

Centre: 12430
Candidate: 8483
Name: Aarogya Yadav

NEA Task Song Quiz

Analysis:

Description of Task:

Noel is making a music game. It works by choosing random songs of random artists from a list of data and displays the artist's name and the first letter of each word in the song's name. For Example: "Metallica: M I F" being the song by Metallica called "Moth into flame". The user then has two tries to guess the song's name via input, if they get the guess of the song name ,correct on the first try they get 3 points , if they get the guess of the song name on the second try they get 1 point and if they get the guess of the song name wrong twice : it will end the game. As long as the program has not quit the game, the songs will continue to be generated. Only authorised users are allowed to play the game.

Success Criteria:

1. The program requires an authorisation process which allows the user to register or if they have already registered, to login by entering their details.
2. The list of player names and the points scored by those usernames are stored in a file.
3. The artist names and the songs linked to the artist are stored in a file.
4. The program is then required to select a random artist and song from the songs and artists file.
5. The program must be able to display the name of the artist and the name of the song in the form that the description of the task states. (In the format: "Metallica: M I F" being the song by Metallica called "Moth into flame".)
6. The program must be able to give the user two guesses at guessing the name of the song correctly.
7. The program must accumulate the score when the user gets the song guess correct, where if it is the first guess the score accumulated is 3 and if it is the second guess then the score accumulated is 1.
8. If the user gets the guesses wrong both times, the program must record the score and player name of the user and saving it into the list of scores and player names.
9. Then the program must exit the game.
10. After exiting the game, the program must display the user's score and the top 5 scorers in the file of scores and their names.

Plan Of Approach:

1. Import modules such as csv and random. (use of csv files to store songs separately with delimiters and allow for easy access, use of random to allow for the random selection of song for the question)

Song Quiz Programming Project

Centre: 12430

Candidate: 8483

Name: Aarogya Yadav

2. Write song and artist names into the file, which includes at least 5 songs from each artist and has at least 6 artists.
3. Create a login and registration process as a subroutine, which uses a username and password and player name (with validation for the user's input e.g. presence check, length check etc)(Test if login process allows user to be able to play and denied if incorrect)
4. Create a subroutine that reads all of the song names and artist's names from the file, which can be accessed later.
5. Create another subroutine that handles the string manipulation to get first letters and artist names along with the input of the user (being the guess) and validates it. (Test if string manipulation gives correct output)
6. Then have a subroutine that counts up the amount of wrong guesses and points, being ready to end the game and display the score and the player name. (Test if display is correct and the counter does not over count or undercount)
7. The program must have another subroutine that reads from the file with all the scores and player names.(Test if it reads and writes correctly onto each file)
8. Next, have a subroutine that hold the player name and score and adds it to the list of scores and player names.
9. Then the list is sorted with a sorting algorithm of your choice, after which the program writes the list to the file and then uploads and outputs the top five scores and their player names. (If the file has written to the file , should be tested)
10. Lastly, give the user the option to close the program or try again (with validated input over here so that no errors are encountered).

Test	What am I testing?	What Data will I use?	Normal/Boundary/Erroneous	Expected Result	How to test
1	Login process	The username and password (make more specific)	Normal: should allow the user to proceed into the game. Boundary and Erroneous: exits the user from the program.	Proceed into game after outputting that authentication was successful.	Checking the part of the program at the start where an input of username and password are required and then seeing if the code accepts them.
2	String Manipulation	The string from file and the string outputted to user	Normal: The data should be printed in the format where the artist's name and first letters of each word of the song are outputted. Boundary and Erroneous: No values since all are converted into string.	Outputs in correct format	Go into artist and song file, editing the name of one of them and seeing the changes to the output given by the program in the game.
3	Correct Display	The user score and player name	Normal: it should output display in correct format and with no errors, continuing to the next subroutine. Boundary and Erroneous: output in incorrect format.	Output in correct format	Change player name in file and try to get different scores checking that the output at the end is reasonable.
4	Writing to files	Files	Normal: Write in data with no errors and in correct format Boundary and Erroneous:	Write in correct format	Check if registration details and user scores

Song Quiz Programming Project

Centre: 12430

Candidate: 8483

Name: Aarogya Yadav

			Written with incorrect format or encounter an error		can take in different values.
--	--	--	--	--	----------------------------------

This program would then be based on a simple display, which displays the artist's name, and the first letters of the words in the song name, allowing the user to input their guess until they lose and it displays the final messages of their score, along with the top 5 scorers then a last input from the user to either close the program or try the game again.

Design

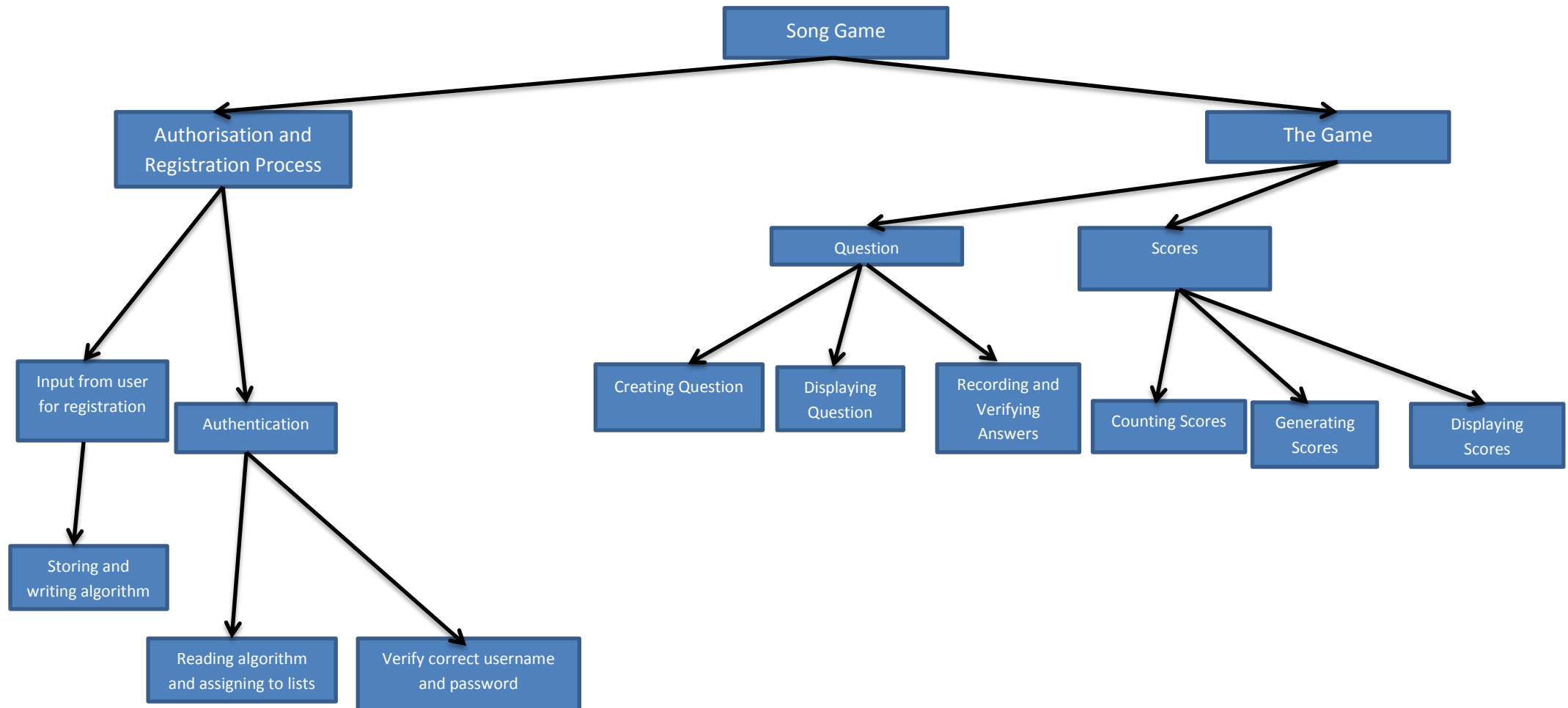
Top-Down Diagram Decomposition

Song Quiz Programming Project

Centre: 12430

Candidate: 8483

Name: Aarogya Yadav



Centre: 12430
 Candidate: 8483
 Name: Aarogya Yadav

Pseudocode for each section:

