CIS7 Project Team 3: Case 2

Team members: Mario Cuaya, Jason Atalig, Marina Gonzalez

Project Summary

What problems are you solving in this project?

In this project, there are three main problems. The first problem is figuring out a way to store and associate information like language and specialization for a list of 100 students. The second is finding a way to store information about countries like language and name. The third problem is creating a method to distribute students to each country, storing this information for probability, and displaying this information.

What solutions are you implementing in the project?

- For problem 1, a Student class is implemented that stores the relevant language and specialization information in string format, and a vector of 100 Students is created with randomized information for each Student.

- For problem 2, a Country class is implemented that stores the name of the country and language in a string format. Country objects are hard-coded with the appropriate names and languages.

- For problem 3, the Country class has several functions that calculate and store the percentage of Student objects that are matched based on their specializations and languages which are also displayed to the user through a print function. The students are distributed in the main function through a series of nested for-loops.

Provide explanation of calculations and algorithm implementation.

For the Student class, a random number function is implemented that generates random numbers based on a seed that depends on a real time value which allows for a different set of random values each runtime. These random values are then used for determining which language or specialty each Student has.

In the Country class, the percentage calculations for specialization and language probability for each country are done by functions that loop through the Students vector, sum the amount of students that match the specialization/language, and return the percentage (matched students/total students) of students.

In the main function, students are distributed to each country based on their language. This is done by having vectors for filtering students and countries. A nested for-loop is utilized to cycle through the list of languages, adding country information to a filteredCountry vector based on if the country matches the current language iteration. In the same manner, Student objects from the participants vector are added to a filteredStudent vector, and removed from the participant vector. A while loop is performed to distribute Students from the list of filteredStudents across the list of filtered Country. The filtered Countries then dump these students into their matching country in the main Countries vector. This ensures that students match with countries with the appropriate language.

What are the program objectives? Explain how your program is interacting with the user and its purpose.

The program's purpose is to simulate 100 medical participants and calculate the probability of each of these participants being assigned to various countries. Each time the user runs the program, different simulations are performed.

How is discrete structures implemented in the C++ program?

Discrete structures are implemented through the use of the random algorithm in the Student class, the probability algorithm in the Country class, the sorting algorithm in the Country class that searches through the vector of Students to determine probability data, and the sorting algorithm in main function that distributes students in various countries based off of language information.

On the distribution of students: The program distributes the participants equally across each country with the same language. The following describes this pattern in detail:

Let C be the set of all countries supported by the program, and $c \in C$.
Let L be the set of all languages supported by the program and $\ell \in L$.
Let L_c be the set of all languages for country c.
Let C_ℓ be the set of all countries that have language ℓ.
Let P be the total number of participants, and P_c be the number of participants matched to country c.

Since the participants are randomly generated, there is an equal distribution of languages among them. For any given language ℓ, there will be $P/|L|$ amount of students that match that language. And because our program distributes the participants equally based on language, then each country with language ℓ will get $P/|C_\ell||L|$. Then the amount of participants that a given country will receive is.

$$P_c = \sum_{\ell \in L_c} \frac{P}{|L||C_\ell|}$$

What are the limitations of the program?

The limitations in this program is that there are not any user-defined values, so the user could not use this program to determine probability of their own specific participant information. It is assumed that another program would get user input and store it to some database. Then our program could be modified to grab the data from there to do the calculations.

The program is also somewhat inefficient in terms of speed and memory due to the intense use of nested looping and vectors. In a real world scenario, you would only need to run this program once, so speed is not a concern. However, problems may occur with not having enough memory to perform the calculations. Repl.it seems to handle it just fine, though.

The method of distribution may not be favored by the facilities, as it gives countries with more languages more students. This isn't necessarily a bad thing, since it makes sense that countries with more languages can support more students. However, a more equal distribution could be favored.

Another criticism of the distribution is that it does not consider student preferences or students with multiple languages.

Provide recommendation on improving the limitations of the program.

