**Analysis**

I am being asked to create a game involving random numbers, managing and authorising user data. There will be a separate text file which will hold the contents of the authorised users. The program will also be able to create new accounts, delete accounts and resetting passwords for accounts. There will be a maximum of 100 accounts as to avoid a large file size.

The first thing my program will do will run validation checks; these checks will include checking if the text file exists, if it contains the default account and password. It’ll then store the usernames, passwords, codes, and keys into separate array lists, but in the same position in their respective array list as so they can be validated correctly. This process is estimated to take a few milliseconds as to increase efficiency and user friendliness.

The first thing the user will see is a list asking them what they want to do; the options will be as mentioned: Create account, Log-in, delete account or reset password. To select one of the options they will type a number 1 through 4 corresponding to an item on the list, if they mistype the program will state: “invalid option, please choose a number from 1-4”.

If they select: “Create account”, they will be prompted to enter a username, and to enter a password twice. Before writing the username and password to the text file, they will be generated a random 4-digit number that will be used if they ever need to reset their password. This will then be written into the text file in the format: “{username}:{password}”.

If they instead select: “Delete account”, they will be prompted to enter their username, their password twice, and their unique code. The program will then run through the text file putting all the accounts into their respective array list. Once this is done they will be checked against the user’s input, if they do not match the process will be invalidated and therefore cancelled, but if they do match, the username, and password will be deleted from their array lists. Everything else will then be moved forward as to avoid any null errors when writing and reading to the text file. As the arrays had to be changed, the value of a bool “changed” will be set to true, this will cause the file to be re-written before the end of the program.

If they choose: “Reset Password”, the user will be prompted to enter their username and code, as the user doesn’t know their password the only way of authorisation is the code and therefore the account is at higher risk. The program will then run through the password array list looking for the corresponding password. The username will then be cross-checked with the user’s input, cancelling the process if they do not match. If they do match however, they will be prompted to enter a new password twice and advised to store it safely as to not forget it. The value of the bool “changed” will then be set to true, which will trigger a re-writing of the text file before the program ends. The program will then end as to trigger the re-writing of the file.

After the first person logs in, they will ask for the second user to log-in. The second user will see the same screen as the first user, along with the same processes.

Once both users have logged in, the game will begin. Player one will be assigned to the first user, player two will be assigned to the second user. They will be shown the rules of the game, which are as follows: Players one or two will roll their die, they will then add it to their total, if their total is now even, they will add ten points to their score, if their total is instead odd, they will subtract five from their total score. If they roll a double, another dice will be rolled, this will then be added to their score, and they will either lose or gain points. The score of any player should not go below zero, this will be hard coded in to prevent this. After five rounds the player with the highest score wins, if both players have the same total score, they each get one more dice roll until someone wins.

**Here is an example of how the users would see the console: (Anything italicized is not visible in the console, anything underlined is user input)**

First User Account

1 – Log In

2 – Create Account

3 – Delete Account

4 – Reset Password

Enter a number corresponding to an option:

*We are going to run through each selectable option*

4

Enter username to reset:

USERNAME

Enter code:

1234

*If the username and code is false, the following happens:*

Incorrect username/code. Cancelling process.

*Program Ends*

*If the username and code is true, the following happens:*

Enter new password:

Password123

Re-enter password:

Password123

Created new password, make sure to securely store this password as to not forget!

*Now we are going to run through option 3 instead of option 4.*

3

Enter username of account that you want to delete:

Username123

Enter the password of the account:

Password123

Re-enter the password:

Password123

Enter your unique code:

1234

*If the account and its details are valid, they will be deleted, and the following will be shown:*

Successfully deleted account.

*If the account and its details are invalid, the following will be shown:*

The account/details are invalid, cancelling process.

*If the user instead enters 2, the following will be shown:*

2

Enter a new username:

Username123

Enter a new password:

Password123

Re-enter the password:

Password123

*If the username is unique and the passwords are the same, the following will be shown:*Account has been created, your unique code is: 1234. Please keep this safe.

*If the username isn’t unique the following will be shown:*

This username already exists.

*Or if the passwords do not match the following will be shown:*

Passwords don’t match. Cancelling process.

