

Group Members: Phillip Gao, Simon Roling, & Elliu Huang

Our Project:

Find My Quaker is a unique way to visualize connections between our elite athletes here at the University of Pennsylvania. Find My Quaker provides many features to learn about our athletes and create social networks based on shared characteristics. All of the information about these players that we use is publically available at <https://pennathletics.com/>.

Find My Quaker uses JSOUP to gather information about players, and creates a graph representation of the athletes using an adjacency list and adjacency matrix based on user inputted characteristics: sport, academic year, hometown, high school, or height. By utilizing the concept of triadic closure, Find My Quaker can find existing connections and recommend potential connections between individuals who share similar characteristics.

Find My Quaker's implements a Java Swing interface to create a simple user experience. At the top, the instruction button further explains what each button does. The functionality of Find My Quaker is two fold: it can be used to query data about the athletes, and it can create social networks and form closures based on certain characteristics.

Overall, Find My Quaker is a fantastic way to learn about athletes and teams, as well as find possible connections between players!

Assumptions:

The assumptions of this project primarily relate to closure and network formation. Our first assumption is that athletes on the same team will automatically be connected to each other in the network because it makes sense that everyone on the same team knows each other. The rest of the characteristics for network formation are actually decided by the user, and can include height, academic year, high school, and hometown.

Closures can be formed from this by analyzing each player's connections. We use triadic closure to see who each player's neighbors are connected to, and suggest a connection between the given player and the players adjacent to the given player's neighbors.

We also made a few assumptions when parsing data from the Penn Athletics website. Firstly, we assumed that each player contained at least their name, biography link, and image link when parsing the information from the website. Some teams decided not to list height, academic year, high school, or hometown, so we made sure to check that those values were not null before we added them to our data structures.

Instructions:

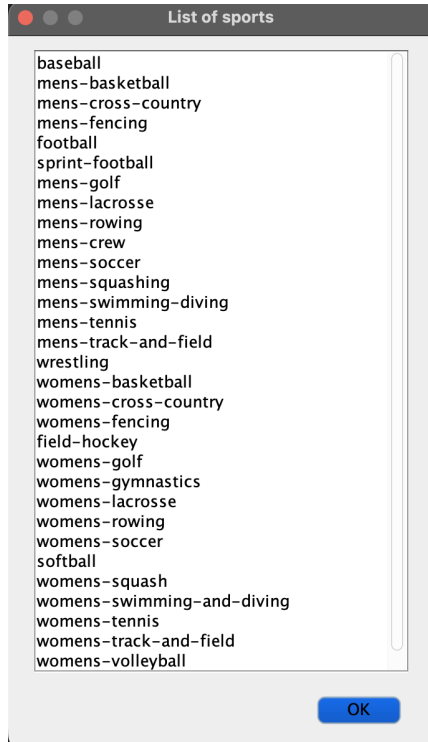
Step 1

Run GUI.java, a window will pop up with a panel of different options at the top of the window. It is recommended that you start with the instructions button as that will tell you how to use it, but this User Manual should also suffice. Any button you click will open a new window with the necessary information to allow the program to work properly. After running GUI.java, this window will pop up:

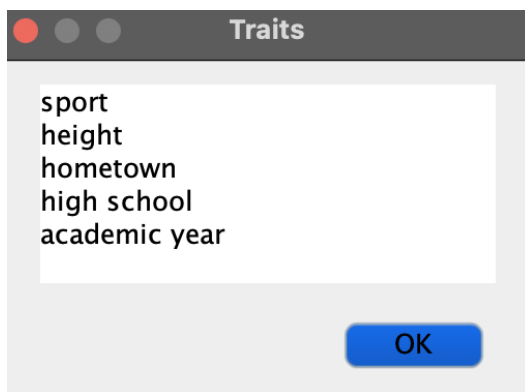


Step 2

The first button (besides instructions) will give the user a list of sports. This list is properly formatted, so the user should copy and paste the sport(s) they are interested in finding more information about. This will be necessary whenever the user is prompted to enter a sport by name. This is what the user will see when they click the sports button:



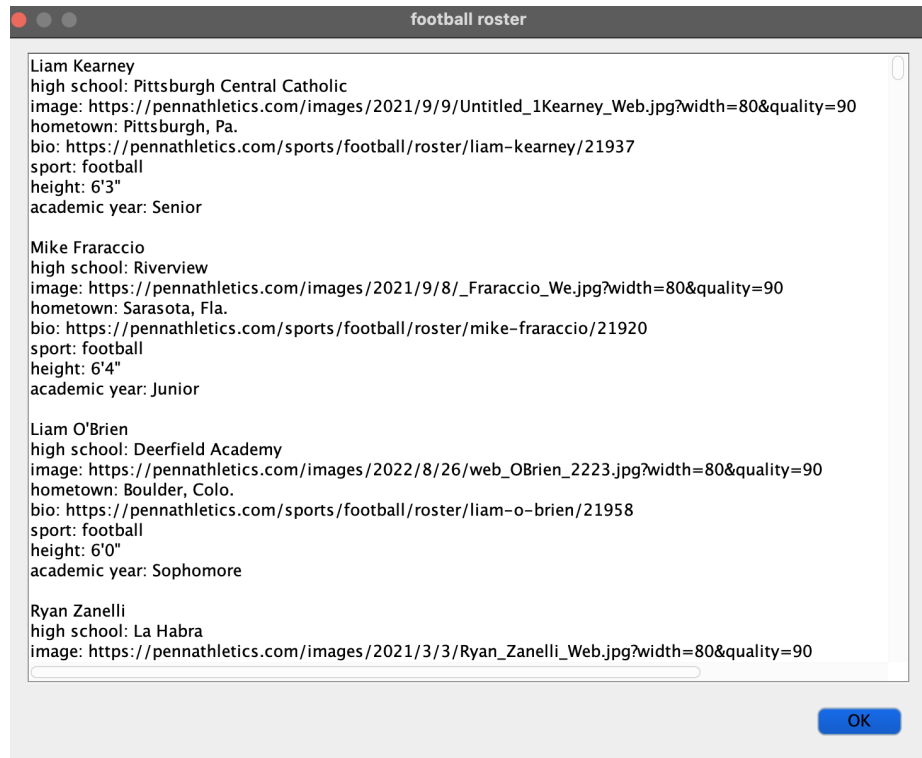
Next the user will come across the traits button. Clicking this will reveal the list of traits that can be used whenever the user is prompted for a trait. The traits listed here are properly formatted for input with our platform, variations may cause the platform to not work as expected. This is what the user will see when they click it:



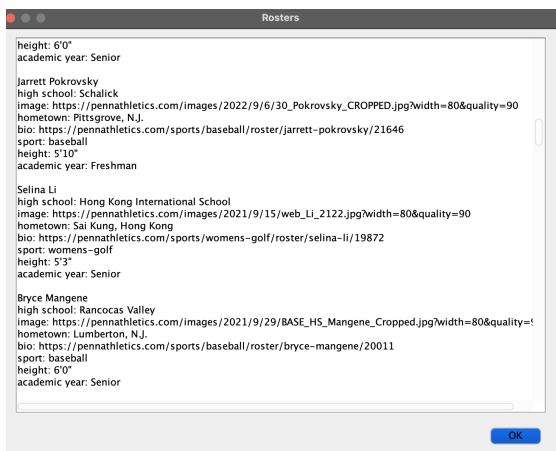
Next, the profile button allows the user to find player information given a properly spelled athlete's name. Users can find player names by using the roster function, or using outside knowledge of a player's name. It is possible to copy profile links into google to find the player's images and bios. Please note that it may take a few seconds to return a profile. Here is what an athlete profile looks like for Courtney Savage, a freshman women's lacrosse player:



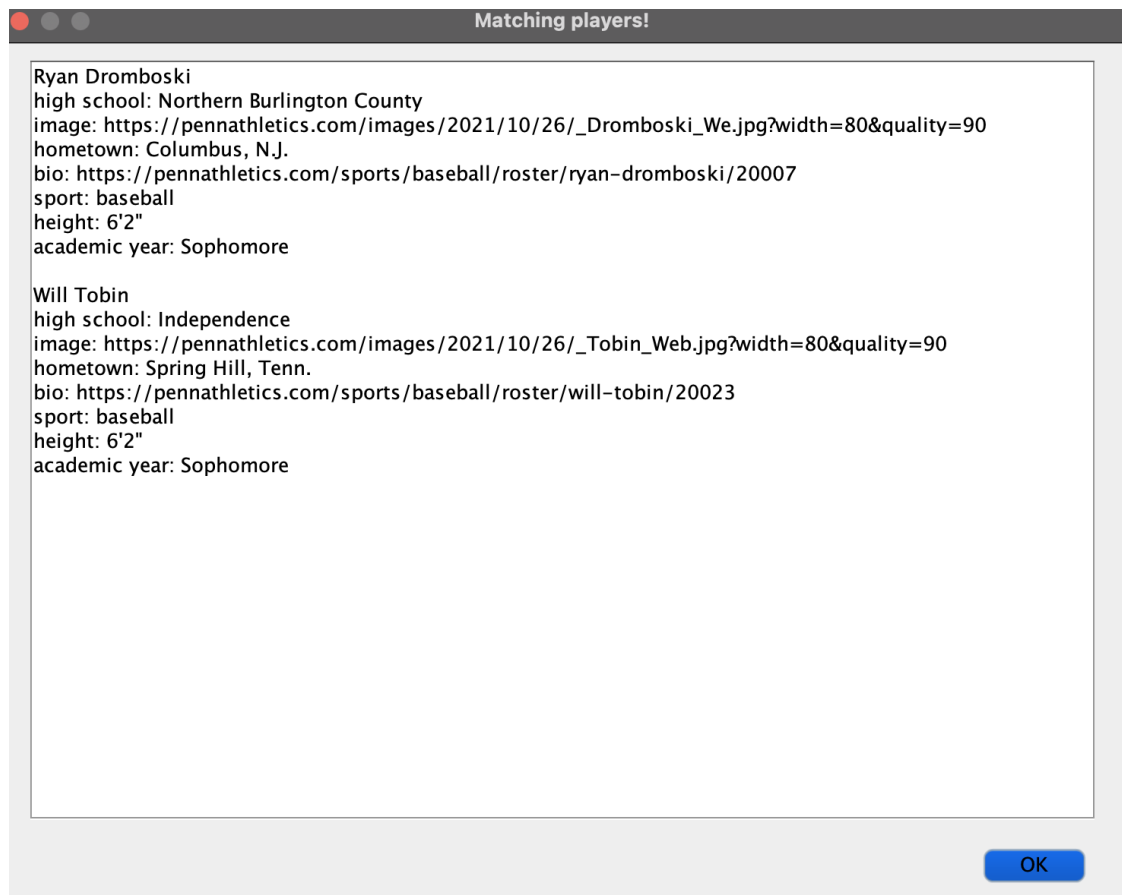
The next button is roster. The user can input any sport from the list of sports (the first button), and it will return a roster with basic information about each player within the team. Please note that if the penn athletics website does not have specific information about a certain player or sport, the roster will not provide that information within the description. Here is what part of the football roster looks like:



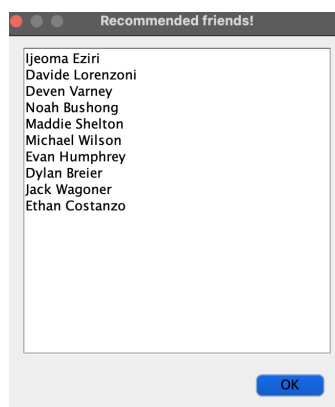
The multiple rosters button allows the user to do the exact same thing as the rosters button, but with as many sports as the user wishes. The user will input the number of sports, as well as which sports, and it will display a roster for all sports listed. Here is what part of the women's golf and men's baseball roster looks like:



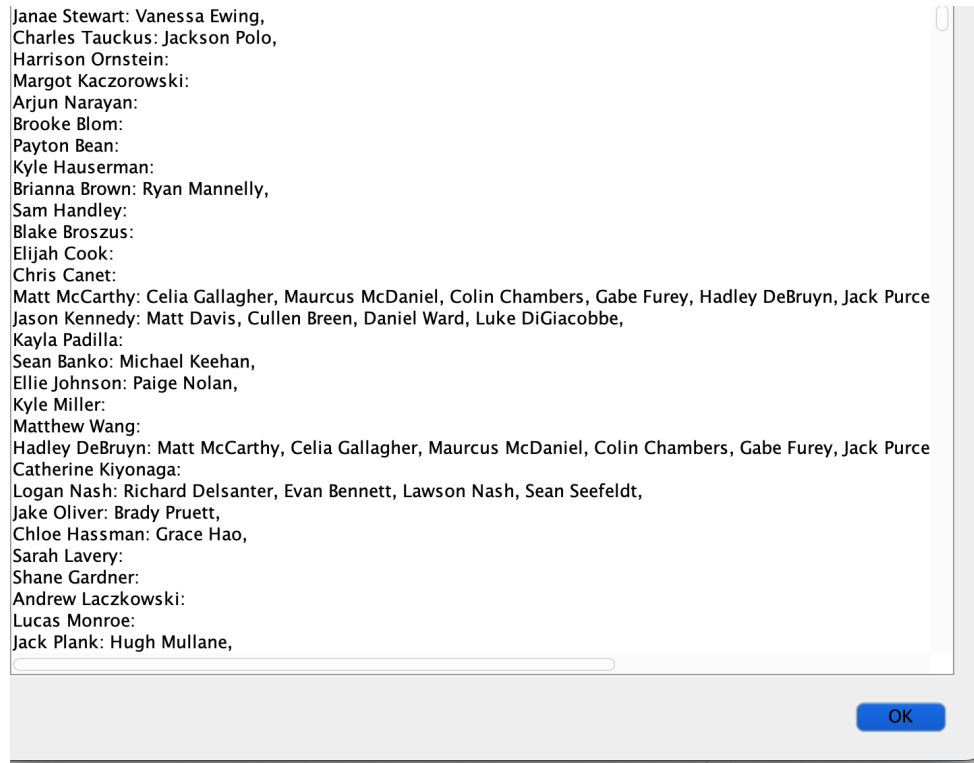
The match athletes button provides a list of all athletes that match whichever traits are inputted by the user. The user can input up to 5 traits including academic year, height, sport, high school, and hometown. Inputting no traits will display every single and their profile. Any trait not inputted will be disregarded. Here is a list of all of the sophomore baseball players who are 6'2" tall:



The find friends button will prompt the user to input the same traits as the match athletes button, however, this returns a (randomized) list of up to 10 athletes that have the exact traits that the user searches for. For example, if a user wanted to find a friend who was a 6'0" freshman athlete, they could enter that into the prompts and Find my Quaker will return a list similar to this (albeit with different names):



The adjacency list button takes in at least one trait, and will create connections between each athlete, and those who share at least one of the traits with the athlete. The user can form a social network between athletes by inputting different traits! Any isolated node in this social network will not have any names to the right of the colon. This is what part of the adjacency list of athletes who went to high school together looks like:

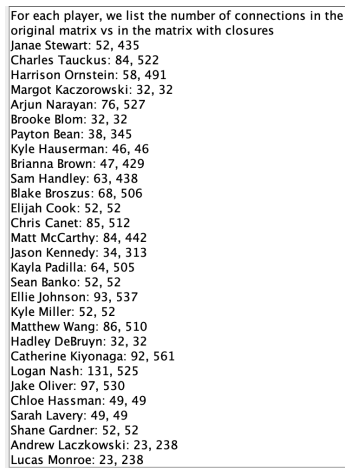


Janae Stewart: Vanessa Ewing,
Charles Tauckus: Jackson Polo,
Harrison Ornstein:
Margot Kaczorowski:
Arjun Narayan:
Brooke Blom:
Payton Bean:
Kyle Hauserman:
Brianna Brown: Ryan Mannelly,
Sam Handley:
Blake Broszus:
Elijah Cook:
Chris Canet:
Matt McCarthy: Celia Gallagher, Maurcus McDaniel, Colin Chambers, Gabe Furey, Hadley DeBruyn, Jack Purce
Jason Kennedy: Matt Davis, Cullen Breen, Daniel Ward, Luke DiGiacobbe,
Kayla Padilla:
Sean Banko: Michael Keehan,
Ellie Johnson: Paige Nolan,
Kyle Miller:
Matthew Wang:
Hadley DeBruyn: Matt McCarthy, Celia Gallagher, Maurcus McDaniel, Colin Chambers, Gabe Furey, Jack Purce
Catherine Kiyonaga:
Logan Nash: Richard Delsanter, Evan Bennett, Lawson Nash, Sean Seefeldt,
Jake Oliver: Brady Pruett,
Chloe Hassman: Grace Hao,
Sarah Lavery:
Shane Gardner:
Andrew Laczkowski:
Lucas Monroe:
Jack Plank: Hugh Mullane,

