HELP AND USER MANUAL

Ignite Mentoring

Coders for Causes

1. How the program works

The Pairing algorithm works off of the stable marriage (stable hospital to be more precise) principle. Boiled down to its core the algorithm simulates each person sending requests to join a class, accepting and retroactively rejecting previous ones should it better one's outcome. The program is not guaranteed to return the optimal arrangement for each class but for the system of classes as a whole. This means that the program would rather allocate volunteers so that it partially satisfies all schools than have a smaller number of schools, each completely allocated.

The program reads an excel file for all input. The first sheet of the file contains information like the number of volunteers and classes and most importantly the number of different skills and the names. For example a volunteer could have the traits: Experienced mentor, Male, German speaker etc. This list of skills will be used by the program to determine who the best match for a class is. It is recommend that there be one universal skill named total.

The next sheet has a list of the different classes, each with a different set of values for skills. This set of numbers is the total value of the volunteer's score in that skill that the program should aim to allocate to a class. This sheet must have a field named "Name" ,"Max Size" and fields named the exact same as the skills on the first sheet of the input file.

This last sheet of the file is a list of each volunteer, their skills, and their preferences. There must be a field for each of the skills listed on the first sheet and a field for each "Name" listed on the second sheet.

The program can take into account groups of people applying together, simply treat this group as one person with the sum of their skills. Note that the program may unfairly select the group above individuals if there are more volunteers then spaces in all classes for this reason it is suggested that groups be used sparingly or not at all.

Skills may be more or less important to reflect this it is possible to multiply all the important skills, demand (in the class list), and supply (in the volunteer list) by a number α. This skill will then be α times more important to the program, for example it may be very important that the schools that need German speakers get German speakers and much less important that the schools have an even mix of male and female volunteers.

The program will not in any way edit the input file, as such feel free to run the program as many times as you like with different class requests to get different pairings. It is recommended that you keep a backup of the original file should an unknown error somehow crash the program and affect your data.

The output file will be saved in the version of excel that the input file was. The created output file will be a normal excel spreadsheet file in every way, so feel free to edit it. Remember, you can always get a new copy by running the program using the same input again.

2. How to add or remove skills

The program can take any number of different skills. To remove a skill from an allocation, all you need to do is remove it from the list of skills in the 1st sheet and update the number of skills in cell A2 to reflect this. The program will now treat the columns in sheet 2 and 3 just like any other text field simply copying it across without using it for the pairing.

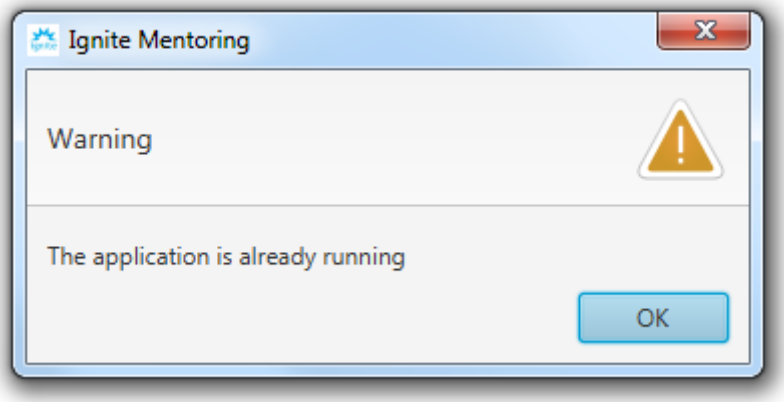
To add a skill you must add the name of the skill (for example “can drive”) to the list of skills on the 1st sheet, update the number of skills (sheet 1 cell A2) and provide “Supply” and “Demand” information. To update “Demand”, add a column to sheet 2 named exactly the same in the list of skills, then give a value of that skill that each class should strive to get. To update “Supply”, add a column named exactly the same as in the list of skills to sheet 3, then for each volunteer give them a score for that skill.