Authentication & Registration Process:

LOGIN :

This bit of pseudocode sets the variable “login” to False to be turned True when the user successfully logs in. Then code will reads the file for past details, then splitting each line of the file by the comma delimiter and storing the first item as the username, the second item as the password and the third item as player name in separate lists. Then it asks for the username and if it exists then the code will ask for the password which would then be authenticated using the index of the username since the password allocated to the username would be at the same index in its list. If either username or password is incorrect, the program is quit.

```

FUNCTION loginProcess()
    login=False
    file= openRead("UserDetails.txt")
    usernames=[]
    passwords=[]
    playerNames=[]
    WHILE NOT file.endOfFile()           //while it is not the end of the file
        contents= file.readLine()
        splitContents= contents.split(",") //split the contents at every comma
        usernames.append(splitContents[0])
        passwords.append(splitContents[1])
        playerNames.append(splitContents[2])
    ENDWHILE
    file.close()
    user= INPUT ("What is your username?")
    FOR i = 0 to usernames.length -1
        IF user == usernames[i] then
            passw= INPUT("What is your password?")
            iOfPass= usernames.index(user) //iOfPass = index of password
            password= passwords[iOfPass]
            IF passw == password then
                OUTPUT ("You have logged in!")
                playerName= playerNames[iOfPass]
                OUTPUT("Welcome", playerName,"!")
                RETURN
            ELSE then
                OUTPUT(" Incorrect Password")
                Exit program
        
```

Centre: 12430

Candidate: 8483

Name: Aarogya Yadav

```

ENDIF
ELSE then
    OUTPUT("Invalid Username")
    Exit program
ENDIF
NEXT i

```

REGISTRATION:

The pseudocode here reads the file as well, splitting each line of the file by the comma delimiter, where the first item is put into the usernames list, the second item is put into the passwords list and the third item is put into the playerNames list. The user is required to type their name, username and password (after being validated) which will be added to the lists and written back onto the file in the correct format. Then the user is returned to the login process and asked to login.

```

FUNCTION registrationProcess()
    file= openRead("UserDetails.txt")
    usernames=[]
    passwords=[]
    playerNames=[]
    WHILE NOT file.endOfFile()           //while it is not the end of the file
        contents= file.readLine()
        splitContents= contents.split(",")           //contents are split at every comma
        usernames.append(splitContents[0])
        passwords.append(splitContents[1])
        playerNames.append(splitContents[2])
    ENDWHILE
    file.close()
    playerName= INPUT(" What is your name?")
    (Validating #1)
    playerUsername= INPUT("What would you like your username to be?")
    (Validating #2)
    playerPassword= INPUT("What would you like your password to be?")
    usernames.append(playerUsername)
    passwords.append(playerPassword)
    playerNames.append(playerName)
    File=openWrite("UserDetails.txt")
    playerDetails=[]
    FOR x= 0 to usernames.length-1
        playerDetails.append(usernames[x])
    NEXT x
    FOR y = 0 to passwords.length-1
        iOfUD= passwords.index(passwords[y])           // iOfUD = index of user details
        playerDetails[iOfUD]= playerDetails[iOfUD] + "," + passwords[y]
    NEXT y

```

Centre: 12430

Candidate: 8483

Name: Aarogya Yadav

```
FOR z = 0 to playerNames.length-1
    iOfUD= playerNames.index(playerNames[z])
    playerDetails[iOfUD]=playerDetails[iOfUD] + "," + playerNames[z]
NEXT z
FOR i = 0 to playerDetails.length-1
    File.WriteLine(playerDetails[i])
NEXT i
File.close()
loginProcess()
```

MENU

This section of pseudocode requires the user to make a choice to either register or login, by the use of an option between the numbers 1 or 2 taking them to the allocated subroutine.

```
FUNCTION aAndR()          //aAndR = authentication and registration
    loginChoice=INPUT("Choose one of the options from below by typing its corresponding
    number:
        1. Login
        2. Register
        ::")
    WHILE loginChoice is not "1" or "2"
        loginChoice=INPUT("Choose one of the options from below by typing its
        corresponding number:
            1. Login
            2. Register
            ::")
    IF loginChoice == "1" then
        nameOfPlayer=loginProcess()
    ELSEIF loginChoice == "2" then
        nameOfPlayer=registrationProcess()
```

VALIDATION

These bits of pseudocode make sure the user's input is reasonable by making sure there are no duplicate usernames or playernames and also making sure that they do not leave any of the details blank.

VALIDATING #1:

```
WHILE playerName == "":
    playerName= INPUT(" Please input your name:")
    FOR i = 0 to playerNames.length -1
```

Song Quiz Programming Project

Centre: 12430

Candidate: 8483

Name: Aarogya Yadav

```
        IF playerNames[i] == playerName:
            OUTPUT ("This name already exists")
            duplicate= TRUE
        ELSE:
            duplicate=FALSE
        ENDIF
    IF duplicate == TRUE:
        playerName= ""
    ENDIF
NEXT i
ENDWHILE
```

VALIDATING #2:

```
WHILE playerUsername == "":
    playerUsername= INPUT(" Please input your username:")
    FOR i = 0 to playerUsernames.length-1
        IF usernames[i] == playerName:
            OUTPUT ("This username already exists")
            duplicate= TRUE
        ELSE:
            duplicate = FALSE
        IF duplicate == TRUE:
            playerUsername= ""
        NEXT i
```

IPOD TABLE

INPUT	PROCESS	OUTPUT	DECISION
username and password	Loop and read file to match the input on the file.	The game	If the user input matches the data on the file then output the game, if not then end program.
playerName, playerUsername and playerPassword	Assign values to variables while checking there are no duplicates	Login process	If reasonable allow them to login but if not ask for input again.

VARIABLE TABLE

VARIABLE NAME	DATA TYPE	PURPOSE
login	Boolean	To hold the value for if the user has logged in
file	File header	To hold the file to be called upon in read mode
usernames	List	To hold the usernames from the file or from the user's input
passwords	List	To hold passwords from the file or from the user's input

Song Quiz Programming Project

Centre: 12430
 Candidate: 8483
 Name: Aarogya Yadav

content	String	To hold onto each line from the file
splitcontents	List	To hold the different words in the lines separately (username, password and player name)
user	string	To hold the user's input for username to be authenticated
passw	String	To hold the user's input for password to be authenticated
iOfPass	Integer	To hold the index of the real password allocated to the username
password	String	To hold the real password allocated to the username
playerNames	List	To hold the player names from the file or from the user's input
playerName	String	To hold the user's input of their name
playerUsername	String	To hold the user's input of their desired username
playerPassword	String	To hold the user's input of their desired password
File	File header	To hold the file to be called upon in write mode
playerDetails	List	To hold a list of each user's data to be written to the file
iOfUD	Integer	To hold the index of each user's details to go from the start of the list to the end
loginChoice	string	To hold the first choice of the user being either "1" or "2"
nameOfPlayer	string	To hold the player's name for later use
duplicate	boolean	To turn true if there is a duplicate username or player name in the file.

VALIDATION

VALIDATION TYPE	WHERE	REASON
Presence check	playerName, playerUsername and playerPassword	To make sure no details are left empty and to make sure playerName and playerUsername are not duplicated

TEST PLAN

TEST	TEST TYPE	TEST DATA	REASON	EXPECTED OUTCOME
------	-----------	-----------	--------	------------------

Centre: 12430

Candidate: 8483

Name: Aarogya Yadav

NUMBER				
1	Valid	user and passw (username and password)	To see if a valid username and password get through	It should bypass the authentication process and proceed into game.
2	Invalid	user and passw (username and password)	To see if an invalid username and password get through	It should exit the program immediately.
3	Duplicate	playerName and playerUsername	To see if a duplicate input gets through	It should ask for another input.