OK

[illegible][illegible]

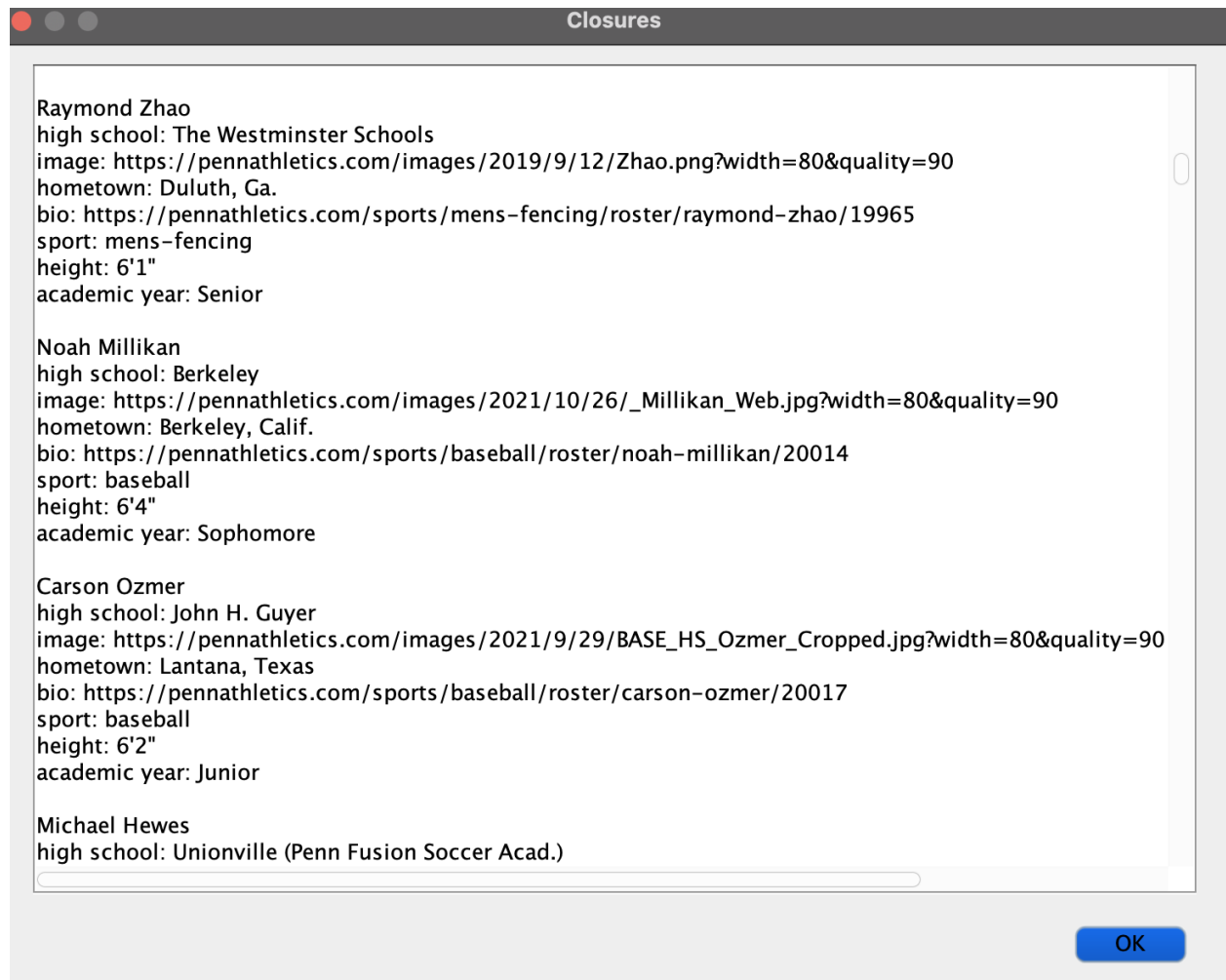
The Form Closures button suggests new connections between athletes. When the button is clicked, the user inputs traits that will be included in the original network. It is **strongly** recommended that the user inputs at least two (2) traits for this to function at its full potential. Inputting one trait will cause no new closures to occur. First, the platform creates a social network of the athletes based on the inputted traits. Then, Find My Quaker performs triadic closure to suggest new connections between players. Find My Quaker analyzes each athlete's connections, and suggests a connection between the athlete and whoever is connected to the athlete's connections (and is not already connected to the athlete). The list that is returned to the user contains the original number of connections that each athlete has based on the network created from user inputs, as well as the number of suggested connections based on the original network as well as the triadic closure. Here is what this looks like for a network of athletes in which height and sport are considered:



For each player, we list the number of connections in the original matrix vs in the matrix with closures

Janae Stewart:	52, 435
Charles Tauckus:	84, 522
Harrison Ornstein:	58, 491
Margot Kaczorowski:	32, 32
Arjun Narayan:	76, 527
Brooke Blom:	32, 32
Payton Bean:	38, 345
Kyle Hauserman:	46, 46
Brianna Brown:	47, 429
Sam Handley:	63, 438
Blake Broszus:	68, 506
Elijah Cook:	52, 52
Chris Canet:	85, 512
Matt McCarthy:	84, 442
Jason Kennedy:	34, 313
Kayla Padilla:	64, 505
Sean Banko:	52, 52
Ellie Johnson:	93, 537
Kyle Miller:	52, 52
Matthew Wang:	86, 510
Hadley DeBruyn:	32, 32
Catherine Kiyonaga:	92, 561
Logan Nash:	131, 525
Jake Oliver:	97, 530
Chloe Hassman:	49, 49
Sarah Lavery:	49, 49
Shane Gardner:	52, 52
Andrew Laczkowski:	23, 238
Lucas Monroe:	23, 238

Our final button, entitled “All Athletes” shows the user the profiles of every single athlete at Penn! Here is what part of this list looks like:



Class Descriptions:

FindMyQuaker.java

The main function in FindMyQuaker.java provides a command line interface for all the functions provided in the final java swing interface. When you run the main function, there will be 10 options to choose from.

The first option simply prints out all the different sports teams. The second option receives a sports team (mens-fencing, womens-tennis, etc.) and prints the list of athletes on the team including all their characteristics. The third option returns all the characteristics given the name of an athlete, including sport, academic year, hometown, high school, and height, and links to their bio and profile picture.

FindMyQuakerGUI.java

This file contains all the backend functionality for converting the command line output of methods in FindMyQuaker.java into a displayable GUI pop-up window.

PennAthleticsParser.java

This file hosts all the code related to parsing pennathletics.com, such as parsing all the rosters for each sport, finding characteristics of an athlete, and returning athletes sharing similar characteristics.

PennAthleticsNetwork.java

After collecting all the data from the website, this file creates a social network of athletes based on user-inputted characteristics. We represented the social network with both an adjacency list and adjacency matrix. In addition to creating the social networks, we also have a function to create closures based on the other characteristics that haven't been used to create the social network. For example, we can initialize a social network based on sports and height where every athlete is connected to every other athlete on the sports team or people with the same height. If athlete A and athlete B share the same sport, A will be connected to B. If athlete B and athlete C share the same height, B will be connected to C. Then by triadic closure, A and C will be connected in the new graph.

RunFindMyQuaker.java

This file creates the GUI interface for the FindMyQuaker UI.

GUI.java

Runs the GUI interface.

Troubleshooting

Please take note of the following:

- In the GUI, when prompted to enter a trait, sport, or other value, you must enter something in the field. Exiting from the prompt without any submission may result in a `NullPointerException`.
- Sports team names and traits must be spelled exactly as they appear in the “Sports” and “Traits” pop-up window. All words are lowercase.
- Please enter integer values when the prompt asks for the number of traits, for example.
- The GUI may take a while to load since it has to parse and sort through all the data on the Penn Athletics website.