed(java.awt.event.ActionEvent evt)
            {
                enterManuallyButtonActionPerformed(evt);
            }
        });

        titleLabel.setText("How do you want to gather the data?");

        javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addGroup(layout.createSequentialGroup()
                        .addGap(284, 284, 284)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                            .addComponent(enterManuallyButton)
                            .addComponent(readFromFileButton)))
                    .addGroup(layout.createSequentialGroup()
                        .addGap(242, 242, 242)
                        .addComponent(titleLabel)))
                .addContainerGap(248, Short.MAX_VALUE))
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(104, 104, 104)
                .addComponent(titleLabel)
                .addGap(35, 35, 35)
                .addComponent(readFromFileButton)
                .addGap(31, 31, 31)
                .addComponent(enterManuallyButton)
                .addContainerGap(116, Short.MAX_VALUE))
        );

        pack();
    }// </editor-fold>//GEN-END:initComponents

    private void readFromFileButtonActionPerformed(java.awt.event.ActionEvent evt)
    {//GEN-FIRST:event_readFromFileButtonActionPerformed
         readDataFromFile letsGo = new readDataFromFile();
         letsGo.setTitle("From File Input Screen");
         dispose();
         letsGo.setVisible(true);
    }//GEN-LAST:event_readFromFileButtonActionPerformed

    private void enterManuallyButtonActionPerformed(java.awt.event.ActionEvent evt)
    {//GEN-FIRST:event_enterManuallyButtonActionPerformed
        manualInputScreen letsGo = new manualInputScreen(null, true);
        letsGo.setTitle("Manual Input Screen");
        dispose();
        letsGo.setVisible(true);
    }//GEN-LAST:event_enterManuallyButtonActionPerformed

    public static void main(String args[])
    {
        java.awt.EventQueue.invokeLater(new Runnable()
        {
            public void run()
            {
                entryTypeScreen dialog = new entryTypeScreen(new javax.swing.JFrame(), true);
                dialog.addWindowListener(new java.awt.event.WindowAdapter()
                {
                    @Override
                    public void windowClosing(java.awt.event.WindowEvent e)
                    {
                        System.exit(0);
                    }
                });
                dialog.setVisible(true);
            }
        });
    }

    // Variables declaration - do not modify//GEN-BEGIN:variables
    private javax.swing.JButton enterManuallyButton;
    private javax.swing.JButton readFromFileButton;
    private javax.swing.JLabel titleLabel;
    // End of variables declaration//GEN-END:variables
}
```

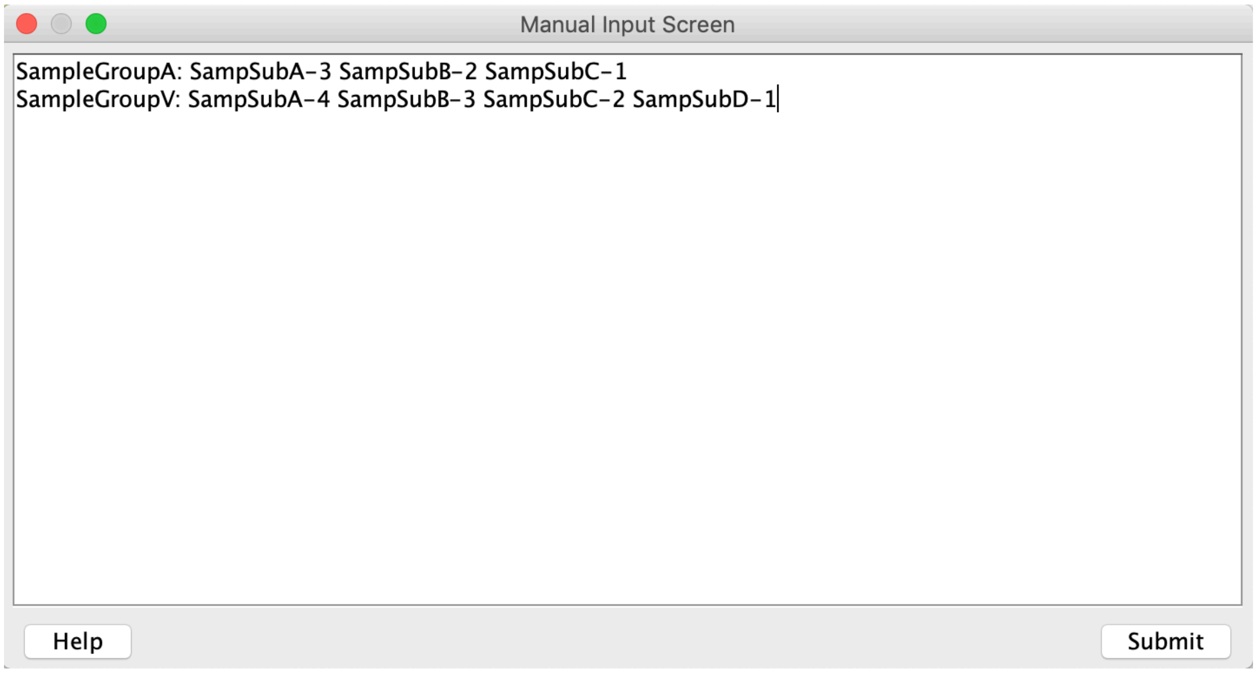


How do you want to gather the data?

manualEntryScreen.java – Code File 6

