**EEE 206 - PROGRAMMING**

**Project 2 Report**

**Voting**

Submitted by:

*Batuhan Yaman (EEE)*

*Ali Yener Dikmen (EEE)*

*Ömer Tezel (EEE)*

Project Mentors:

Dr. Doğan Çörüş

Faculty of Engineering and Natural Sciences

Kadir Has University

Spring 2022

**TABLE OF CONTENTS**

[1 INTRODUCTION 3](#_Toc101703823)

[1.1 Objectives of the Project 3](#_Toc101703824)

[1.2 Flow Chart 3](#_Toc101703825)

[1.3 Pseudo Code 7](#_Toc101703826)

[2 C SCRIPT OF THE PROGRAM 9](#_Toc101703827)

[3 SIMULATIONS 14](#_Toc101703828)

[4 CONCLUSION 23](#_Toc101703829)

# **1 INTRODUCTION**

Voting has been a very common application that is used all around the world for identifying the agreed idea among people. With advanced technology and computer science, it is also been applied in computer programs. In this project, we aimed to create a program that people can login and vote for an issue. In the oncoming section, we are going to explain the main objectives of our project.

## **1.1 Objectives of the Project**

We are required to create a program that people can login with their ID’s and passwords to vote about an issue. We can understand that we need a password and ID matching system in our program. Users can vote yes, no, or they can stay abstain. If the majority of the users agreed on yes or no, the program needs to terminate itself without asking other users since it is pointless. Users should be able to change their passwords and votes if they want. Also, we need to write the votes to a file to be able to retrieve the last state of the vote, if the program shuts down unintendedly. To be able to understand the process of the required algorithm for the program, we need to write a pseudo code and make a flow chart, and we are going to provide them in the next sections.

## **1.2 Flow Chart**

Flow charts are very useful method to understand the required algorithm for the program that we suppose to implement. You can see the flow chart of our algorithm that we created in figures below.

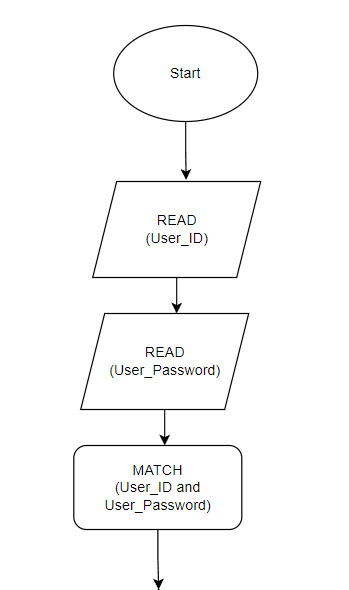


Figure 1.2.1: Beginning of the algorithm

In the beginning of the algorithm, we need to know users’ real ID’s and passwords and we need to match. Figure 1.2.1 represents this idea.

In Figure 1.2.2, you can see that we used a for loop to be able ask every user in the system. We need to create a user interface for every user until we do not need more people to vote.

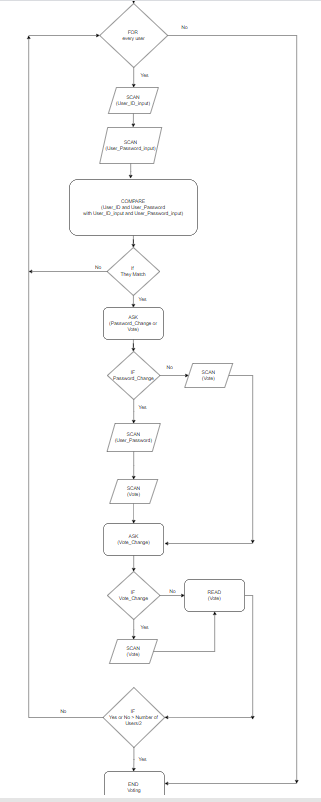


Figure 1.2.2: For Loop

At the end of the algorithm, we need to print the result according to the number of votes. In Figure 1.2.3, you can see the end of our flow chart.