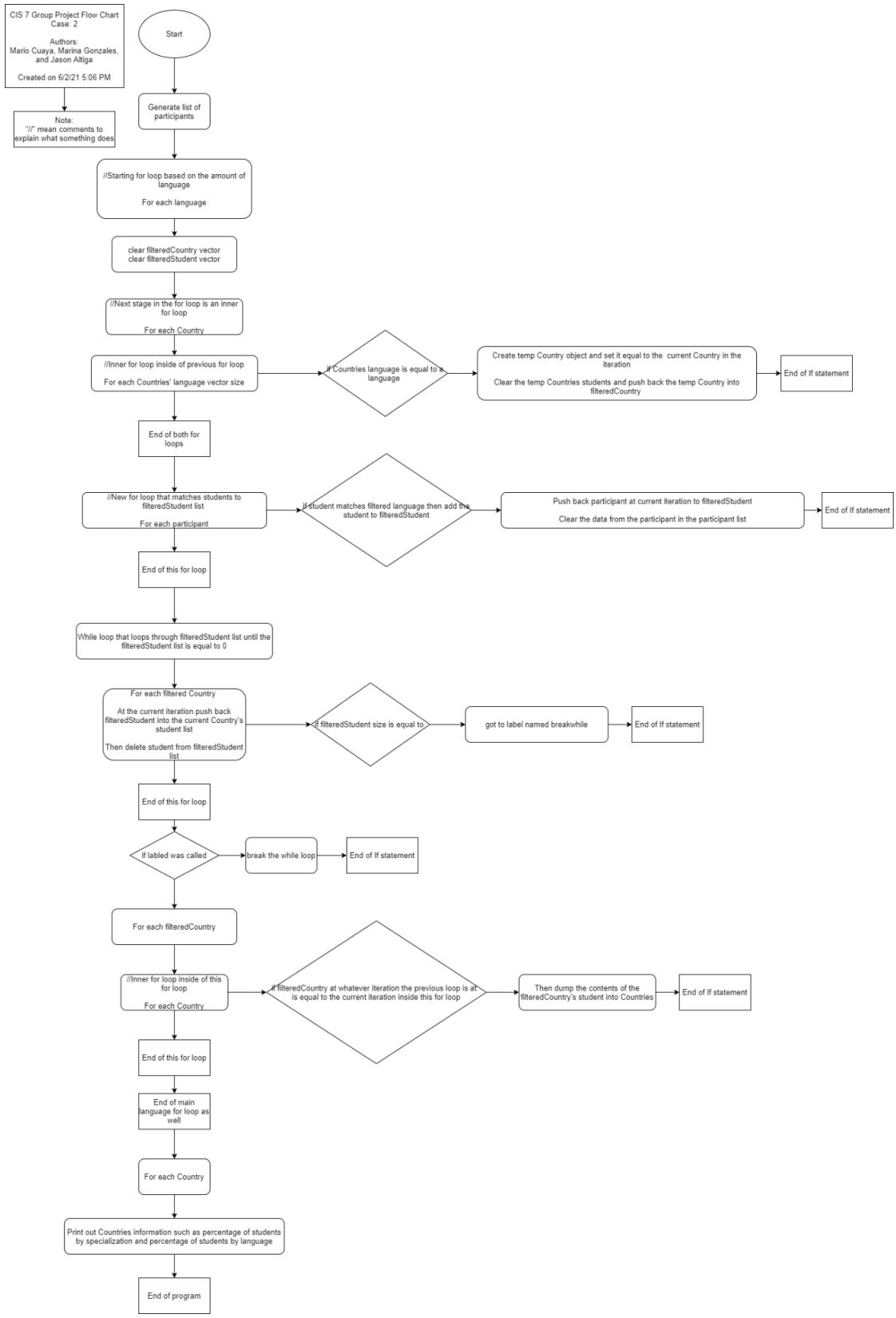
As stated before, the first limitation can be addressed with a program to collect student data to then be processed by our program. If we wanted a student to look at their own individual statistics, we can create an alternative branch dedicated to analyzing those features.

The second limitation can be improved by first creating a presorted group of students based on their language since the country assignment process is language based. This would allow for a smoother and faster search, since we would no longer need to utilize vectors for filters, and matching is easier with predictable positions of students with a given language.

If we wanted a more equal distribution of students across each country, what we could do is add a ranking system for multilingual countries, favoring students to countries with less languages. This would have to be implemented cautiously, however, since we need to consider the majority languages of each country and provide a reasonable proportion of students matching said languages.

To put more control for the students, we could expand the student object to include a list of ranked languages and country preferences. We would then match the students to the most preferred available country. There could potentially be limits to how many students a country could take, or this ranking system could be combined with the equal distribution system described above.

Flowchart:


CIS 7 Group Project Flow Chart
Case: 2
Authors:
Mario Cuaya, Marina Gonzales, and Jason Altiga
Created on 6/2/21 5:06 PM
Note:
"//" mean comments to explain what something does
Start
Generate list of participants
//Starting for loop based on the amount of language
For each language
clear filteredCountry vector
clear filteredStudent vector
//Next stage in the for loop is an inner for loop
For each Country
//Inner for loop inside of previous for loop
For each Countries' language vector size
If Countries language is equal to a language
Create temp Country object and set it equal to the current Country in the iteration
Clear the temp Countries students and push back the temp Country into filteredCountry
End of If statement
End of both for loops
//New for loop that matches students to filteredStudent list
For each participant
If student matches filtered language then add the student to filteredStudent
Push back participant at current iteration to filteredStudent
Clear the data from the participant in the participant list
End of If statement
End of this for loop
While loop that loops through filteredStudent list until the filteredStudent list is equal to 0
For each filtered Country
At the current iteration push back filteredStudent into the current Country's student list
Then delete student from filteredStudent list
If filteredStudent size is equal to
got to label named breakwhile
End of If statement
End of this for loop
If labled was called
break the while loop
End of If statement
For each filteredCountry
//Inner for loop inside of this for loop
For each Country
If filteredCountry at whatever iteration the previous loop is at is equal to the current iteration inside this for loop
Then dump the contents of the filteredCountry's student into Countries
End of If statement
End of this for loop
End of main language for loop as well
For each Country
Print out Countries information such as percentage of students by specialization and percentage of students by language
End of program