3. How to use the program

The following will show the user how to use the program on Windows and Macintosh using screenshots taken from both operating systems. For this manual, Windows 7 64-bit version and OS X El Capitan were used (Note that there will be minor differences in the program window with different versions of operating systems).

The program was written to allow only one instance of the application to run. The following error message will pop up if the program is made to run more than once. Mac does not provide an error message, instead it brings the running application from the background to the front.

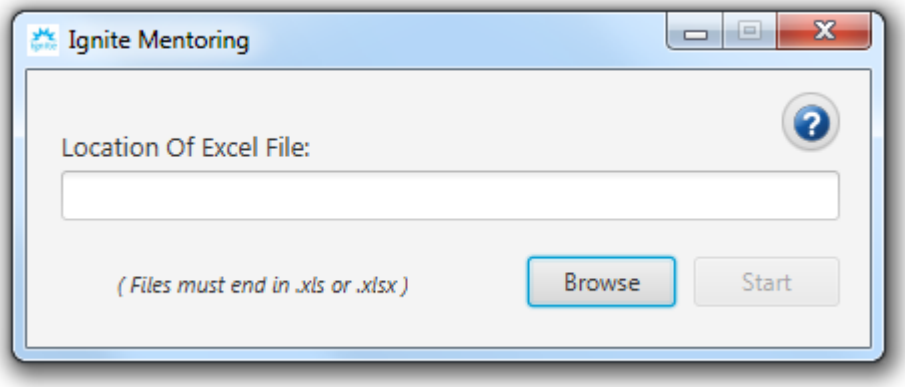
<u>For Windows:</u>



3.1.1 *Pop-up message when program runs more than once on Windows*

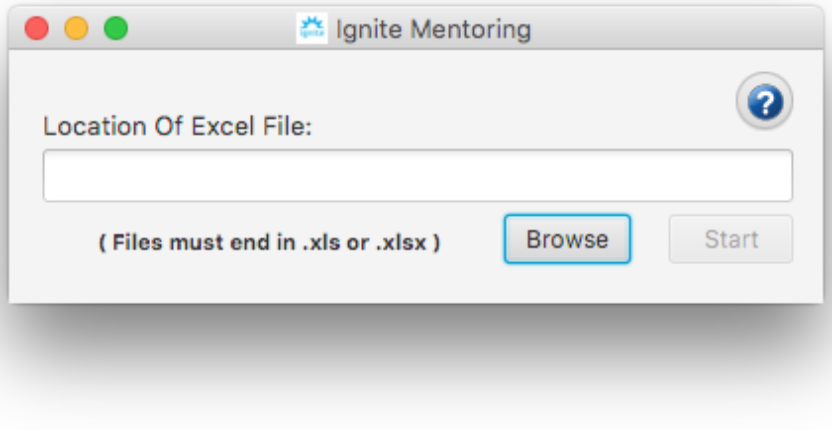
When the program runs, the window (respective to the running operative system) below appears first. This non-resizable window contains a text field and three buttons namely the “Help” button denoted by the question mark, the “Browse” button and the “Start” button. In the text field, the user may either type in the path to the excel file manually or use the “Browse” button (which will bring up the window from images 3.3.1 and 3.3.2) to search the computer for the required excel file. The “Help” button opens up this user manual when clicked. The “Start” button will remain inactive until the text field is filled. When the “Start” button is clicked, the program checks if the selected file is an excel file and that it exists, if not, it displays an error message.