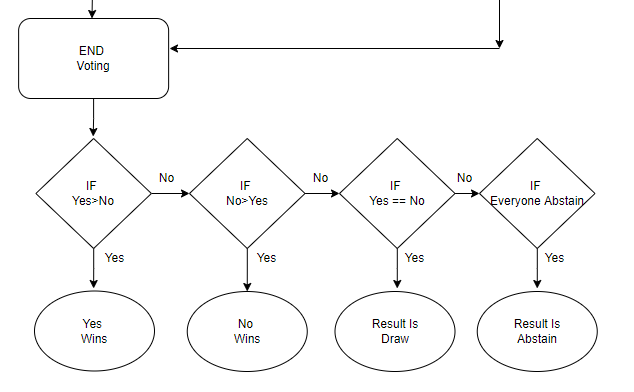


Figure 1.2.3: End of the flow chart

In this section, we prepared a flow chart to understand the basics of the algorithm. According to the new bugs that we may encounter while coding, there might be small differences between the flow chart and the code. In the next section, we are going to put this algorithm into the form of pseudo code.

## **1.3 Pseudo Code**

We have prepared a pseudo code that may help us understand the required algorithm and put it to the computer language easier. We utilized the flow chart and wrote the pseudo code that you can see in the below.

//pseoudo code of the algorithm

READ(User\_ID);

READ(User\_Password);

MATCH(User\_ID and User\_Password);

FOR every user

SCAN(User\_ID\_input);

SCAN(User\_Password\_input);

COMPARE(User\_ID and User\_Password with User\_ID\_input and User\_Password\_input);

IF they match

ASK(password\_change or vote);

IF password\_change;

SCAN(User\_Password);

SCAN(vote);

IF vote

SCAN(vote);

ASK(vote\_change);

IF yes

SCAN(vote);

READ(vote);

IF yes or no > number of users/2

END voting;

END FOR;

IF yes>no

yes wins;

IF no>yes

no wins;

IF yes == no

draw;

IF everyone abstain

result is abstain;

By utilizing the pseudo code, we wrote the C script for our program. In the next section, I am going to provide the C script and explain it.

# **2 C SCRIPT OF THE PROGRAM**

We have prepared a flow chart and written a pseudo code to be able to understand the logic of the algorithm that we need to create our program. Then, we wrote our script to create the program. In this chapter, we are going to explain the basics of our script.

In this project, we need to store votes and passwords to a file to be able to retrieve them if the program shuts down suddenly. In the first part of our code, we defined the libraries, some variables that we are going to use, and open the files as you can see in Figure 2.1.

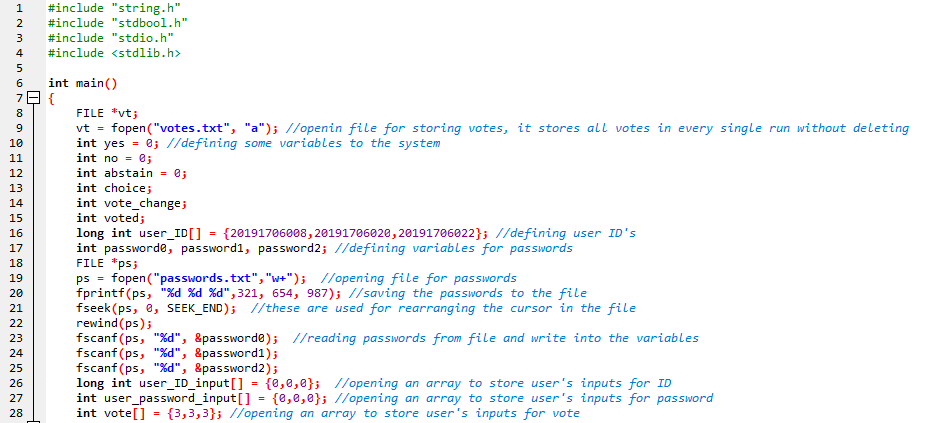


Figure 2.1: The first part of our code

We are writing the passwords to a file to be able to update them later when user wants to change it. Also, we are storing all votes to avoid losing data. fopen() function is used for opening the file. fseek() and rewind() functions are used for moving the cursor in the text file that arranges where the input is going to be written. All passwords are going to be written in the file, and read from there and saved into integers with the functions of fprintf() and fscanf() respectively. Users’ inputs are going to be saved in arrays to compare them with the registered ID’s and passwords.