```
/*
Author: Cody Kovar
IDE: NetBeans IDE 8.2
School: McMurry University
Senior Honors Thesis
Creating a Framework for Voting Power

manualInputScreen.java
This file is where the user will manually input the data to calculate the voting power of the
given groups. If the user chooses to read in from a file, this file we open again and the
data from the file will automatically populate the text feild.

The file also has a help button which is what users will click when they need to know the
format in which to enter the data manually.
*/

package sspicalculator;
import java.util.*;

public class manualInputScreen extends javax.swing.JDialog
{

    public manualInputScreen(java.awt.Frame parent, boolean modal)
    {
        super(parent, modal);
        initComponents();
    }

    //Auto-Generated Code
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN:initComponents
    private void initComponents()
    {

        helpButton = new javax.swing.JButton();
        submitButton = new javax.swing.JButton();
        jScrollPane1 = new javax.swing.JScrollPane();
        inputTextArea = new javax.swing.JTextArea();

        setDefaultCloseOperation(javax.swing.WindowConstants.DISPOSE_ON_CLOSE);

        helpButton.setText("Help");
        helpButton.addActionListener(new java.awt.event.ActionListener()
        {
            public void actionPerformed(java.awt.event.ActionEvent evt)
            {
                helpButtonActionPerformed(evt);
            }
        });

        submitButton.setText("Submit");
        submitButton.addActionListener(new java.awt.event.ActionListener()
        {
            public void actionPerformed(java.awt.event.ActionEvent evt)
            {
                submitButtonActionPerformed(evt);
            }
        });
        inputTextArea.setColumns(20);
        inputTextArea.setRows(5);
        jScrollPane1.setViewportView(inputTextArea);

        javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(jScrollPane1, javax.swing.GroupLayout.DEFAULT_SIZE, 708,
Short.MAX_VALUE)
                    .addGroup(layout.createSequentialGroup()
                        .addComponent(helpButton)
                        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                        .addComponent(submitButton)))
                .addContainerGap())
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jScrollPane1, javax.swing.GroupLayout.DEFAULT_SIZE, 319,
Short.MAX_VALUE)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(submitButton)
                    .addComponent(helpButton)))
        );

        pack();
    }// </editor-fold>//GEN-END:initComponents

    private void submitButtonActionPerformed(java.awt.event.ActionEvent evt)
    {//GEN-FIRST:event_submitButtonActionPerformed
        resultsScreen letsGo = new resultsScreen(null, true);
        letsGo.setTitle("Results Screen");
        ArrayList<ArrayList<subGroup>> input_datas =
          new ArrayList<ArrayList<subGroup>>();
        ArrayList<subGroup> input_data = new ArrayList<subGroup>();
        String[] all_line = inputTextArea.getText().split("\n");
        for(int i = 0; i < all_line.length; i++)
        {
            String sans_title = all_line[i].split(":")[1].trim();
            String[] all_groups = sans_title.split(" ");
            for(int k = 0; k < all_groups.length; k++)
            {
                input_data.add(new subGroup((all_groups[k].split("-")[0]),
                Integer.parseInt(all_groups[k].split("-")[1]),
                all_line[i].split(":")[0], k));
            }

            input_datas.add(input_data);
            input_data = new ArrayList<subGroup>();
        }
        int fact = 0;

        for(int c = 0; c < input_datas.size(); c++)
        {
            fact = input_datas.get(c).size(); //Set Factorial
            for(int i = fact - 1; i > 1; i--)
            fact *= i;

            ArrayList<ArrayList<subGroup>> all_groups =
              manualPermutations(input_datas.get(c));
            input_datas.set(c, getPivotVoter(all_groups,
              getPivotVoteCount(input_datas.get(c))));

            // Set effective voting power loop
            for(int i = 0; i < input_datas.get(c).size(); i++)
                input_datas.get(c).get(i).setEffectiveVotingPower(
                  Math.round(input_datas.get(c).get(i).getPivotCount()/
                  (double)fact*1000000)/10000.0);

            Collections.sort(input_datas.get(c), new subGroupComparator());