The Game

This section of pseudocode is the game. This first reads the file for the artists and songs, appending them to two lists. These two lists are then copied so that any changes do not change the real data; allowing it to be used again. The question is made by using the random module which generates a random index in the duplicate song list, which is used to choose the artist and song, and then it manipulates the song name's string by getting only the first letters, concatenating it with the artist name to be outputted. Then the index is popped from the duplicate list so that the same song is not chosen again. Next, the question is outputted and the user's input is stored in a variable. The answer is checked for being correct or incorrect, taking the appropriate response (either asking again or accumulating score). This continues until game over which is when they get their answer incorrect twice in a row.

```

FUNCTION theGame()
    file= openRead("Songs.txt")
    artists=[]
    songs=[]
    WHILE NOT file.endOfFile()
        contents= file.readLine()
        splitContents= contents.split(",")    //contents are split at every comma
        artists.append(splitContents[0])
        songs.append(splitContents[1])
    ENDWHILE
    file.close()
    unchosenArtists=artists
    unchosenSongs=songs

```

Song Quiz Programming Project

Centre: 12430

Candidate: 8483

Name: Aarogya Yadav

gameOver= FALSE

WHILE gameOver == FALSE

IF unchosenArtists.length == 0 then

 unchosenArtists=artists

 unchosenSongs=songs

ENDIF

randomNumber= Random integer from 0 to unchosenSongs.length

randomSong= unchosenSongs[randomNumber]

SongFirstLetters= ""

FOR i= 0 to randomSong.length-1

 IF i== 0 then

 SongFirstLetters+= randomSong [i]

 ELSEIF randomSong[i] == " " then

 SongFirstLetters+= randomSong [i]

 SongFirstLetters+= randomSong [i+1]

 ENDIF

NEXT i

SongArtist=unchosenArtists[randomNumber]

question= SongArtist + ":" + SongFirstLetters

unchosenArtists.pop(randomNumber)

unchosenSongs.pop(randomNumber)

answer= INPUT(question, "Type the song's full name here (without missing capitals for each word):")

IF answer == randomSong then

 score+=3

ELSE then

 OUTPUT("That was incorrect. You have one more chance.")

answer= INPUT(question, "Type the song's full name here (without missing capitals for each word):")

IF answer == randomSong then

 score+=1

ELSE then

 OUTPUT("That was incorrect. You lose.")

 OUTPUT("Game Over")

 gameOver= True

ENDIF

ENDIF

ENDWHILE

OUTPUT("You scored",score, "points!")

file2= openRead("playerScores.txt")

playerNames=[]

Song Quiz Programming Project

Centre: 12430

Candidate: 8483

Name: Aarogya Yadav

```
scores=[]
```

```
WHILE NOT file.endOfFile()
```

```
    contents= file2.readLine()
```

```
    splitContents= contents.split(",")           //contents are split at every comma
```

```
    playerNames.append(splitContents[0])
```

```
    scores.append(splitContents[1])
```

```
ENDWHILE
```

```
file2.close()
```

```
playerNames.append(nameOfPlayer)
```

```
scores.append(score)
```

```
Sorted=False
```

```
IF scores.length< 2 then
```

```
    Sorted=True
```

```
WHILE Sorted== False
```

```
    Swap=0
```

```
    FOR i = 0 to scores.length -2
```

```
        IF i== scores.length -1
```

```
            BREAK
```

```
        ELSEIF scores[i] < scores[i+1]:
```

```
            tempScore= scores[i]
```

```
            scores[i]=scores[i+1]
```

```
            scores[i+1]=tempScore
```

```
            tempPlayerName= playerNames[i]
```

```
            playerNames[i]= playerNames[i+1]
```

```
            playerNames[i+1]= tempPlayerName
```

```
            Swap+=1
```

```
        ENDIF
```

```
    NEXT i
```

```
    IF Swap == 0 then
```

```
        Sorted = True
```

```
IF playerNames.length <5:
```

```
    OUTPUT("These are the top", playerNames.length," scores!")
```

```
    FOR i= 0 to playerNames.length -1
```

```
        OUTPUT( playerNames[i] + str(scores[i]))
```

```
    NEXT i
```

```
ELSE
```

```
    OUTPUT(" These are the top 5 scores!")
```

```
    FOR i= 0 to 5
```

```
        OUTPUT( playerNames[i] + str(scores[i]))
```

```
    NEXT i
```

```
File= openWrite("playerScores.txt")
```

```
playerScoreDetails=[]
```

```
FOR x = 0 to playerNames.length-1
```

```
    playerScoreDetails.append(playerNames[x])
```

Song Quiz Programming Project

Centre: 12430

Candidate: 8483

Name: Aarogya Yadav

NEXT x

FOR y = 0 to playerScoreDetails.length-1

playerScoreDetails[y]=playerScoreDetails[y] + “,” + scores[y]

NEXT y

FOR i = 0 to playerScoreDetails.length-1

File.WriteLine(playerScoreDetails[i])

theGame()

IPOD TABLE

INPUT	PROCESS	OUTPUT	DECISION
answer	Check is correct answer is inputted by user.	Depending on correct or incorrect take appropriate response.	If correct, accumulate score, if incorrect twice, end game.

VARIABLE TABLE

VARIABLE NAME	DATA TYPE	PURPOSE
file	File header	To hold the songs' file in read mode.
artists	List	To hold the artists from the songs file
songs	List	To hold the song names from the songs file
contents	String	To hold each line of the file
splitContents	List	To hold each line of the file split by a comma into a list
unchosenArtists	List	To hold all the unchosen artists in the game
unchosenSongs	List	To hold all the unchosen songs in the game
gameOver	Boolean	To hold the value of true when the game ends
randomNumber	Integer	To choose a random number to be used as an index in the range of the list's length
randomSong	String	To hold the song chosen by the random number
SongFirstLetters	String	To hold the first letters of the song
SongArtist	String	To hold the artist's name
question	String	To hold the artist's name and song's first letters concatenated to be outputted
answer	String	To hold the input of the user being their answer

Song Quiz Programming Project

Centre: 12430

Candidate: 8483

Name: Aarogya Yadav

file2	File header	To hold the playerScores file in read mode
playerNames	List	To hold the playerNames from the list of the players in the game
scores	List	To hold the scores from the list of the players in the game
Sorted	Boolean	To be True when the list did not have a single swap occur
Swap	Integer	To hold the number of swaps that occur each time the list is looped through
tempScore	Integer	To hold the contents of the score that will be overwritten so that it is not lost and can be used
tempPlayerName	String	To hold the contents of the player name that will be overwritten so that it is not lost and can be used
File	File header	To hold the playerScores file in write mode
playerScoreDetails	List	To hold the each player's score and name to be written back into the file

TEST PLAN

TEST NUMBER	TEST TYPE	TEST DATA	REASON	EXPECTED OUTCOME
1	Valid/ Invalid	answer	To make sure the program continues on both valid and invalid input	The program continues to the next question or ends the game.

Development

Login and Registration

Used as a reference:

<https://www.pythonforbeginners.com/random/how-to-use-the-random-module-in-python>

To start, I import a module called "random" to be used later in the game. Then I create a subroutine called "loginProcess" being for the user to be able to authenticate them. Firstly, I globalise the variable "login" so that it can be accessed outside of this function and set it to