In the second part of our code, we created a for loop. This loop is created for make every user use the program. You can see the second part of our code in Figure 2.2.

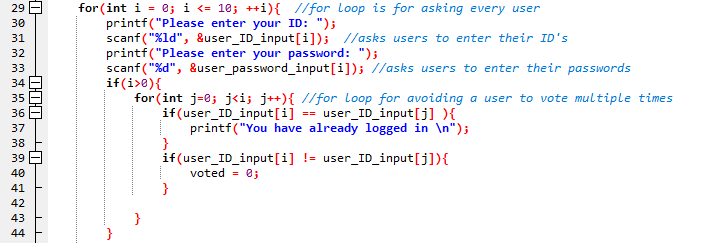


Figure 2.2: Beginning of the for loop

In this part, we basically ask the users enter their ID’s and passwords by using printf() and scanf() functions. Then, we created another for loop to check that if the same ID has been logged in to the system before. If it has, the user is not allowed to enter to the interface. In the third part, we are checking if the password and ID matches correctly or not, and if the user did not login before; user can reach the interface. You can see the password matching and user interface system part of the code in Figure 2.3.

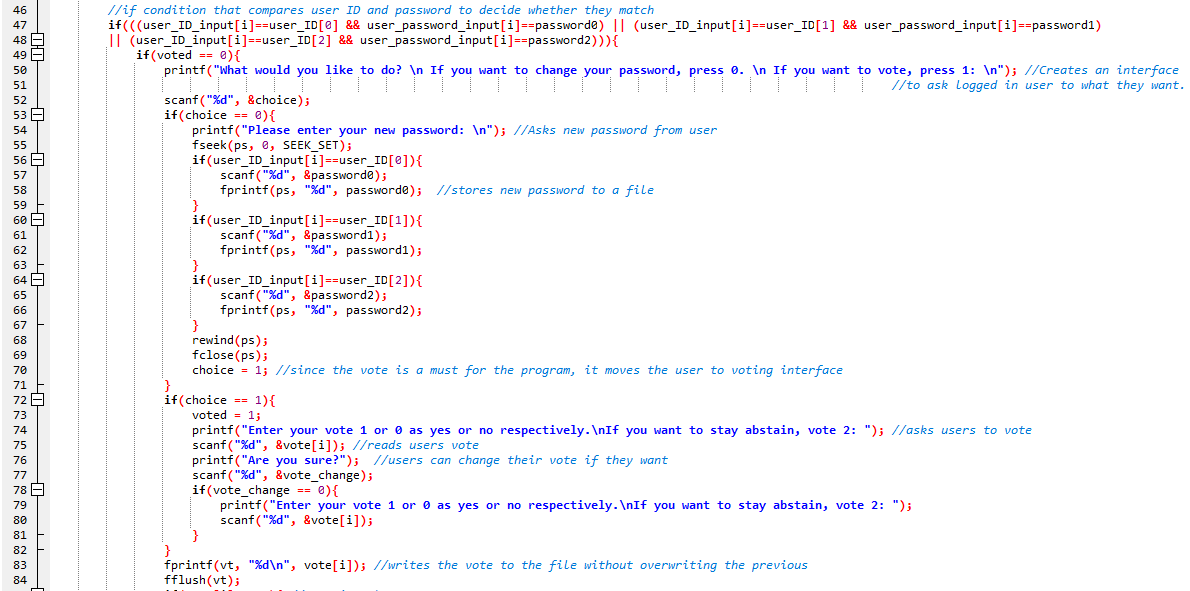


Figure 2.3: User interface

We have an if condition that includes logical operations that compares the input ID and passwords with the registered ID and passwords. If it matches, user login to the system. Then, we create an interface by asking the user what they want, vote or changing password. If user choose changing password, we are asking them to enter new password and save it to the file again by fprintf(). After that, users are directly guided to the voting process. Here, users vote about an issue with pressing 1,0 or 2 as meaning of yes, no or abstain respectively, and the votes have been stored in a file with fprintf() function. fflush() function is for avoiding the overwriting to another vote.