            String line = String.format("%25s:",
              input_datas.get(c).get(0).getRegionName());
            for(int i = 0; i < input_datas.get(c).size(); i++)
            {
                if(i < input_datas.get(c).size() - 1)
                    line = line + String.format("%10s-%08.4f%%",
                      input_datas.get(c).get(i).getGroupName(),
                      input_datas.get(c).get(i).getEffectiveVotingPower());

                else
                    line = line + String.format("%10s-%08.4f%%\n",
                      input_datas.get(c).get(i).getGroupName(),
                      input_datas.get(c).get(i).getEffectiveVotingPower());
            }

            letsGo.resultsTextArea.append(line);
          }
          letsGo.resultsTextArea.setEditable(false);
          dispose();
          letsGo.setVisible(true);
    }//GEN-LAST:event_submitButtonActionPerformed

    private void helpButtonActionPerformed(java.awt.event.ActionEvent evt)
    {//GEN-FIRST:event_helpButtonActionPerformed
        helpScreen letsGo = new helpScreen();
        letsGo.setTitle("Help Screen");
        letsGo.helpTextArea.append("The format needed to fuction properly is:\n\n");
        letsGo.helpTextArea.append("GroupName: x1-pop x2-pop xn-pop\n\n");
        letsGo.helpTextArea.append("Where group name is the name of the group of
subgroups\n");
        letsGo.helpTextArea.append("Where xn is the name of the subgroup\n");
        letsGo.helpTextArea.append("Where pop is the population of the subgroup\n");
        letsGo.helpTextArea.setEditable(false);
        letsGo.setAlwaysOnTop(true);
        letsGo.setVisible(true);
    }//GEN-LAST:event_helpButtonActionPerformed

    public static void main(String args[])
    {
        java.awt.EventQueue.invokeLater(new Runnable()
        {
            public void run()
            {
                manualInputScreen dialog =
                  new manualInputScreen(new javax.swing.JFrame(), true);
                dialog.addWindowListener(new java.awt.event.WindowAdapter()
                {

                    @Override
                    public void windowClosing(java.awt.event.WindowEvent e)
                    {
                        System.exit(0);
                    }
                });
                dialog.setVisible(true);
            }
        });
    }

    public static ArrayList<subGroup> getPivotVoter(ArrayList<ArrayList<subGroup>>
subGroup_List, int pivotVoteCount)
    {
        int total_votes = 0;
        ArrayList<subGroup> pvc = new ArrayList<subGroup>();
        pvc = subGroup_List.get(0);
        for(int i = 0; i < subGroup_List.size(); i++)
        {
            total_votes = 0;
            for(int j = 0; j < subGroup_List.get(i).size(); j++)
            {
                total_votes += subGroup_List.get(i).get(j).getPopulation();
                if(total_votes >= pivotVoteCount)
                {
                    pvc.set(pvc.lastIndexOf(subGroup_List.get(i).get(j)),
                      subGroup_List.get(i).get(j).isPivotVoter());
                    break;
                }
            }
        }
        return pvc;
    }

    public static int getPivotVoteCount(ArrayList<subGroup> subGroup_List)
    {
        double pivotVoteCount = 0;
        for(subGroup g: subGroup_List)
            pivotVoteCount += g.getPopulation();

        pivotVoteCount /= 2;
        pivotVoteCount = Math.ceil(pivotVoteCount);
        pivotVoteCount++;