Centre: 12430

Candidate: 8483

Name: Aarogya Yadav

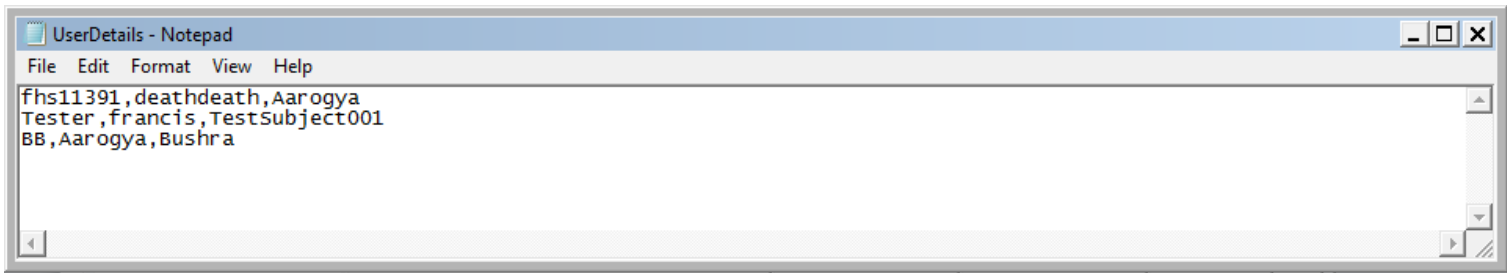
False, then I print 2 newline characters and then a string that shows the user they are about to login. I then open a file named "UserDetails.txt" in read mode, and then I create 3 empty lists to be appended to being "usernames", "passwords" and "playerNames". I then initialize the "contents" variable to any value, in this case "True" so that the while loop can be entered, as it has the condition to run while contents is not empty. The variable "contents" is set to each line in the text file, going through them one by one and stripping them of Whitespace characters etc. Then, I split the contents by the delimiter being a comma into username, password and player name to be appended to the allocated lists from before. Assigning details in this form allows for all the details of one person to have the same index so that they can be referenced later. When the loop reaches the end of the file, there will be a 1 item list containing the end of file character and that is when the code breaks out of the loop. The file is then closed. Next, the user is asked for their details where the username is stored and then the usernames list is looped through, to find a matching username. If a username does exist, the code will ask for their password and use the index of the username to find the corresponding password for that user to compare with the user input to check if it is correct, if the username does not exist then the program is quit with a string being printed to say what detail they got incorrect, this also happens if the password is incorrect. If, however the username and password are both correct. The program will print a message saying "You have logged in!" and another string saying "Welcome" and their player name while setting the variable "login" to True.

This can be seen below:

```
import random
#The line above imports the random module, to be used to generate random numbers later on.
def loginProcess():
    global login
    login=False
    print("\n\n")
    print("Login Process:")
    file= open("UserDetails.txt","r")
    usernames=[]
    passwords=[]
    playerNames=[]
    contents= True
    while contents != "":
        contents=file.readline()
        contents=contents.strip()
        splitContents=contents.split(",")
        if len(splitContents)==1:
            break
        usernames.append(splitContents[0])
        passwords.append(splitContents[1])
        playerNames.append(splitContents[2])
    file.close()
    user= input("What is your username?\n")
    for i in usernames:
        if i ==user:
            passw=input("What is your password?\n")
            iOfPass= usernames.index(i)
            password= passwords[iOfPass]
            if passw == password:
                print("You have logged in!")
                playerName=playerNames[iOfPass]
                global nameOfPlayer
                nameOfPlayer=playerName
                print("Welcome",playerName,"! \n")
                login= True
            else:
                print("Your password is incorrect")
                quit()

#The subroutine above reads the "UserDetails" file to act as a reference for usernames and passwords to then authenticate the user's input of their username
#and password, outputting a string if the login is successful.
```

Centre: 12430
 Candidate: 8483
 Name: Aarogya Yadav



```
Choose one of the options from below by typing its corresponding number:
1. Login
2. Register
:1
```

```
Login Process:
What is your username?
fhs11391
What is your password?
deathdeath
You have logged in!
Welcome Aarogya !
```

I then create a function called “registrationProcess()” for the registration of the user to create a their user details to authenticate into the game. The subroutine starts by opening a file named “UserDetails.txt” in read mode, to check if any other user details already exist and to

make sure the new user details are not duplicates. The same process is used of looping over each line, splitting it into “username”, “password” and “playerName” and appending them into the corresponding lists so that they all have the same index. The file is closed and a while loop is entered after initializing the variable “playerName” as “” so that the while loop is entered. The while loop has the conditions to execute the code if the variable “playerName” is empty or the variable “duplicate” is set to True. Next, the user is asked for an input of their “playerName” which is then checked so that no duplicate exists. This is done by looping through the “playerNames” list, comparing each item to the user input. If there is a match, the variable “duplicate” is set to True and a string is outputted stating “This name already exists.”, if there is no match then the variable “duplicate” is set to False. When the variable “duplicate” is set to True, the code in the while loop executes again, asking the user for input , but when set to False the code continues by initializing the “playerUsername” variable to “” to be able to enter the following while loop. The following while loop follows the same process as the loop before, checking that the user’s input for username is not empty or a duplicate so that errors do not occur in the authentication process. The user is then asked for an input of their password which is not checked because the password does not need to be checked as it can be anything that the user wants and would not cause errors. The new user details are appended onto the corresponding lists, again at the same index in each list. The file “UserDetails.txt” is now opened in write mode.


```

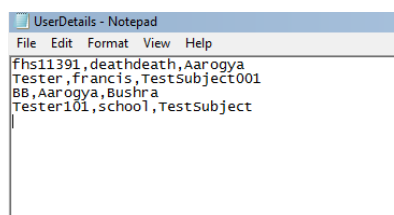
def registrationProcess():
    file=open("UserDetails.txt","r")
    usernames=[]
    passwords=[]
    playerNames=[]
    contents= True
    while contents != "":
        contents=file.readline()
        contents=contents.strip()
        splitContents=contents.split(",")
        if len(splitContents)==1:
            break
        usernames.append(splitContents[0])
        passwords.append(splitContents[1])
        playerNames.append(splitContents[2])
    file.close()
    playerName= ""
    while playerName=="" or duplicate== True:
        playerName= input("What is your name?\n")
        for i in playerNames:
            if i == playerName:
                print("This name already exists.")
                duplicate= True
                break
            else:
                duplicate=False
        playerUsername=""
    while playerUsername =="" or duplicate== True:
        playerUsername=input("What would you like your username to be?\n")
        for i in usernames:
            if i == playerUsername:
                print("This username already exists.")
                duplicate= True
                break
            else:
                duplicate= False
    #Checks for duplicate
    playerPassword= input("What would you like your password to be?\n")
    usernames.append(playerUsername)
    passwords.append(playerPassword)
    playerNames.append(playerName)
    File= open("UserDetails.txt","w")
    playerDetails=[]
    for x in usernames:
        playerDetails.append(x)
    for y in passwords:
        iOfUD= passwords.index(y)
        playerDetails[iOfUD]=playerDetails[iOfUD] +","+ y
    for z in playerNames:
        File= open("UserDetails.txt","a")
        File.write(i)
        File.write("\n")
    File.close()
    loginProcess()
#The subroutine above takes in details from the user and then writes it back onto the file in the correct format with username, password and player name
#separated by commas on each line.
    File.close()

```

A new variable called “playerDetails” is created which will have all the user details appended to it with each item in the list being

“username,password,playerName”.

After appending all the data from the lists “usernames”, “passwords” and “playerNames”, the list “playerDetails” is looped through and is written onto the text file along with a newline character, the first item will be written onto the file so that all previous data is overwritten and the following data is appended onto the file. The file is then closed to save the changes and lastly the subroutine “loginProcess()” is run to get the user to login. This can be seen on the left:



Choose one of the options from below by typing its corresponding number:

1. Login
2. Register

:2

What is your name?

TestSubject

What would you like your username to be?

Tester101

What would you like your password to be?

school

Login Process:

What is your username?

Tester101

What is your password?

school

You have logged in!

Welcome TestSubject !

Centre: 12430
 Candidate: 8483
 Name: Aarogya Yadav

I then created a login decision for the startup of the code in the subroutine “aAndR()” which is a short form for “authentication and registration”. The user is asked to input either “1” or “2” where 1 is a login and 2 is a registration, this is then stored into the variable “loginChoice”. This variable is also validated to only allow the inputs of “1” or “2”, any other input causes the code to ask for the user’s input again.

```
def aAndR():
    loginChoice=input("""Choose one of the options from below by typing its corresponding number:
1. Login
2. Register
: """)
    while loginChoice != "1" and loginChoice != "2":
        loginChoice=input("""Choose one of the options from below by typing its corresponding number:
1. Login
2. Register
: """)
    if loginChoice == "1":
        loginProcess()
    elif loginChoice == "2":
        registrationProcess()
#The subroutine above asks the user for an input of either 1 or 2, where depending on which is chosen, the allocated subroutine is run. For 1, it is the
#loginProcess() subroutine and for 2, it is the registrationProcess() subroutine.
```