In the fourth part, we are counting the votes according the users’ inputs. If any input apart from 1,0 or 2 entered to the system, program does not allow that vote. You can see the vote counting system in Figure 2.4.

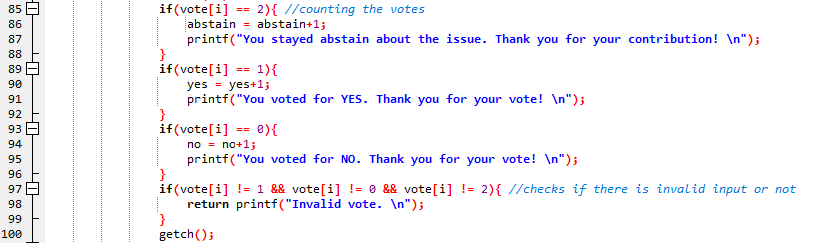


Figure 2.4: Vote counting

In every loop, it checks whether enough people voted or not. If greater than half of the users choose one of the votes, the program prints the result without needing to ask to the other users. If not, it waits for the last user’s vote to print the result. You can see this part in Figure 2.5.

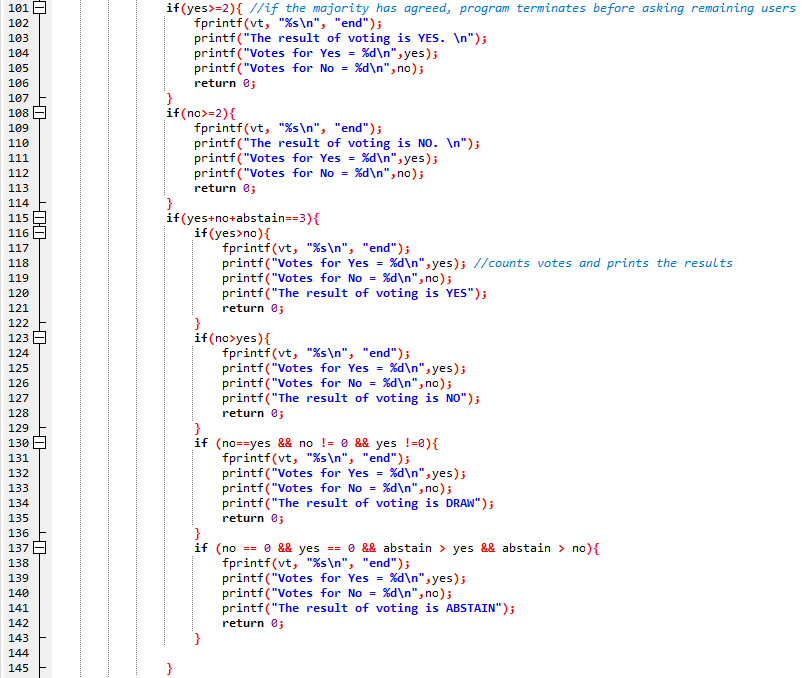


Figure 2.5: Printing the result

If yes votes are more than no, the program prints yes won; if no votes are more than yes, it prints no won. If the number of yes and no votes are equal, the program prints draw. If everyone stays abstain, then it decides that the result of the voting is abstain according to our script.