        return (int)pivotVoteCount;
    }

    public static ArrayList<ArrayList<subGroup>> manualPermutations(ArrayList<subGroup>
input_list)
    {
            ArrayList<ArrayList<subGroup>> all_permutations =
        new ArrayList<ArrayList<subGroup>>();
       all_permutations.add(new ArrayList<subGroup>());
```

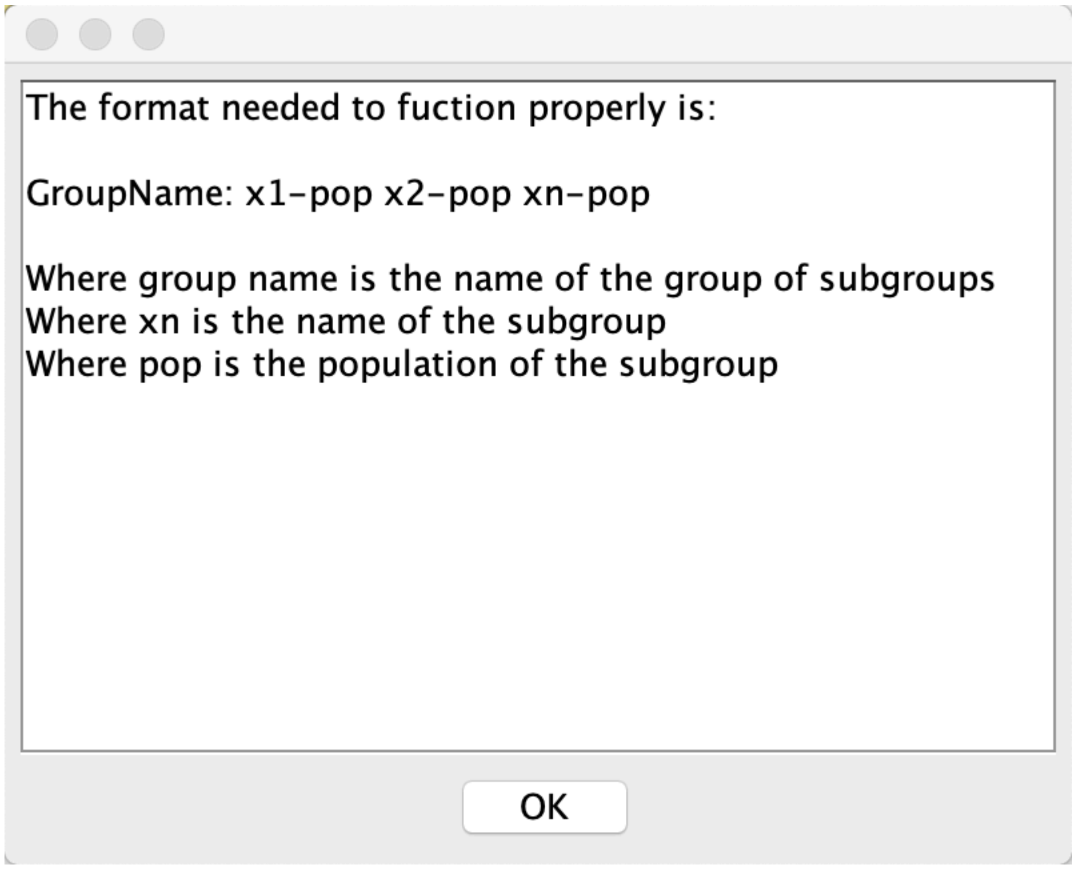
```
            for (int i = 0; i < input_list.size(); i++)
            {
               ArrayList<ArrayList<subGroup>> current_build =
            new ArrayList<ArrayList<subGroup>>();

               for (ArrayList<subGroup> sg : all_permutations)
               {
                   for (int j = 0; j < sg.size()+1; j++)
                   {
                       sg.add(j, input_list.get(i));
                       ArrayList<subGroup> temp = new ArrayList<subGroup>(sg);
                       current_build.add(temp);
                       sg.remove(j);
                   }
               }
               all_permutations = new ArrayList<ArrayList<subGroup>>(current_build);
           }

           return all_permutations;

    }

    // Variables declaration - do not modify//GEN-BEGIN:variables
    private javax.swing.JButton helpButton;
    public javax.swing.JTextArea inputTextArea;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JButton submitButton;
    // End of variables declaration//GEN-END:variables
}
```



```
SampleGroupA: SampSubA-3 SampSubB-2 SampSubC-1
SampleGroupV: SampSubA-4 SampSubB-3 SampSubC-2 SampSubD-1
```

helpScreen.java – Code File 7

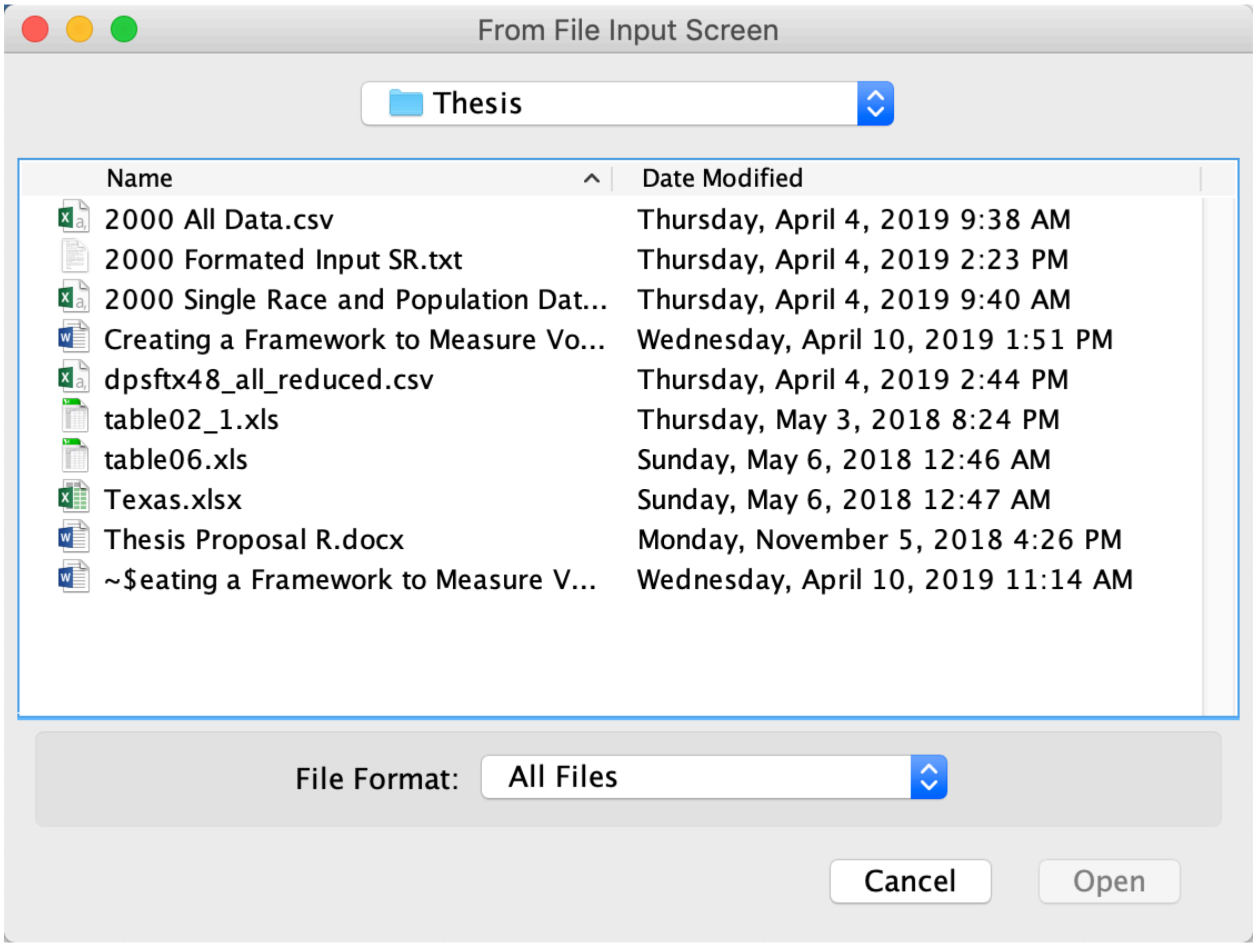
```
/*
Author: Cody Kovar
IDE: NetBeans IDE 8.2
School: McMurry University
Senior Honors Thesis
Creating a Framework for Voting Power