The Game

I then create another subroutine, this is the main code for the whole game. It starts with the opening of the file “Songs.txt” in read mode. This text file contains the artist name and the song name separated by a delimiter of a comma. By using the same method of reading the file line by line, splitting each line by the delimiter comma and then appending each item to its allocated list, which in this case would be the empty list “artists” and “songs”. After appending all items so that data on the same line of the file will have the same index again, the file is closed. I then set two variables as empty lists, being “unchosenArtists” and “unchosenSongs”. Then I loop through the “artists” and “songs” lists, and appending each item into the “unchosen...” list: this is done to create duplicates of the lists to be altered so that the original still exists. Then I print a string to give the format of the question that will be answered by the user when the question is outputted. I initialize the variable “gameOver” to False and the variable “score” to 0. The “gameOver” variable being set to false tells the program to run the game until the user has triggered the variable to become true which happens when they get the question incorrect twice in a row. The while loop contains a decision construct which makes sure that when the two “unchosen...” lists have no items in them, they are reset by re-appending all the data from the original lists. I then set a maximum value being the amount of items in the unchosen Songs list -1, we subtract one because indexes start from 0 although the number of items will start from 1, which is used to create a random index from 0 to the last index in the list. This is done using the random module that I imported at the start, which has the function “.randint()” which takes in a range from which it chooses a random number. I utilise this and the maximum number to create a random index in the list which chooses a random song from the list of songs, this is

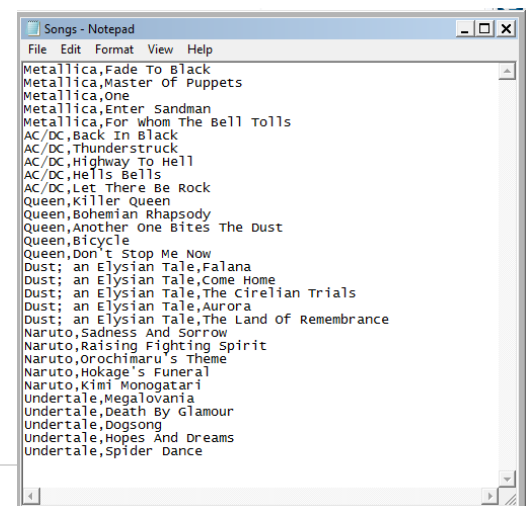
Song Quiz Programming Project

Centre: 12430

Candidate: 8483

Name: Aarogya Yadav

done to make sure that the questions are in a random order. The variable “randomSong” is assigned the value of an item from the “unchosenSongs” list being at the index chosen by the random module in the last variable “randomNumber”. The “SongFirstLetters” is set to an empty list for the letter to be appended to it later. To find the first letter, we loop through the string of the song name obtained from the file. We first append the letter at index 0 by utilising an if statement which will append “i” (the string character that is being looped through) to the variable “SongFirstLetters” if the index of the character is 0 or if the previous character in the string was a whitespace character while also appending the whitespace character so that in the string that is stored in the variable “SongFirstLetters”, each letter is spaced out from one another. Next, I set the variable “SongArtist” to the item at the random index from before in the list “artists” (this is possible because the song and artist name were appended line by line causing them to have the same index in the two different lists.). The variable “question” is set to the concatenation of the “SongArtist” variable, a colon and the “SongFirstLetters” variable. After this, from both the “unchosen...” lists, I pop the item at the random index so that the same song will not be outputted twice as a question. Next, the question is outputted, being part of a string and the variable “answer” is assigned the user input of their answer to the question. The following decision construct then checks if the user input is the same as the original song name from before that was stored in the variable “randomSong”, where if it matches, the scores (which was initialized as 0) is incremented by 3, if it does not match, the user is asked for an input again, after outputting that their last input was incorrect, then this input will also be checked for a match where if it matches on this attempt, the score is incremented by 1 and if it does not match, the program outputs “Game Over” and assigns the variable “gameOver” to True which then exits the while loop. (The game will continue choosing random indexes from the list, making questions and popping from the list, if the user does not get their answer wrong, where if the lists have all of their items popped then the list is reset and the questions are asked again in a random order until game over is triggered, which sets the variable “gameOver” to True.). This code can be seen below:



```
File Edit Format View Help
Metallica,Fade To Black
Metallica,Master of Puppets
Metallica,One
Metallica,Enter Sandman
Metallica,For whom The Bell Tolls
AC/DC,Back In Black
AC/DC,Thunderstruck
AC/DC,Highway To Hell
AC/DC,Hells Bells
AC/DC,Let There Be Rock
Queen,Killer Queen
Queen,Bohemian Rhapsody
Queen,Another One Bites The Dust
Queen,Bicycle
Queen,Don't Stop Me Now
Dust; an Elysian Tale,Falana
Dust; an Elysian Tale,Come Home
Dust; an Elysian Tale,The Círelían Trials
Dust; an Elysian Tale,Aurora
Dust; an Elysian Tale,The Land of Remembrance
Naruto,Sadness And Sorrow
Naruto,Raising Fighting Spirit
Naruto,Orochimaru's Theme
Naruto,Hokage's Funeral
Naruto,Kimi Monogatari
Undertale,Megalovania
Undertale,Death By Glamour
Undertale,Dogsong
Undertale,Hopes And Dreams
Undertale,Spider Dance
```

Centre: 12430

```

def theGame():
    file= open("Songs.txt","r")
    artists=[]
    songs=[]
    contents=True
    while contents != "":
        contents= file.readline()
        contents=contents.strip()
        splitContents= contents.split(",")
        if len(splitContents)==1:
            break
        artists.append(splitContents[0])
        songs.append(splitContents[1])
    file.close()

#The block of code above opens and reads the file names "Songs", separating the artist names and song names and then appending them to two separate lists.
unchosenArtists=[]
unchosenSongs=[]
for i in artists:
    unchosenArtists.append(i)
for i in songs:
    unchosenSongs.append(i)
print("\n Each question will be presented in this form:
      [Artist Name: Song Initials with spaces] \n")
gameOver=False
score=0

#This block of code duplicates the artists and songs lists to be used later. Then the code prints the form of the questions and lastly initializes the
#gameOver and score variables, where gameOver is set to False so that the while loop can be entered. The score is set to 0 to be incremented later.
while gameOver==False:
    if len(unchosenArtists) == 0:
        for i in artists:
            unchosenArtists.append(i)
        for i in songs:
            unchosenSongs.append(i)
    maximumNumber= ((len(unchosenSongs))-1)
    randomNumber=random.randint(0,maximumNumber)
    randomSong= unchosenSongs[randomNumber]
    SongFirstLetters=""
    for i in range(len(randomSong)):
        if i == 0:
            SongFirstLetters+= randomSong[i]
        elif randomSong[i] == " ":
            SongFirstLetters+= " "
            SongFirstLetters+= randomSong[i+1]
    SongArtist= unchosenArtists[randomNumber]
    question= SongArtist+ " " + SongFirstLetters
    unchosenArtists.pop(randomNumber)
    unchosenSongs.pop(randomNumber)

#The following code generates a random index and then makes the question with the randomly chosen song, removing that song from the list so that it cannot
#be chosen again.
print("\n\n")
print("[",question,""]
Type the song's full name here (without missing capitals for each word) \n")
answer= input("Type your answer here:")
if answer == randomSong:
    score+=3
else:
    print("That was incorrect. You have one more chance.")
    print("[",question,"] Type the song's full name here (without missing capitals for each word) \n")
    answer= input("Type your answer here:")

    if answer == randomSong:
        score+=1
    else:
        print("That was incorrect. You lose.")
        print("Game Over")
        gameOver= True

```

In the first attempt at making this subroutine, I did encounter an error which was extremely confusing to me . This error stated that I was giving an empty list to the randint function although I had reset the list to the original list in the code below:

I realized later on that my method of resetting the list by equating the original list to the duplicate would cause the two variable to be interlinked rather than for the duplicate to have all of the original list's items to be copied onto it, "unchosenArtists=artists
unchosenSongs=songs".