As the last part of our code, we print password or ID is wrong if user enters wrong couple. After that, for loop ends, and we close all files as you can see in Figure 2.6

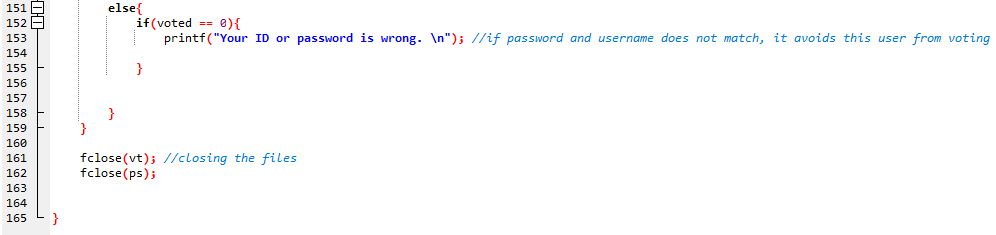


Figure 2.6: End of the code

In this chapter, we have covered our script. In the next chapter, we are going to simulate the program and define what we could achieve and what we could not.

# **3 SIMULATIONS**

In this chapter, we are going to simulate our program by running and using it. Simulation results will give us what our program is capable to do or not.

In Figure 3.1, you can see that when we run the program, it asks the ID of user.

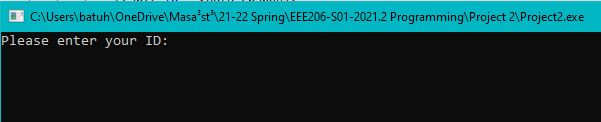


Figure 3.1: Running the program

At that point, we should check the text file that we suppose to store our passwords. You can see inside of the file in Figure 3.2.

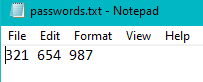


Figure 3.2: Inside of the password.txt

We can see that we successfully store the passwords to the file. Now we have to enter user ID and password.

If we enter the ID and password correctly, the program asks user to choose what it wants to do as you can see in Figure 3.3. That means, we also successfully read the password from the file.

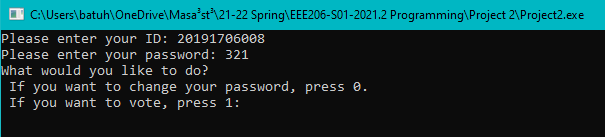


Figure 3.3: Successful login

If the user enters wrong password, the program asks again as you can see in Figure 3.4.

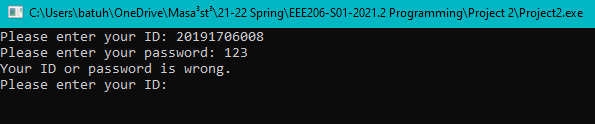


Figure 3.4: Failed login

If user chooses changing password, the program asks to enter a new password. When the user sets the new password, the program asks the vote directly, since there is no need for going back to the interface.

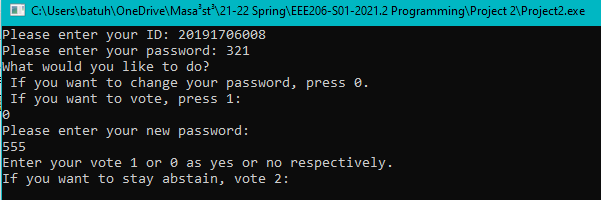


Figure 3.5: Entering new password

Now, we should check the file to see whether the new password is stored in the same position. You can see the inside of the updated text file in Figure 3.6.

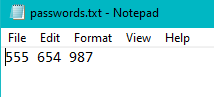


Figure 3.6: Inside of the updated text file

As we can see in Figure 3.6, we managed to update the text file.

After the user votes, the system asks to user whether they want to change their password or not. If the users want to change their vote, they can do it as you can see in Figure 3.7

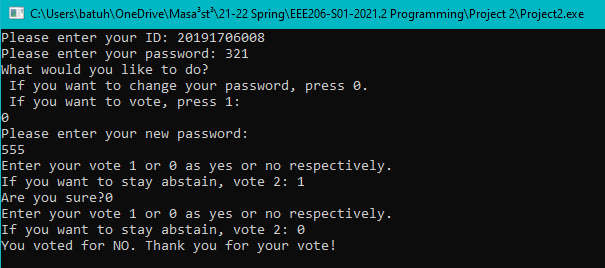


Figure 3.7: Voting process

After that point, one loop for one user has been completed. It asks other users to vote. If the majority agrees upon one vote, the program prints the result without asking other users as you can see in Figure 3.8.

