Project 1:

Team Members: Zaw Htun, Aiden Hert, Zain Hassan, Thompson, Conner Bennett

**NOTE: Couldn’t find the format of how to name the ZIP file for the project in the instructions so I used the same format we used for CS210.**

Zaw Htun

I created a person class which included two constructors. In these two constructors, one can be instantiated with the given name of the person, and the other can be called as a default constructor. The name of the person will be set as empty string in the default one while the name in the other constructor will be set with the given person name, and the children arraylist will also be instantiated. Moreover, there are three get methods in the class to return the name of person, mother and father and then one toString method is to return the person name. Furthermore, there are two set method to set the person of mother and father.

Also, addchild method is to add the children of the person and there is null for no children for the person.

Aidan Hert and Zain Hassan

We worked on the second part of the FileReader class (the familyTree method). First created a scanner and had the scanner to scan next line. Then it gets the mother from the next line and the father from the line after that. It also adds the person as a child to the mother and father. It does this for all the people in the list. It uses a Person object for this. The object lets you add a mother, father, and children to it. This creates a family tree from the beginning and not just as you input a person like I had originally planned.

Thompson:

For this project, I worked on the Main.java file that prompts the user for a name from the "tudor.dat" file and returns the appropriate output. The beginning of the program up to the point where I declared an ArrayList variable was pretty straightforward, as I’ve done this multiple times before. After that, I wrote the code from where I declared the scanner object up to the else if statement. When I ran my code I realized that the search can only be done once, so I decided to put it into a while loop and declare a new boolean variable. Regarding the code in the while loop, the first three lines weren’t difficult for me at all since I just needed to use my skills learned from CS 210 to do it. However, the next part after that did prove to be quite a challenge, as a lot of thinking and brainstorming had to be done on my part to figure out what types of loops I need, what they are for, and how to write them. The first loop that occurred to me was an if-else loop, an if statement for if the name typed in is one of the names on the tudor.dat file, the else statement being for if the name typed in isn’t one of the names on the tudor.dat file, in that case, it prints "No records found, please try again." The next idea that came to mind was having an if statement inside of the if statement I already made. The if loop turned out to be if(toString().equals(choice)), but then I realized that I needed to use the ArrayList I made earlier somehow in order to get the actual names of the people and to also have a for loop that ranges from 0 to the size of the ArrayList. So I put the if statement in the for loop and added familyTree.get(i) to the if statement. After that, the contents of the if statement were pretty simple, as I just needed to use familyTree.get(i) to get the person’s name and I just needed to add .getMother(), .getFather(), .getChildren() to that in order to get the person’s mother, father, or children. The only small problem with that was that square brackets showed up when printing the names of the children, but that was a simple fix using .toString().replace(). The code was almost complete at this point, but it was missing one thing and that was a way for the user to exit the code. So I decided to make an else if statement outside of the long inception of loops. This else if statement occurs when the user types “quit” or “Quit” (in the latter case, it converts it to all lowercase). When this occurs, it prints “Search ended.” and changes the boolean value to true, allowing the user to exit the program.

Conner Bennett:

For this project, I worked on the FileReader class file alongside Zain. My contribution to this project involved two methods, first being the listOfPeople, which accepts a file scanner as it’s perimeter. The method then reads the tudor file and creates a list of people from before the first instance of END. This method was nothing out of the ordinary compared to other programs from this class. Only involving the basics of file scanning and adding elements to an ArrayList. The next method I put into this piece of code was the getPersonString method that accepts a String as it’s perimeter and cycles through the list generated in the ListOfPeople and returns a null value if a person of a given name is not found. This method took a little creativity as I need a way to search the list and check if a name is found. I accomplish this with the use of a for loop and an if statement. Alongside the code I implemented into the program, I tried my best to also help in the form of troubleshooting bug fixes, contributing ideas, and attempting to keep things organized.