For Windows:



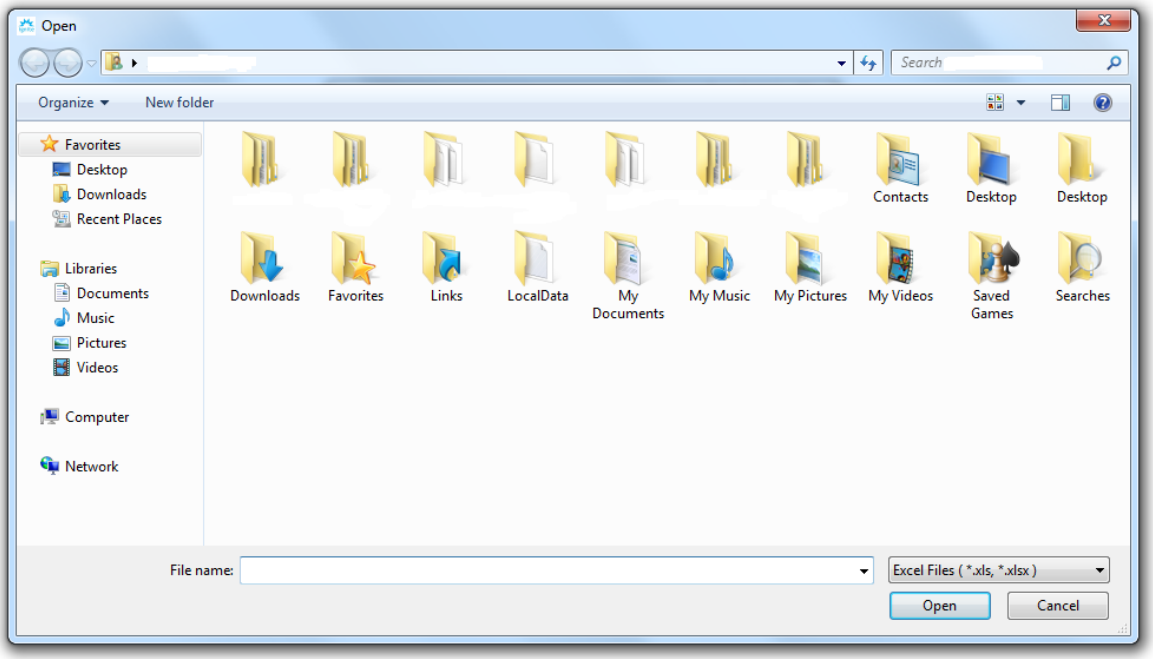
3.2.1 First window of the program on Windows

For MacOS:



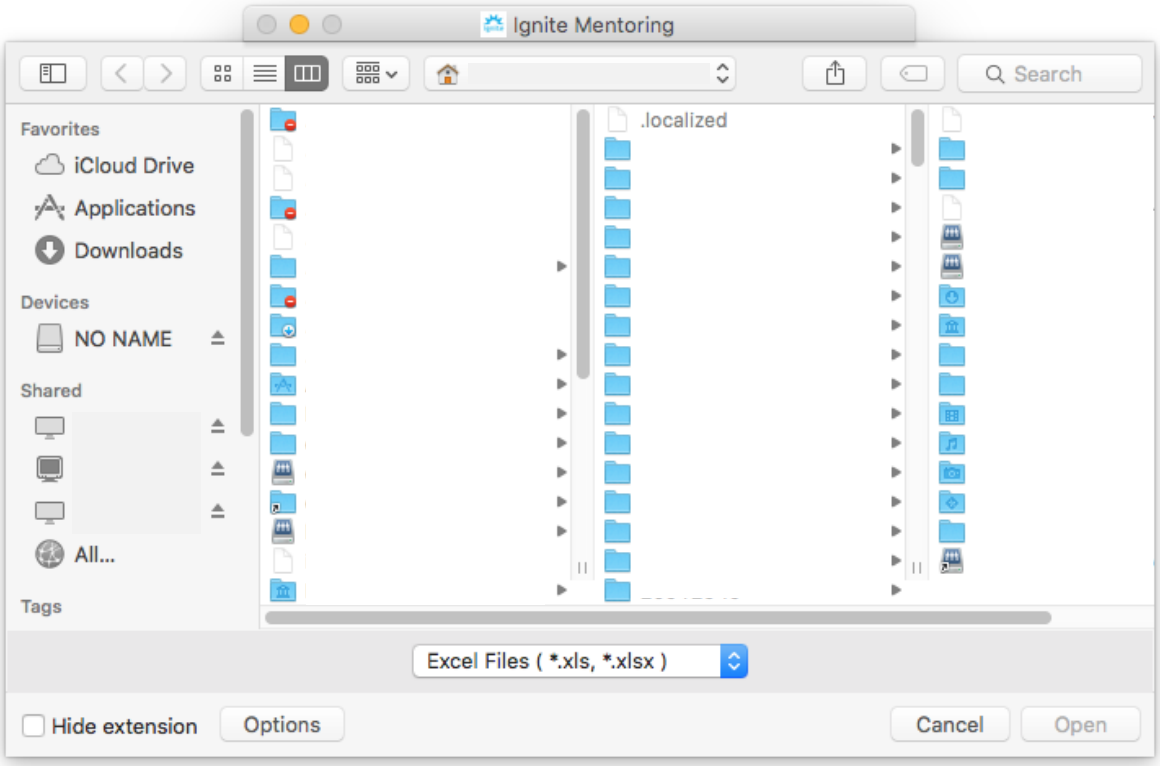
3.2.2 *First window of the program on Mac*