Song Quiz Programming Project

Centre: 12430

Candidate: 8483

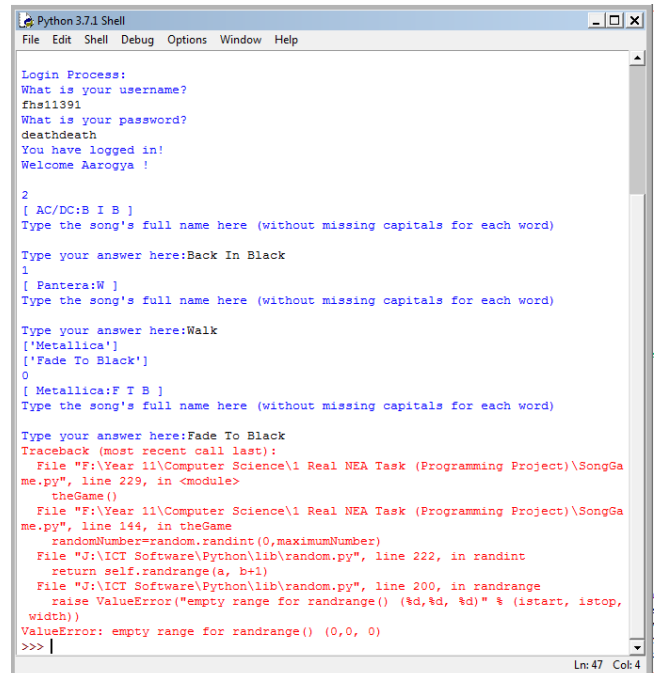
Name: Aarogya Yadav

```
artists.append(splitContents[0])
songs.append(splitContents[1])
file.close()
unchosenArtists=artists
unchosenSongs=songs
gameOver=False
score=0

e gameOver==False:
if len(unchosenArtists) == 1:
    unchosenArtists=artists
    unchosenSongs=songs
    print(unchosenArtists)
    print(unchosenSongs)
maximumNumber= ((len(unchosenSongs))-1)
randomNumber=random.randint(0,maximumNumber)
randomSong= unchosenSongs[randomNumber]
SongFirstLetters=""
for i in range(len(randomSong)):
    if i == 0:
        SongFirstLetters+= randomSong[i]
    elif randomSong[i] == " ":
        SongFirstLetters+= " "
        SongFirstLetters+= randomSong[i+1]
SongArtist= unchosenArtists[randomNumber]
question= SongArtist+ ":" + SongFirstLetters
unchosenArtists.pop(randomNumber)
unchosenSongs.pop(randomNumber)
print(len(unchosenArtists))

print("[",question,""]
: song's full name here (without missing capitals for each word) \n'
answer= input("Type your answer here:")
if answer == randomSong:
    score+=3
else:
    print("That was incorrect. You have one more chance.")
    print("[",question,"] Type the song's full name here (without
    answer= input("Type your answer here:")

if answer == randomSong:
    score+=1
else:
    print("That was incorrect. You lose.")
    print("Game Over")
    gameOver= True
```



```
Python 3.7.1 Shell
File Edit Shell Debug Options Window Help

Login Process:
What is your username?
fms11391
What is your password?
deathdeath
You have logged in!
Welcome Aarogya !

2
[ AC/DC:B I B ]
Type the song's full name here (without missing capitals for each word)

Type your answer here:Back In Black
1
[ Pantera:W ]
Type the song's full name here (without missing capitals for each word)

Type your answer here:Walk
['Metallica']
['Fade To Black']
0
[ Metallica:F T B ]
Type the song's full name here (without missing capitals for each word)

Type your answer here:Fade To Black
Traceback (most recent call last):
  File "F:\Year 11\Computer Science\1 Real NEA Task (Programming Project)\SongGa
me.py", line 229, in <module>
    theGame()
  File "F:\Year 11\Computer Science\1 Real NEA Task (Programming Project)\SongGa
me.py", line 144, in theGame
    randomNumber=random.randint(0,maximumNumber)
  File "J:\ICT Software\Python\lib\random.py", line 222, in randint
    return self.randrange(a, b+1)
  File "J:\ICT Software\Python\lib\random.py", line 200, in randrange
    raise ValueError("empty range for randrange() (%d,%d, %d)" % (istart, istop,
width))
ValueError: empty range for randrange() (0,0, 0)
>>>
```

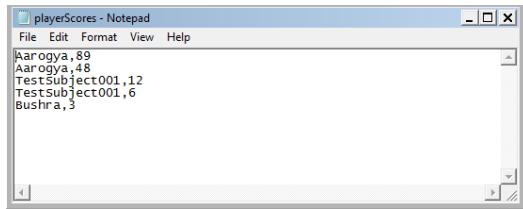
Centre: 12430

Candidate: 8483

Name: Aarogya Yadav

Score Setting algorithm:

Lastly, I had to save the score of the user onto the “playerScores.txt” file. First, right after exiting the game loop, I output the user’s score to the user. Next, I open the text file called “playerScores.txt” in read mode, and read all of its values, which are stored with the score and the player name separated by a comma, by using the same process as before.



After reading from the file, I append the user’s name and user’s new score onto the list of “scores” and “playerNames”. I set the variable “Sorted” to False so that the while loop is entered. The while loop sorts the scores in the order of highest to lowest by the use of a bubble sort. This is done by going through each item in

the list and comparing it to the next item in the list, if the next item is smaller than the item that is being looped through, the two items will switch places in the list (along with the items in the playerNames at those indexes) by setting the item that is being looped through into a temporary variable, and then setting the item being looped through to the next item and lastly setting the next index to the temporary variable. At the end of each loop through, if a single swap has taken place in the code, it means that the list has not been sorted but if no swaps take place, then the program will set the variable “Sorted” to True and so exiting the while loop.

Next, I open the file “playerScores.txt” in write mode, while creating a variable “playerScoreDetails”, assigning it an empty list to be appended to. Then I loop through the “playerNames” list and appending each index to the empty list from before. Then, I add on to each index, the scores of those users by setting each index to itself + the score from the list “scores”.

Lastly, I loop through the “playerScoreData”, writing each index onto the file along with a newline character, so that the file is overwritten and the new sorted scores can be written onto it.

Centre: 12430

Candidate: 8483

Name: Aarogya Yadav

minimize

```

print("You scored",score,"points!")
file2=open("playerScores.txt","r")
playerNames=[]
scores=[]
contents= True
while contents != "":
    contents=file2.readline()
    contents=contents.strip()
    splitContents= contents.split(",")
    if len(splitContents)==1:
        break
    playerNames.append(splitContents[0])
    scores.append(splitContents[1])
file2.close()
playerNames.append(nameOfPlayer)
scores.append(score)
Sorted= False
#The next block of code opens and reads from the file "playerScores" separating scores and allocated player names into different lists and then appending
#the new score and player name. Lastly, it initializes the variable Sorted to False to enter the next while loop.
if len(scores) <2:
    Sorted=True
while Sorted == False:
    Swap=0
    for i in range(len(scores)-1):
        if i== (len(scores)):
            break
        elif int(scores[i]) < int(scores[i+1]):
            tempScore= scores[i]
            scores[i]=scores[i+1]
            scores[i+1]=tempScore
            tempPlayerName= playerNames[i]
            playerNames[i]=playerNames[i+1]
            playerNames[i+1]=tempPlayerName
            Swap+=1
    if Swap == 0:
        Sorted= True
#The following code above checks that the list has more than 1 item to sort the list and then utilizes a bubble sort to sort the scores and player names
#from high to low.
if len(playerNames)<5:
    print("These are the top",len(playerNames),"scores!")
    for i in range(len(playerNames)):
        print (playerNames[i] +" with "+ str(scores[i])+" points!")
else:
    print("These are the top 5 scores")
    for i in range(5):
        print(playerNames[i] +" with "+ str(scores[i])+" points!")
#The block of code above outputs the top 5 scorers if there are 5 or more recorded scores, if not it will print all the scores.

File = open("playerScores.txt","w")
playerScoreDetails=[]
for x in range(len(playerNames)):
    playerScoreDetails.append(playerNames[x])
for y in range(len(playerScoreDetails)):
    playerScoreDetails[y]=playerScoreDetails[y]+ " "+ str(scores[y])
for i in range(len(playerScoreDetails)):
    File.write(playerScoreDetails[i])
    File.write("\n")

```

After creating all of the subroutines, I execute the “aAndR()” subroutine (the authentication and registration function) and then “theGame()” subroutine to end if they logged in successfully.

```

#The last block of code in the subroutine, opens the file "playerScores" in write mode to then write all the score details onto the file in the correct
#format.
aAndR()
if login==True:
    theGame()

#The last 2 lines of code run the aAndR() subroutine and then the theGame() subroutine if they have logged in successfully.