helpScreen.java
This file creates the helpScreen window where the format for input is shown.
*/

package sspicalculator;

public class helpScreen extends javax.swing.JFrame
{

    public helpScreen()
    {
        initComponents();
    }

    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN:initComponents
    private void initComponents()
    {

        jScrollPane1 = new javax.swing.JScrollPane();
        helpTextArea = new javax.swing.JTextArea();
        okButton = new javax.swing.JButton();

        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);

        helpTextArea.setColumns(20);
        helpTextArea.setRows(5);
        jScrollPane1.setViewportView(helpTextArea);

        okButton.setText("OK");
        okButton.addActionListener(new java.awt.event.ActionListener()
        {
            public void actionPerformed(java.awt.event.ActionEvent evt)
            {
                okButtonActionPerformed(evt);
            }
        });

        javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jScrollPane1, javax.swing.GroupLayout.DEFAULT_SIZE, 388, 
Short.MAX_VALUE)
                .addContainerGap())
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, 
layout.createSequentialGroup()
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                .addComponent(okButton)
                .addGap(160, 160, 160))
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED_SIZE, 253,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(okButton)
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
        );

        pack();
    }// </editor-fold>//GEN-END:initComponents

    private void okButtonActionPerformed(java.awt.event.ActionEvent evt)
    {//GEN-FIRST:event_okButtonActionPerformed
        dispose();
    }//GEN-LAST:event_okButtonActionPerformed

    public static void main(String args[])
    {

        java.awt.EventQueue.invokeLater(new Runnable()
        {
            public void run() {
                new helpScreen().setVisible(true);
            }
        });
    }

    // Variables declaration - do not modify//GEN-BEGIN:variables
    public javax.swing.JTextArea helpTextArea;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JButton okButton;
    // End of variables declaration//GEN-END:variables
}
```



```
The format needed to fuction properly is:

GroupName: x1-pop x2-pop xn-pop

Where group name is the name of the group of subgroups
Where xn is the name of the subgroup
Where pop is the population of the subgroup
```

readDataFromFile.java – Code File 8

```
/*
Author: Cody Kovar
IDE: NetBeans IDE 8.2
School: McMurry University
Senior Honors Thesis
Creating a Framework for Voting Power

readDataFromFile.java
This file creates the readDataFromFile window where the user chooses what text file to read
and then places the data in the manualInputScreen window.
*/

package sspicalculator;
import java.io.*;
import java.util.*;

public class readDataFromFile extends javax.swing.JFrame
{

    public readDataFromFile()
    {
        initComponents();
    }

    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN:initComponents
    private void initComponents()
    {

        fileChooser = new javax.swing.JFileChooser();