For Windows:



3.3.1 *File explorer opened when "Browse" button is clicked on Windows*

For MacOS:

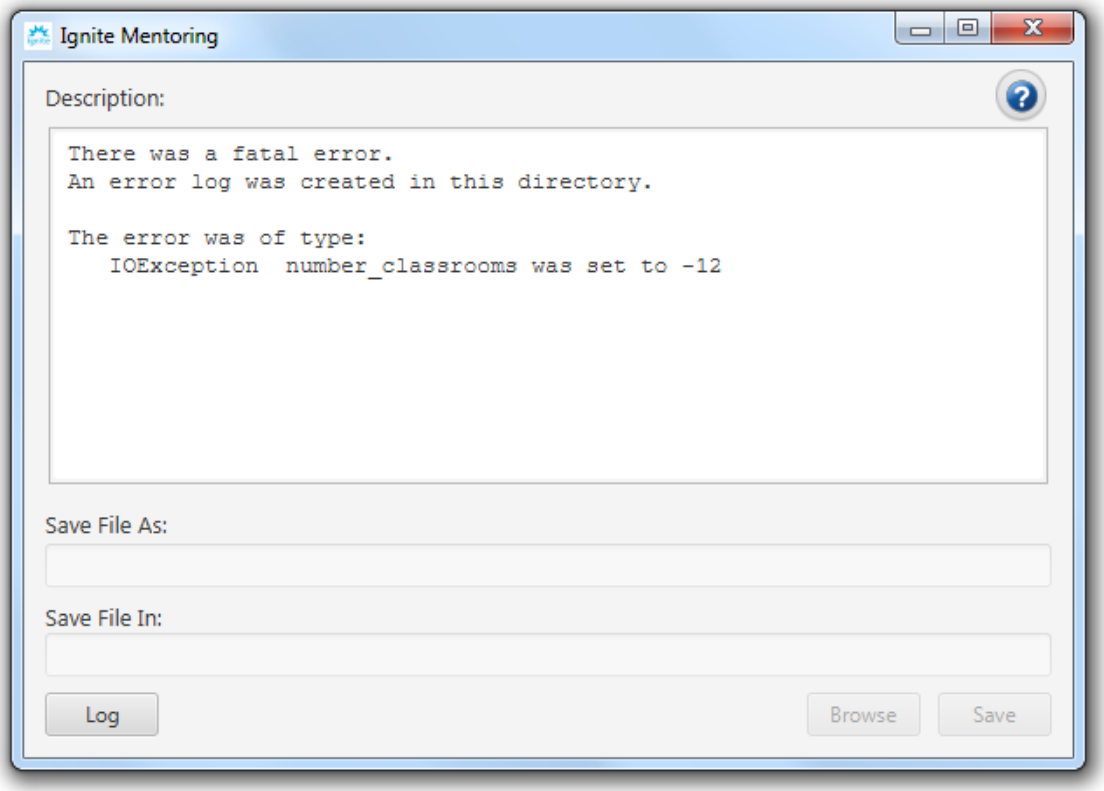


3.3.2 File explorer opened when "Browse" button is clicked on Mac

Once the start button is clicked, the program then opens the selected excel file internally and the algorithm runs. It then goes to the next window which is the output window. This window contains a scrollable text area, two text fields namely “Save File As” and “Save File In” and four buttons which are the “Help” button, the “Browse” button, the “Save” button and the “Log” button.

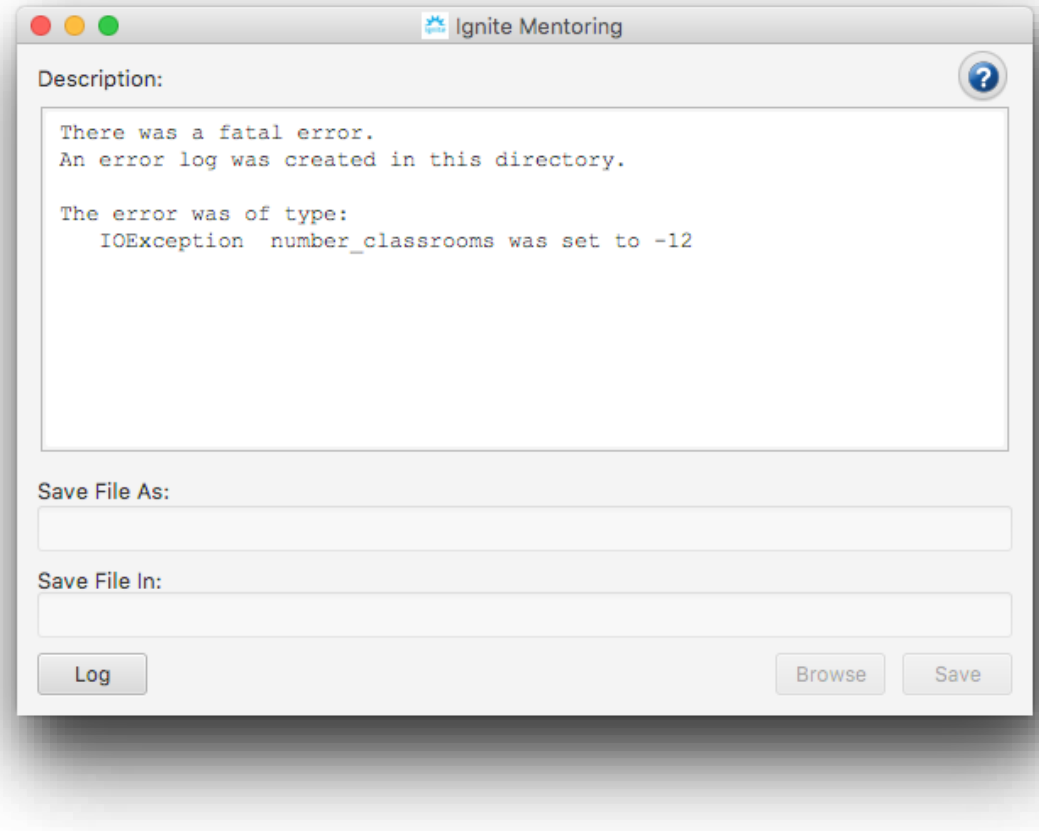
If there was an error in any of the fields, the program closes the excel file, generates an error log file named according to the time the error was found, and displays an error message. The program then shows the below window with all buttons and fields un-selectable except for the “Help” and “Log” buttons.