```

Song Quiz Programming Project

Centre: 12430
Candidate: 8483
Name: Aarogya Yadav

Full Layout a program run sequence of the game:

```
Login Process:
What is your username?
Tester101
What is your password?
school
You have logged in!
Welcome TestSubject !

Each question will be presented in this form:
[Artist Name: Song Initials with spaces]

[ Metallica:F W T B T ]
Type the song's full name here (without missing capitals for each word)

Type your answer here:For Whom The Bell Tolls

[ Naruto:H F ]
Type the song's full name here (without missing capitals for each word)

Type your answer here:Hokage's Funeral

[ Queen:D S M N ]
Type the song's full name here (without missing capitals for each word)

Type your answer here:Don't Stop Me Now

[ Undertale:S D ]
Type the song's full name here (without missing capitals for each word)

Type your answer here:Spider Dances
That was incorrect. You have one more chance.
[ Undertale:S D ] Type the song's full name here (without missing capitals for each word)

Type your answer here:Space Dance
That was incorrect. You lose.
Game Over
You scored 9 points!
These are the top 5 scores
Aarogya with 89 points!
Aarogya with 48 points!
TestSubject001 with 12 points!
TestSubject with 9 points!
TestSubject001 with 6 points!
>>> |
```


Song Quiz Programming Project

Centre: 12430
Candidate: 8483
Name: Aarogya Yadav

Testing

TEST NUMBER	TEST TYPE	TEST DATA	REASON	EXPECTED OUTCOME
1	Valid	user and passw (username and password)	To see if a valid username and password get through	It should bypass the authentication process and proceed into game.
2	Invalid	user and passw (username and password)	To see if an invalid username and password get through	It should exit the program immediately.

```
Login Process:
What is your username?
fhs11391
What is your password?
deathdeath
You have logged in!
Welcome Aarogya !
```

```
Login Process:
What is your username?
Invalid Username
>>> |
```

```
Login Process:
What is your username?
fhs11391
What is your password?
Invalid Password
Your password is incorrect
>>>
```

TEST NUMBER	TEST TYPE	TEST DATA	REASON	EXPECTED OUTCOME
3	Valid	loginChoice	To see if a valid response gets through	It will continue onto the login process.
4	Invalid	loginChoice	To see if an invalid response	It will ask for the input again until a valid response is given
5	Duplicate	playerName and playerUsername	To see if a duplicate input gets through	It should ask for another input.

```
Choose one of the options from below by typing its corresponding number:
1. Login
2. Register
:Invalid
Choose one of the options from below by typing its corresponding number:
1. Login
2. Register
:Invalid
Choose one of the options from below by typing its corresponding number:
1. Login
2. Register
:Invalid
Choose one of the options from below by typing its corresponding number:
1. Login
2. Register
:1
```

```
Login Process:
What is your username?
```

```
Choose one of the options from below by typing its corresponding number:
1. Login
2. Register
:1
```

```
Login Process:
```

Song Quiz Programming Project

Centre: 12430
Candidate: 8483
Name: Aarogya Yadav

TEST NUMBER	TEST TYPE	TEST DATA	REASON	EXPECTED OUTCOME
5	Duplicate	playerName and playerUsername	To see if a duplicate input gets through	It should ask for another input.

```
What is your name?  
Aarogya  
This name already exists.  
What is your name?  
Tester2  
What would you like your username to be?  
fhs11391  
This username already exists.  
What would you like your username to be?  
Tester2  
What would you like your password to be?  
|
```

TEST NUMBER	TEST TYPE	TEST DATA	REASON	EXPECTED OUTCOME
6	Valid/ Invalid	answer	To make sure the program continues on both valid and invalid input	The program continues to the next question or ends the game.

Each question will be presented in this form:
[Artist Name: Song Initials with spaces]

```
[ Queen:B R ]  
Type the song's full name here (without missing capitals for each word)  
Type your answer here:Bohemian Rhapsody
```

```
[ Dust; an Elysian Tale:C H ]  
Type the song's full name here (without missing capitals for each word)  
Type your answer here:Come Home
```

```
[ AC/DC:B I B ]  
Type the song's full name here (without missing capitals for each word)  
Type your answer here:Invalid  
That was incorrect. You have one more chance.  
[ AC/DC:B I B ] Type the song's full name here (without missing capitals for each word)  
Type your answer here:Invalid  
That was incorrect. You lose.
```

Centre: 12430
Candidate: 8483
Name: Aarogya Yadav

EVALUATION:

So after making this program, I managed to complete the task of producing a song where random songs are chosen and the user must guess the name of the song. I also managed to successfully create an authentication process that lets the user to login. I successfully managed to create the game and store data into files in the correct format, by being able to read and write from and to files. And lastly, I did manage to complete my code, avoiding any chance of errors from the user input by validating successfully as well.

In conclusion, I also identified that most of my methods could have been executed more efficiently. An example of this is my method of selecting random songs, so that no songs are repeated. Another solution could have been, to shuffle the whole list and to just loop through the list in that form and then once the loop reaches the end of the loop, and I could have shuffled the list once more and then looped through it again. There are also rooms for improvement in creating a better login and registration process, which also makes sure that the user is creating a secure username and password, this could be done by having while loops; each with their own conditions, such as the password length (using the len() function) or the inclusion of symbols and letters, along with capital and lowercase letters, all to make sure that the password chosen is valid and secure. I also realized that instead of manually sorting through the list by using a bubble sort, I could have used the pre-programmed "sort()" function in python to make my coding more efficient. There is also room for improvement in the outputting of values and questions to the user, where the layout could be further improved to be clear. The issue of the list duplicating, where assigning an empty list variable to a filled variable causes the lists to become linked in terms of editing, was resolved by appending all of the items instead which would create a duplicate by adding all the items one by one rather than equating one variable to the other.

Success Criteria and Where I met each criteria:

1. The program requires an authorisation process which allows the user to register or if they have already registered, to login by entering their details. I created a subroutine which was in charge of making sure the user's input was reasonable and matched the data on the file when logging in. Evidence is on pages 14, 15 and 24.
2. The list of player names and the points scored by those usernames are stored in a file. Evidence on page 21.

Song Quiz Programming Project

Centre: 12430

Candidate: 8483

Name: Aarogya Yadav

3. The artist names and the songs linked to the artist are stored in a file. **Evidence on page 19.**
4. The program is then required to select a random artist and song from the songs and artists file. **I utilised the random module to successfully do this by choosing a random number from 0 to the length of the list to ensure a random index is chosen and then popped from the list. Evidence on pages: 14, 18,19 and 20.**
5. The program must be able to display the name of the artist and the name of the song in the form that the description of the task states. (In the format: "Metallica: M I F" being the song by Metallica called "Moth into flame".) **Evidence on page 23.**
6. The program must be able to give the user two guesses at guessing the name of the song correctly. **Evidence on page 23.**
7. The program must accumulate the score when the user gets the song guess correct, where if it is the first guess the score accumulated is 3 and if it is the second guess then the score accumulated is 1. **Evidence on page 19 and 20.**
8. If the user gets the guesses wrong both times, the program must record the score and player name of the user and saving it into the list of scores and player names. **Evidence on page 25**
9. Then the program must exit the game. **Evidence on page 23**
10. After exiting the game, the program must display the user's score and the top 5 scorers in the file of scores and their names. **Evidence on page 23.**