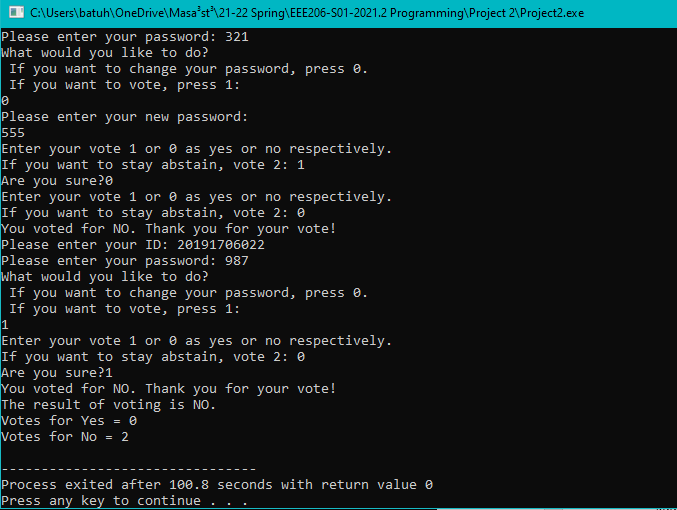


Figure 3.8: Majority agrees upon one vote

The program would ask to the last user for voting if it was necessary as you can see in Figure 3.9.

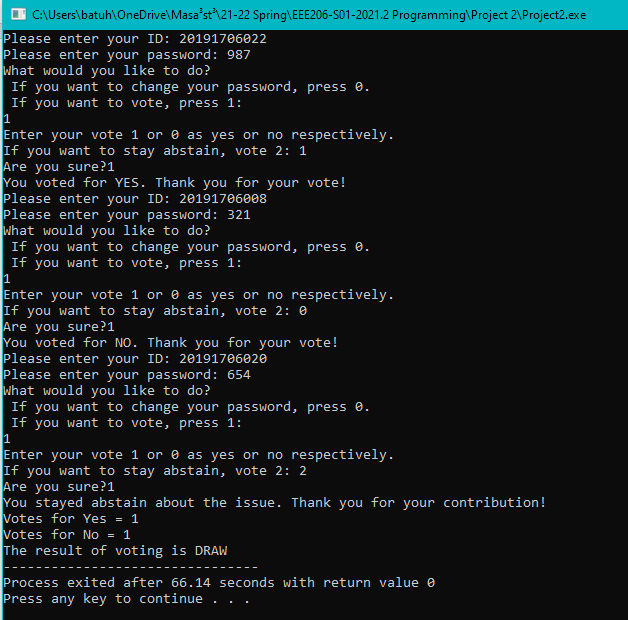


Figure 3.9: Last user is voting

If every user stays abstain, the voting results with abstain as you can see in Figure 3.10.

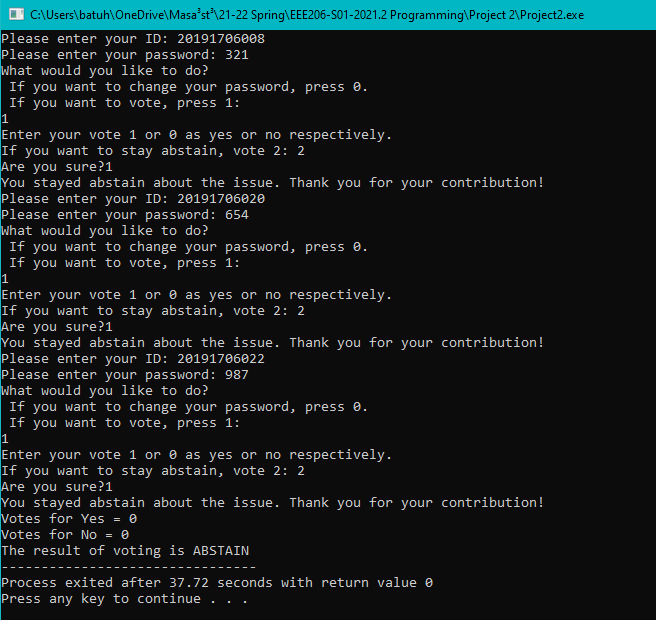


Figure 3.10: Abstain result

If even only one user votes for yes or no, the abstain votes will not be considered for defining the winner of the voting, as you can see in Figure 3.11.