        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);

        fileChooser.addActionListener(new java.awt.event.ActionListener()
        {
            public void actionPerformed(java.awt.event.ActionEvent evt)
            {
                try
                {
                  if(evt.getActionCommand().equals("ApproveSelection"))
                  {
                    File input_file = fileChooser.getSelectedFile();
                    Scanner input = new Scanner(input_file);
                    manualInputScreen letsGo = new manualInputScreen(null, true);
                    letsGo.setTitle("Manual Input Screen");
                    while(input.hasNextLine())
                    {
                      letsGo.inputTextArea.append(input.nextLine() + "\n");
                    }
                    input.close();
                    dispose();
                    letsGo.setVisible(true);
                    if(evt.getActionCommand().equals("CancelSelection"))
                      dispose();
                    }
                }
                catch (IOException ie)
                {
                  System.out.println("Input File cannot be opened... Please try again");
                }
          }
        });

        javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                .addComponent(fileChooser, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
                .addContainerGap())
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addComponent(fileChooser, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(0, 0, Short.MAX_VALUE))
        );

        pack();
    }// </editor-fold>//GEN-END:initComponents

    public static void main(String args[])
    {

        java.awt.EventQueue.invokeLater(new Runnable()
        {
            public void run()
            {
                new readDataFromFile().setVisible(true);
            }
        });
    }

    // Variables declaration - do not modify//GEN-BEGIN:variables
    private javax.swing.JFileChooser fileChooser;
    // End of variables declaration//GEN-END:variables
}
```

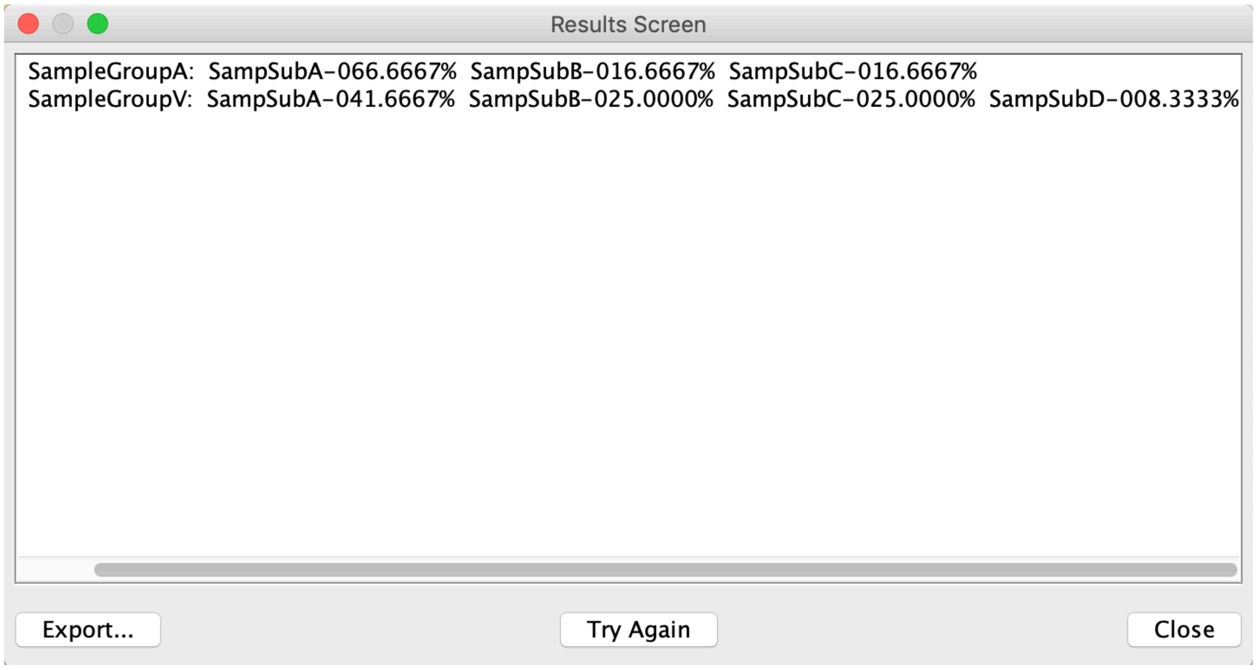


Name	Date Modified
2000 All Data.csv	Thursday, April 4, 2019 9:38 AM
2000 Formated Input SR.txt	Thursday, April 4, 2019 2:23 PM
2000 Single Race and Population Dat...	Thursday, April 4, 2019 9:40 AM
Creating a Framework to Measure Vo...	Wednesday, April 10, 2019 1:51 PM
dpsftx48_all_reduced.csv	Thursday, April 4, 2019 2:44 PM
table02_1.xls	Thursday, May 3, 2018 8:24 PM
table06.xls	Sunday, May 6, 2018 12:46 AM
Texas.xlsx	Sunday, May 6, 2018 12:47 AM
Thesis Proposal R.docx	Monday, November 5, 2018 4:26 PM
~$eating a Framework to Measure V...	Wednesday, April 10, 2019 11:14 AM

File Format: All Files

resultsScreen.java – Code File 9

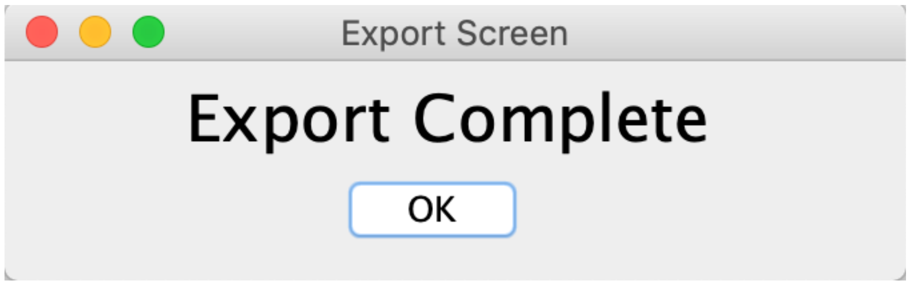
```
/*
Author: Cody Kovar
IDE: NetBeans IDE 8.2
School: McMurry University
Senior Honors Thesis
Creating a Framework for Voting Power