Song Quiz Programming Project

Centre: 12430

Candidate: 8483

Name: Aarogya Yadav

```
import random
#The line above imports the random module, to be used to generate random numbers later on.
def loginProcess():
    global login
    login=False
    print("\n\n")
    print("Login Process:")
    file= open("UserDetails.txt","r")
    usernames=[]
    passwords=[]
    playerNames=[]
    contents= True
    while contents != "":
        contents=file.readline()
        contents=contents.strip()
        splitContents=contents.split(",")
        if len(splitContents)==1:
            break
        usernames.append(splitContents[0])
        passwords.append(splitContents[1])
        playerNames.append(splitContents[2])
    file.close()
    user= input("What is your username?\n")
    for i in usernames:
        if i ==user:
            passw=input("What is your password?\n")
            iOfPass= usernames.index(i)
            password= passwords[iOfPass]
            if passw == password:
                print("You have logged in!")
                playerName=playerNames[iOfPass]
                global nameOfPlayer
                nameOfPlayer=playerName
                print("Welcome",playerName,"! \n")
                login= True
            else:
                print("Your password is incorrect")
                quit()

#The subroutine above reads the "UserDetails" file to act as a reference for usernames and passwords to then authenticate the user's input of their username
#and password, outputting a string if the login is successful.

def registrationProcess():
    file=open("UserDetails.txt","r")
    usernames=[]
    passwords=[]
    playerNames=[]
    contents= True
    while contents != "":
        contents=file.readline()
        contents=contents.strip()
```

Centre: 12430

Candidate: 8483

[Minimize](#)

```

splitContents=contents.split(",")
if len(splitContents)==1:
    break
usernames.append(splitContents[0])
passwords.append(splitContents[1])
playerNames.append(splitContents[2])
file.close()
playerName=""
while playerName=="" or duplicate== True:
    playerName= input("What is your name?\n")
    for i in playerNames:
        if i == playerName:
            print("This name already exists.")
            duplicate= True
            break
    else:
        duplicate=False
playerUsername=""
while playerUsername =="" or duplicate== True:
    playerUsername=input("What would you like your username to be?\n")
    for i in usernames:
        if i == playerUsername:
            print("This username already exists.")
            duplicate= True
            break
    else:
        duplicate= False
#Checks for duplicate
playerPassword= input("What would you like your password to be?\n")
usernames.append(playerUsername)
passwords.append(playerPassword)
playerNames.append(playerName)
File= open("UserDetails.txt","w")
playerDetails=[]
for x in usernames:
    playerDetails.append(x)
for y in passwords:
    iOfUD= passwords.index(y)
    playerDetails[iOfUD]=playerDetails[iOfUD] +","+ y
for z in playerNames:
    iOfUD= playerNames.index(z)

```

Song Quiz Programming Project

Centre: 12430

Candidate: 8483

Name: Aranya Yadav

```
playerDetails[iOfUD]= playerDetails[iOfUD] +","+z
for i in playerDetails:
    if playerDetails.index(i)== 0:
        File.write(i)
        File.write("\n")
    else:
        File.close()
        File= open("UserDetails.txt","a")
        File.write(i)
        File.write("\n")
File.close()
loginProcess()

#The subroutine above takes in details from the user and then writes it back onto the file in the correct format with username, password and pl
#separated by commas on each line.

def aAndR():
    loginChoice=input("""Choose one of the options from below by typing its corresponding number:
1. Login
2. Register
: """)
    while loginChoice != "1" and loginChoice != "2":
        loginChoice=input("""Choose one of the options from below by typing its corresponding number:
1. Login
2. Register
: """)
    if loginChoice == "1":
        loginProcess()
    elif loginChoice == "2":
        registrationProcess()

#The subroutine above asks the user for an input of either 1 or 2, where depending on which is chosen, the allocated subroutine is run. For 1, it is the
#loginProcess() subroutine and for 2, it is the registrationProcess() subroutine.

def theGame():
    file= open("Songs.txt","r")
    artists=[]
    songs=[]
    contents=True
    while contents != "":
        contents= file.readline()
        contents=contents.strip()
        splitContents= contents.split(",")
        if len(splitContents)==1:
            break
        artists.append(splitContents[0])
        songs.append(splitContents[1])
    file.close()

#The block of code above opens and reads the file names "Songs", separating the artist names and song names and then appending them to two separate lists.
unchosenArtists=[]
unchosenSongs=[]
for i in artists:
    unchosenArtists.append(i)
for i in songs:
    unchosenSongs.append(i)
print("""\n Each question will be presented in this form:
[Artist Name: Song Initials with spaces] \n """)
gameOver=False
score=0

#This block of code duplicates the artists and songs lists to be used later. Then the code prints the form of the questions and lastly initializes the
#gameOver and score variables, where gameOver is set to False so that the while loop can be entered. The score is set to 0 to be incremented later.
while gameOver==False:
    if len(unchosenArtists) == 0:
        for i in artists:
            unchosenArtists.append(i)
        for i in songs:
            unchosenSongs.append(i)
    maximumNumber= ((len(unchosenSongs))-1)
    randomNumber=random.randint(0,maximumNumber)
    randomSong= unchosenSongs[randomNumber]
    SongFirstLetters=""
    for i in range(len(randomSong)):
        if i == 0:
            SongFirstLetters+= randomSong[i]
        elif randomSong[i] == " ":
            SongFirstLetters+= " "
            SongFirstLetters+= randomSong[i+1]
    SongArtist= unchosenArtists[randomNumber]
    question= SongArtist+ " " + SongFirstLetters
    unchosenArtists.pop(randomNumber)
    unchosenSongs.pop(randomNumber)

#The following code generates a random index and then makes the question with the randomly chosen song, removing that song from the list so that it cannot
#be chosen again.
print("\n\n")
print("[",question,""]
Type the song's full name here (without missing capitals for each word) \n """)
answer= input("Type your answer here:")
if answer == randomSong:
    score+=3
else:
    print("That was incorrect. You have one more chance.")
    print("[",question,"] Type the song's full name here (without missing capitals for each word) \n")
    answer= input("Type your answer here:")
    if answer == randomSong:
        score+=1
    else:
        print("That was incorrect. You lose.")
        print("Game Over")
        gameOver= True

#The following code prints the question while asking for the answer, which is checked if it is correct and depending on the number of attempts increments
#their score, until they get a question wrong twice, setting gameOver to true and exiting the loop.
print("You scored",score,"points!")
print("\n\n")
file2=open("playerScores.txt","r")
playerNames=[]
scores=[]
contents= True
```

Song Quiz Programming Project

Centre: 12430

Candidate: 8483

Minimize

```
while contents != "":
    contents=file2.readline()
    contents=contents.strip()
    splitContents= contents.split(",")
    if len(splitContents)==1:
        break
    playerNames.append(splitContents[0])
    scores.append(splitContents[1])
file2.close()
playerNames.append(nameOfPlayer)
scores.append(score)
Sorted= False

#The next block of code opens and reads from the file "playerScores" separating scores and allocated player names into different lists and then appending
#the new score and player name. Lastly, it initializes the variable Sorted to False to enter the next while loop.
if len(scores) <2:
    Sorted=True
while Sorted == False:
    Swap=0
    for i in range(len(scores)-1):
        if i== (len(scores)):
            break
        elif int(scores[i]) < int(scores[i+1]):
            tempScore= scores[i]
            scores[i]=scores[i+1]
            scores[i+1]=tempScore
            tempPlayerName= playerNames[i]
            playerNames[i]=playerNames[i+1]
            playerNames[i+1]=tempPlayerName
            Swap+=1
    if Swap == 0:
        Sorted= True

#The following code above checks that the list has more than 1 item to sort the list and then utilizes a bubble sort to sort the scores and player names
#from high to low.
if len(playerNames)<5:
    print("These are the top",len(playerNames),"scores!")
    for i in range(len(playerNames)):
        print (playerNames[i] +" with "+ str(scores[i])+ " points!")
else:
    print("These are the top 5 scores")
    for i in range(5):
        print(playerNames[i] +" with "+ str(scores[i])+ " points!")

#The block of code above outputs the top 5 scorers if there are 5 or more recorded scores, if not it will print all the scores.

File = open("playerScores.txt", "w")
playerScoreDetails=[]
for x in range(len(playerNames)):
    playerScoreDetails.append(playerNames[x])
for y in range(len(playerScoreDetails)):
    playerScoreDetails[y]=playerScoreDetails[y]+ ", "+ str(scores[y])
for i in range(len(playerScoreDetails)):
    File.write(playerScoreDetails[i])
    File.write("\n")
quit()

#The last block of code in the subroutine, opens the file "playerScores" in write mode to then write all the score details onto the file in the correct
#format.
aAndR()
if login==True:
    theGame()

#The last 2 lines of code run the aAndR() subroutine and then the theGame() subroutine.
```