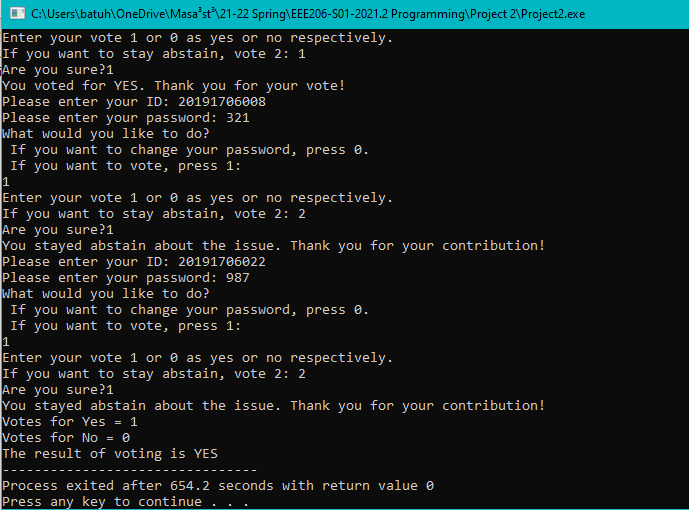


Figure 3.11: Condition of only one user votes for yes or no

A lot of user voted during the simulations. We can check the votes.txt to see whether the votes are stored in a file or not. In Figure 3.12, you can see the votes.txt file.

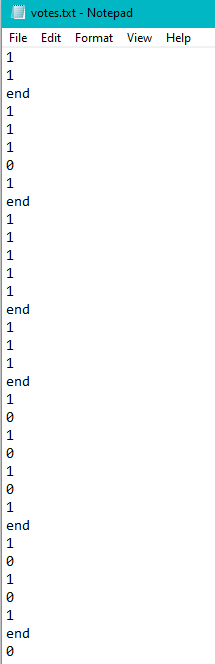


Figure 3.12: Stored votes

As we can see, we have achieved to store all votes. If there is “end” after three votes, that means that voting is completed.

We should also simulate that to see whether the system blocks a user when they try to login second time. In Figure 3.13, you can see the simulation.

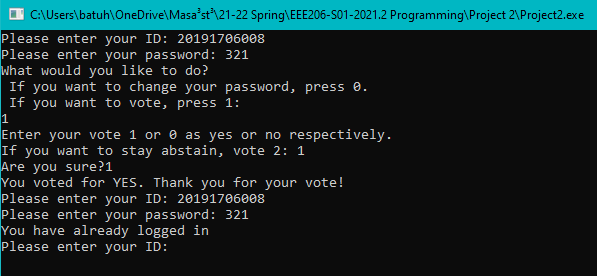


Figure 3.13: Same user tries to login again

We can see that password-ID matching system, user interface, voting process works without any error. However, we could not achieve to store new passwords to the file properly. The problem is, only one user can store the new password, but it always overwrites the first string in the file. When we look at the Figure 3.6, you can see that first user whose ID is 20191706008 could change its password in the file. However, if another user tries to change the password, it will also change the first user’s password in the file. This happens only when we try to write it to the file. The new passwords are saved in their variables in proper places inside of the script.

# **4 CONCLUSION**

In this project, we needed a program that asks people to their votes about an issue. For better understanding of the algorithm, we created a flow chart and pseudo code. By utilizing them, we wrote our script with using C language. According to the results of the simulation, we can say that we achieved to create a program that user can login with ID’s and password to vote about an issue. We also encountered with one error about writing the new passwords to the file as we mentioned about it in detail previously.

This project was very helpful for understanding the mechanism of taking inputs, outputs, loops, updating data and using a file. We learned a lot of new features with new functions.