resultsScreen.java
This file creates the resultsScreen window where the calculated data appears on the screen.
The user also has the opportunity to try again with new data or to save the data calculated.
*/

package sspicalculator;
import java.io.*;

public class resultsScreen extends javax.swing.JDialog
{

    public resultsScreen(java.awt.Frame parent, boolean modal)
    {
        super(parent, modal);
        initComponents();
    }

    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN:initComponents
    private void initComponents()
    {

        exportButton = new javax.swing.JButton();
        tryAgainButton = new javax.swing.JButton();
        closeButton = new javax.swing.JButton();
        jScrollPane1 = new javax.swing.JScrollPane();
        resultsTextArea = new javax.swing.JTextArea();

        setDefaultCloseOperation(javax.swing.WindowConstants.DISPOSE_ON_CLOSE);

        exportButton.setText("Export...");
        exportButton.addActionListener(new java.awt.event.ActionListener()
        {
            public void actionPerformed(java.awt.event.ActionEvent evt)
            {
                exportButtonActionPerformed(evt);
            }
        });

        tryAgainButton.setText("Try Again");
        tryAgainButton.addActionListener(new java.awt.event.ActionListener()
        {
            public void actionPerformed(java.awt.event.ActionEvent evt)
            {
                tryAgainButtonActionPerformed(evt);
            }
        });

        closeButton.setText("Close");
        closeButton.addActionListener(new java.awt.event.ActionListener()
        {
            public void actionPerformed(java.awt.event.ActionEvent evt)
            {
                closeButtonActionPerformed(evt);
            }
        });

        resultsTextArea.setColumns(20);
        resultsTextArea.setRows(5);
        jScrollPane1.setViewportView(resultsTextArea);

        javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addComponent(exportButton)
                .addGap(217, 217, 217)
                .addComponent(tryAgainButton)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 223,
Short.MAX_VALUE)
                .addComponent(closeButton))
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jScrollPane1)
                .addContainerGap())
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jScrollPane1, javax.swing.GroupLayout.DEFAULT_SIZE, 307,
Short.MAX_VALUE)
                .addGap(12, 12, 12)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(exportButton)
                    .addComponent(tryAgainButton)
                    .addComponent(closeButton))
                .addContainerGap())
        );

        pack();
    }// </editor-fold>//GEN-END:initComponents

    private void tryAgainButtonActionPerformed(java.awt.event.ActionEvent evt)
    {//GEN-FIRST:event_tryAgainButtonActionPerformed
        entryTypeScreen letsGo = new entryTypeScreen(null, true);
        letsGo.setTitle("Splash Screen");
        dispose();
        letsGo.setVisible(true);
    }//GEN-LAST:event_tryAgainButtonActionPerformed

    private void closeButtonActionPerformed(java.awt.event.ActionEvent evt)
    {//GEN-FIRST:event_closeButtonActionPerformed
        dispose();
    }//GEN-LAST:event_closeButtonActionPerformed

    private void exportButtonActionPerformed(java.awt.event.ActionEvent evt)
    {//GEN-FIRST:event_exportButtonActionPerformed
        try
        {
            String[] all_line = resultsTextArea.getText().split("\n");
            File output_file = new File("output.txt");
            PrintWriter output_writer = new PrintWriter(output_file);
            for(String line: all_line)
            {
                output_writer.write(line + "\n");
                output_writer.flush();
            }
            output_writer.close();
            exportComplete letsGo = new exportComplete();
            letsGo.setTitle("Export Screen");
            letsGo.setVisible(true);
        }
        catch (IOException io)
        {
            exportComplete letsGo = new exportComplete();
            letsGo.setTitle("Export Screen");
            letsGo.exportCompleteLabel.setText("Export Failed");
            letsGo.setVisible(true);
        }