*If everything is valid, the following will be shown:*

Successfully created account, ending program.

*If the user instead entered 1, the following will be shown:*

Enter username:

Username123

Enter password:

Password123

*If the username and password are incorrect, the following will be shown:*

Incorrect username/password. Cancelling process.

*If the username and password are valid, the following will be shown:*

Successful login.

*Once both users have logged in, the game will begin, and the following will be shown:*

To stop the game, type STOP or QUIT at any typable point.

Type roll to roll the dice:

roll

*The user’s input will be checked for either matching “roll”, “stop”, or “quit”. If the input is “stop” or “quit” then the following will be shown.*

Player *ONE/TWO* wants to quit! Player *ONE/TWO* wins!

*If the user’s input was “roll”, they will be shown this: (the numbers will be random)*Player one rolled: 4 and 6.

Their total is now 10 – This is even, player one’s total is now 20.

Player Two: Type roll to roll the dice:

roll

Player Two rolled a: 5 and 2.

Player two’s total is now 7 – This is odd, player two’s total is now 2.

Player one, type roll to roll the dice:

Roll

Player one rolled a: 2 and 2.

This is a double, player one will get one more dice roll.

Player one rolled a: 3.

Player one’s total is now 35 – this is odd, their total is now 30.

Player Two: type roll to roll the dice:

Roll

Player two rolled: 4 and 2.

Their total is now 10 – This is even, their total is now 20.

Player one: type roll to roll the dice:

Player one rolled: 1 and 3.

Their total is now 34 – this is even, their total is now 44.

Player two: type roll to roll the dice:

QUIT

Player two quits – Player One Wins!

The final scores were: 44 – 20.

*The program has ended. If the bool “changed” is set to “true”, the text file will be re-written.*

**Code Diary**

To make my life easier, I simplified the “sout” function whilst also adding an extra addon:

Graphical user interface, text

Description automatically generatedGraphical user interface, text

Description automatically generated

At the start of the program, the accounts file & folder should be checked to see if they exist.

Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generatedGraphical user interface, text, application

Description automatically generated

The “writeToFile” function simply does as it’s called, writes the username and password of a user to the accounts file:

Text

Description automatically generated

Text

Description automatically generatedGraphical user interface, text, application

Description automatically generated

The “writeDefault” function simply writes the default username and password into the accountsFile:

Text

Description automatically generatedGraphical user interface

Description automatically generated with medium confidence

The “readFile” function does as its called, reads each line from the file and puts it into an array list:

Text

Description automatically generated

Text

Description automatically generated

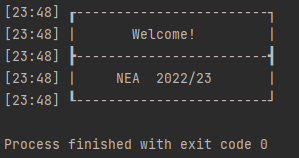
Text

Description automatically generated

Added a visual feature, a “welcome” message, runs in a function as to hide the code:

Text

Description automatically generated



Added “suggestUsername” function, takes users desired username and adds numbers and words (if unchanged by settings).

To do this, I needed to get a list of words usable in usernames, after finding a suitable list, each word had to be formatted, to speed this process up I wrote this little bit of code to do it for me:

Text

Description automatically generated

This is a small sample of what was produced in the console.

“202” is the number of words.