For Windows:



3.4.1 Output screen when an error was found in the excel file on Windows

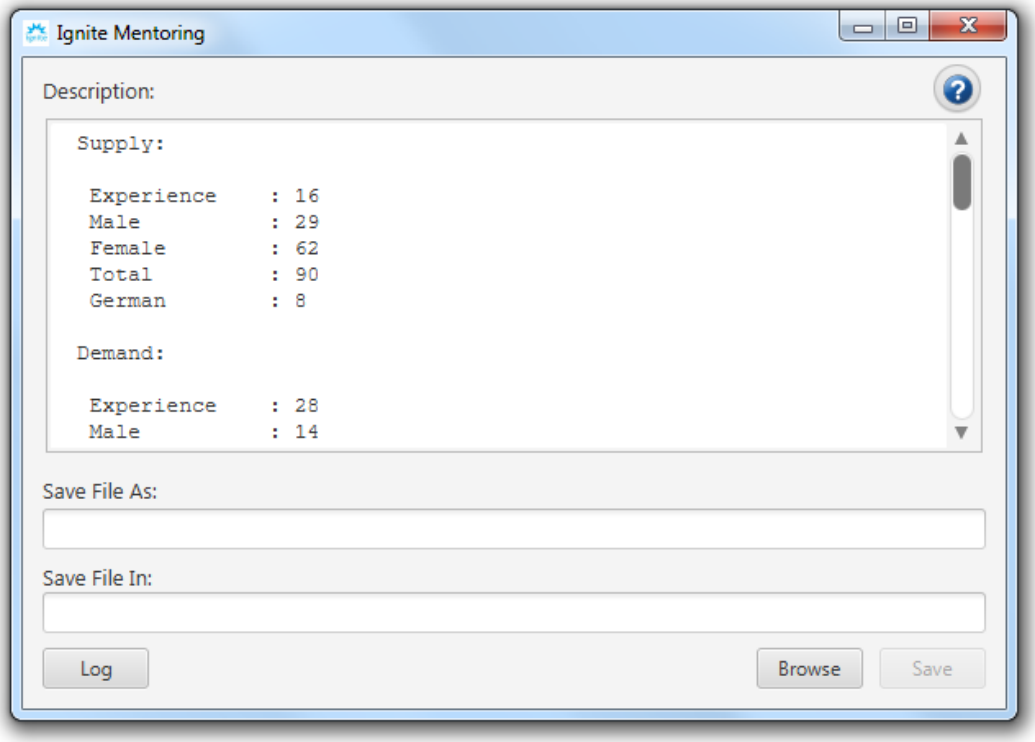
For MacOS:



3.4.2 Output screen when an error was found in the excel file on Mac

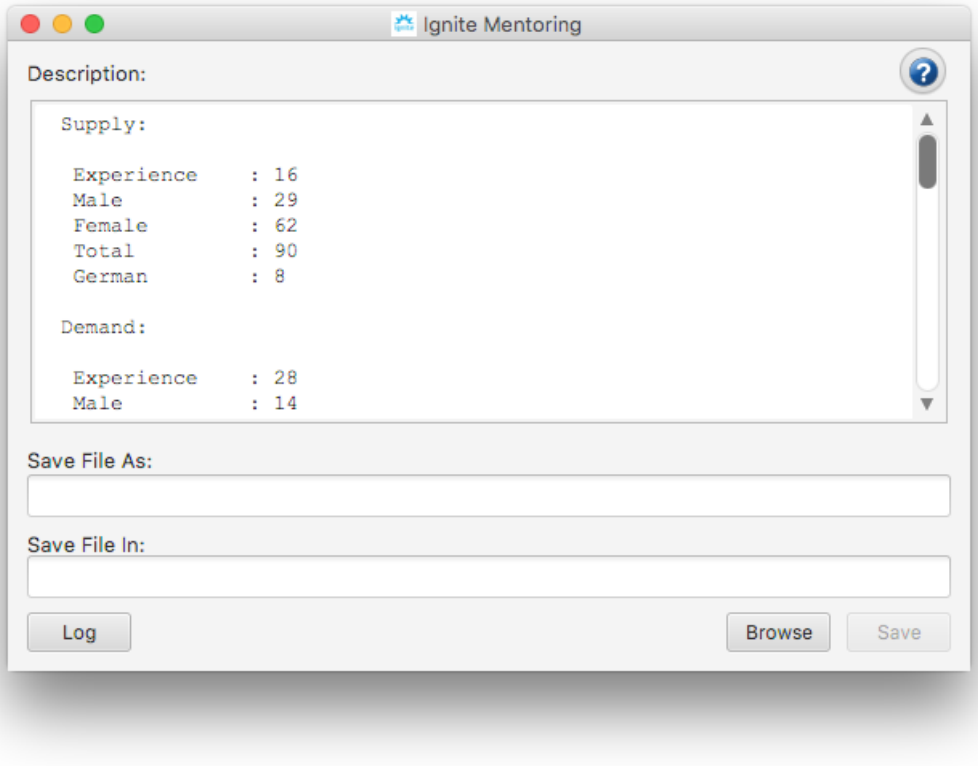
The text area shows a brief summary of the output. The first text field named “Save File As” is a field where the user may wish to name the excel file for the output. Note that this field may generate an error message if there was an error in the file name. The second text field named “Save File In” is a field for the path that the user wishes to save the file to. This path can either be typed in manually or the user may use the “Browse” button (which bring up the file explorer similar to images 3.3.1 and 3.3.2) to access the file or folder they wish to save the excel file to. The “Help” button bring up this help and user manual as well. It is recommended that the user not use the “Log” button as it provides an extremely detailed output of the algorithm and will be used mainly for debugging. The “Start” button then saves the excel file using name and destination given by the user.

For Windows:



3.5.1 Output screen with summary of volunteers on Windows

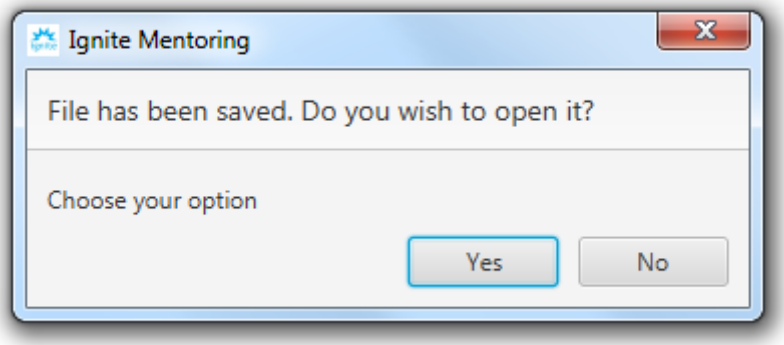
For MacOS:



3.5.2 *Output screen with summary of volunteers on Mac*

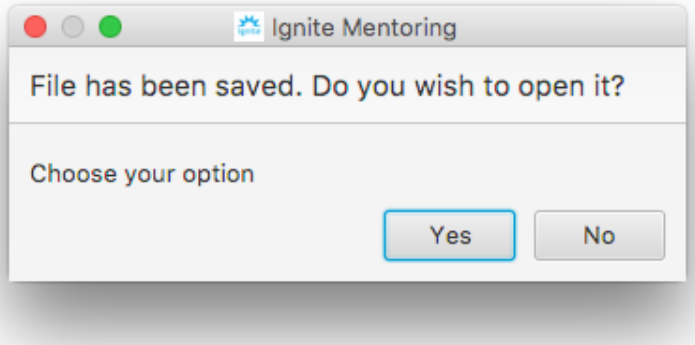
If the file was successfully saved, the program brings up this final window which asks the user if they wish for the saved excel file to be opened. If “Yes”, the program closes and opens the excel file. If “No” was clicked, the program terminates execution.

For Windows:



3.6.1 Dialog box with options after save on Windows

For MacOS:



3.6.2 Dialog box with options after save on Mac

4. Contact

If there are any issues found that could not be fixed, please contact Coders for Causes (CFC) using the email address help@codersforcauses.org