    }//GEN-LAST:event_exportButtonActionPerformed

    public static void main(String args[])
    {
        java.awt.EventQueue.invokeLater(new Runnable()
        {
            public void run() {
                resultsScreen dialog = new resultsScreen(new javax.swing.JFrame(), true);
                dialog.addWindowListener(new java.awt.event.WindowAdapter()
                {
                    @Override
                    public void windowClosing(java.awt.event.WindowEvent e)
                    {
                        System.exit(0);
                    }
                });
                dialog.setVisible(true);
            }
        });
    }

    // Variables declaration - do not modify//GEN-BEGIN:variables
    private javax.swing.JButton closeButton;
    private javax.swing.JButton exportButton;
    private javax.swing.JScrollPane jScrollPane1;
    public javax.swing.JTextArea resultsTextArea;
    private javax.swing.JButton tryAgainButton;
    // End of variables declaration//GEN-END:variables
}
```



```
SampleGroupA:  SampSubA-066.6667%  SampSubB-016.6667%  SampSubC-016.6667%
SampleGroupV:  SampSubA-041.6667%  SampSubB-025.0000%  SampSubC-025.0000%  SampSubD-008.3333%
```

exportComplete.java – Code File 10

```
/*
Author: Cody Kovar
IDE: NetBeans IDE 8.2
School: McMurry University
Senior Honors Thesis
Creating a Framework for Voting Power

exportComplete.java
This file creates the exportComplete window in which the user is notified of the successful
creation of the export file.
*/

package sspicalculator;

public class exportComplete extends javax.swing.JFrame
{

    public exportComplete()
    {
        initComponents();
    }

    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN:initComponents
    private void initComponents()
    {

        exportCompleteLabel = new javax.swing.JLabel();
        okButton = new javax.swing.JButton();

        setDefaultCloseOperation(javax.swing.WindowConstants.DISPOSE_ON_CLOSE);

        exportCompleteLabel.setFont(new java.awt.Font("Lucida Grande", 0, 24)); // NOI18N
        exportCompleteLabel.setText("Export Complete");

        okButton.setText("OK");
        okButton.addActionListener(new java.awt.event.ActionListener()
        {
            public void actionPerformed(java.awt.event.ActionEvent evt)
            {
                okButtonActionPerformed(evt);
            }
        });

        javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addGroup(layout.createSequentialGroup()
                        .addGap(68, 68, 68)
                        .addComponent(exportCompleteLabel))
                    .addGroup(layout.createSequentialGroup()
                        .addGap(123, 123, 123)
                        .addComponent(okButton)))
                .addContainerGap(74, Short.MAX_VALUE))
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(exportCompleteLabel)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(okButton)
                .addContainerGap(11, Short.MAX_VALUE))
        );

        pack();
    }// </editor-fold>//GEN-END:initComponents

    private void okButtonActionPerformed(java.awt.event.ActionEvent evt)
    {//GEN-FIRST:event_okButtonActionPerformed
        dispose();
    }//GEN-LAST:event_okButtonActionPerformed

    public static void main(String args[])
    {
        java.awt.EventQueue.invokeLater(new Runnable()
        {
            public void run()
            {
                new exportComplete().setVisible(true);
            }
        });
    }

    // Variables declaration - do not modify//GEN-BEGIN:variables
    public javax.swing.JLabel exportCompleteLabel;
    private javax.swing.JButton okButton;
    // End of variables declaration//GEN-END:variables
}
```



Bibliography

DeSilver, Drew. "Chart of the Week: The Most Liberal and Conservative Big Cities." Pew Research Center. August 08, 2014. Accessed April 19, 2019. https://www.pewresearch.org/fact-tank/2014/08/08/chart-of-the-week-the-most-liberal-and-conservative-big-cities/.

Epstein, David, and Sharyn O'Halloran. "Measuring the Electoral and Policy Impact of Majority-Minority Voting Districts." American Journal of Political Science43, no. 3 (April 1999): 367-95. Accessed April 11, 2019. doi:10.2307/2991799.

Census 2000 Profiles of General Demographic Characteristics Texas prepared by the U.S. Census Bureau, 2001.

Census 2000 Summary File 1 Texas prepared by the U.S. Census Bureau, 2001.

"The 50 Largest Cities in the United States." PolitiFact. Accessed April 19, 2019. https://www.politifact.com/largestcities/.

"Bush v. Vera." Oyez. Accessed April 1, 2019. https://www.oyez.org/cases/1995/94-805.

"Introduction To Federal Voting Rights Laws." The United States Department of Justice. August 06, 2015. Accessed April 11, 2019. https://www.justice.gov/crt/introduction-federal-voting-rights-laws-1.

"The Generation Gap in American Politics." Pew Research Center for the People and the Press. March 1, 2018. Accessed April 12, 2019. https://www.people-press.org/2018/03/01/the-generation-gap-in-american-politics/.

"Congress Profiles | US House of Representatives: History, Art & Archives." Congressional Profiles | US House of Representatives: History, Art & Archives. Accessed March 20, 2019. https://history.house.gov/Congressional-Overview/Profiles/106th/.