Text

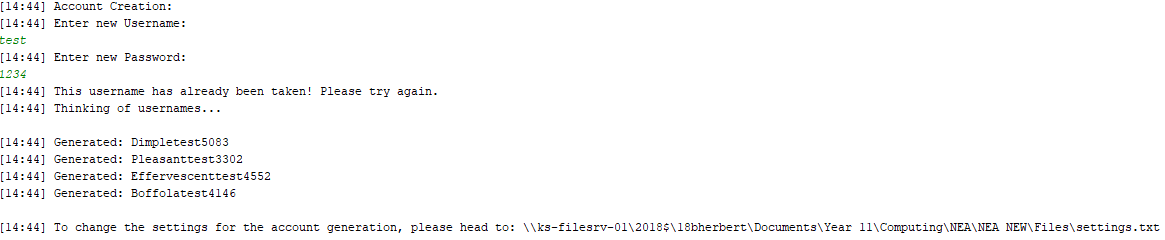
Description automatically generated

Text

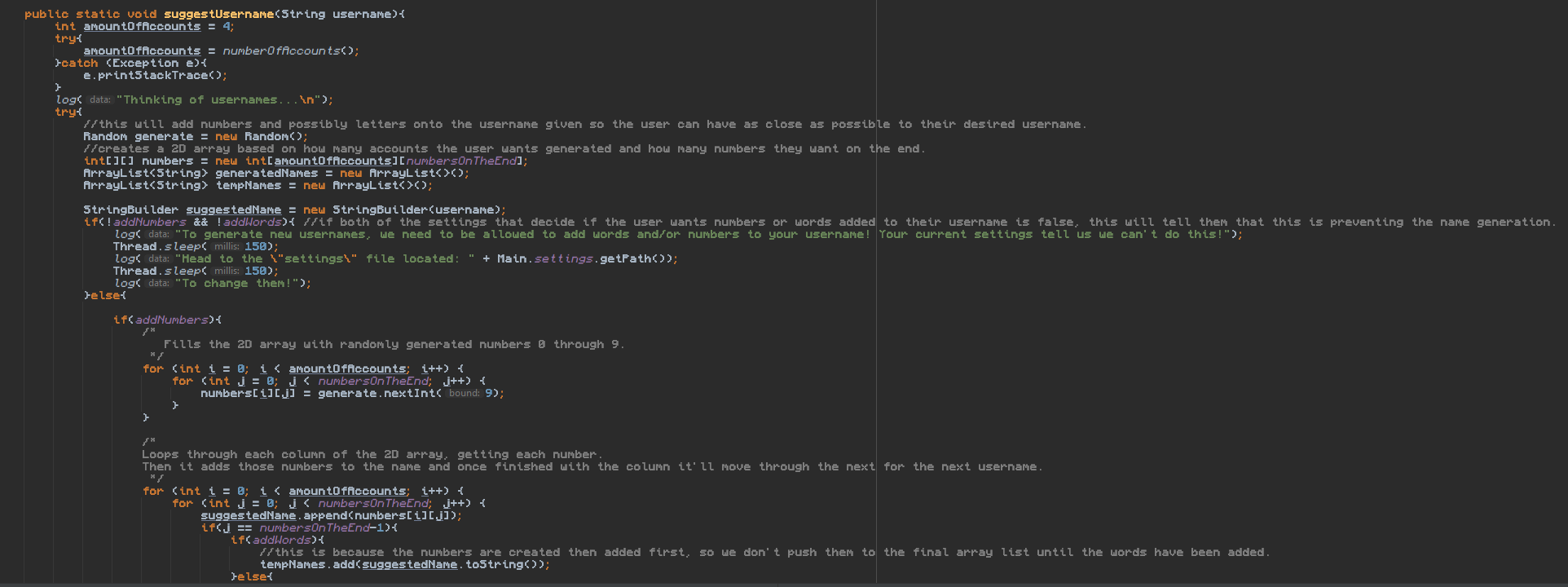
Description automatically generatedAnother small sample of the file produced with the newly formatted words in comparison to the old file:

A close-up of a document

Description automatically generated with low confidence

Producing the names goes like this:

CODE ON NEXT PAGE



Text

Description automatically generated

Due to the new feature, I had to implement the simple way to get usernames and passwords from the “readAccountsFile()” function.

Text

Description automatically generated

To read the settings in from the file, we will use the “readSettingsFile()” function and make it return the values.

Text

Description automatically generated

As we have multiple different files to check for now, the code for the “checkFile()” function has been updated.

Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

This example also applies to the accounts file and works on the same principle.

As we are now creating many files, I had to change the “writeDefault()” function to incorporate these changes.

Please keep in mind that this code is not at all healthy and is something that I am not proud of, it is the only way, as of current, that I could get my idea to function.

Text

Description automatically generated

As we have many file names, I changed the default “readFile()” function to the “readAccountsFile()” for readability.

Text

Description automatically generated

To actually check the details of the users entered username and password, I created a function to call to check them and return a value if valid or invalid:

Text

Description automatically generated

Added the “deleteAccount()” function after finally getting around to doing it.

Text

Description automatically generated

Qr code

Description automatically generatedText

Description automatically generated