Task completed: 2

| Date started | Date completed |
| --- | --- |
| 15/02/2021 |  |

Analysis

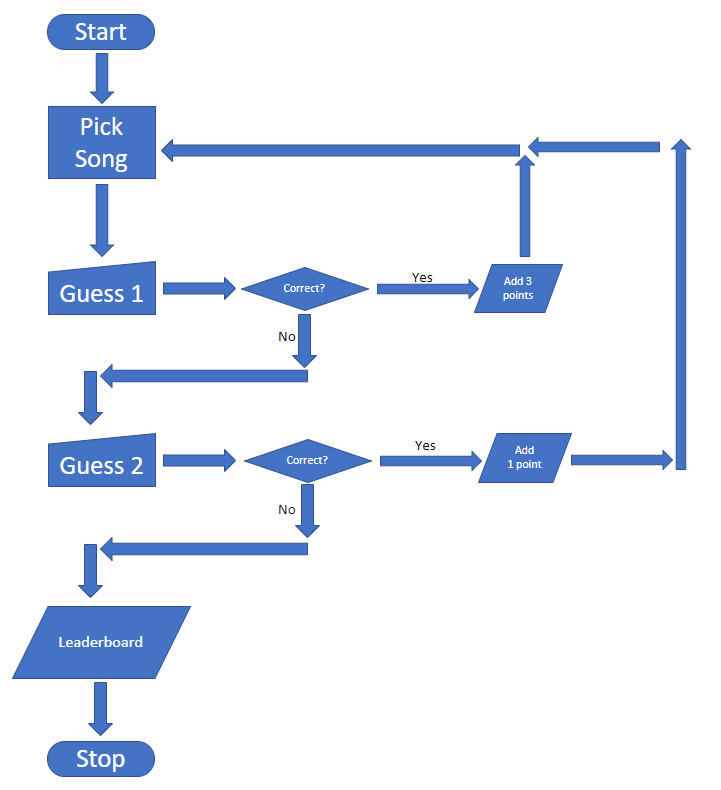
Try and create 3 or more key success criteria for your program.

**Success Criteria:**

1. Efficient and tidy code
2. Easy to edit later
3. Robust and hard to break

Design

* *You may like to create a flow charts which will show broadly how your program will work. If so include your flow chart in this section.*
* *You must create pseudocode for a part of your program (minimum of 15 lines). If possible, try to create all of your program in pseudocode. Use the OCR guide in the specification to help you.*



create a file called “AuthenticatedPlayers”

write player data in file

close file

create a file called “Leaderboard”

write leaderboard data in file

close file

create a file called “Artists”

write the artists that are associated with the songs that the player will be guessing

close file

create a file called “SongNames”

write the names of the songs that the players will be guessing in here

close file

#make sure the song names and the artists are on the same lines as each other

ask for player input while saying “Please enter your name”

check if their name is in the “AuthenticatedPlayers” file

if their name is in the file then continue the program

if their name is not in the file then tell them they are not authenticated

create a variable called “Score”

open a while true loop

generate a random number between 1 and 10

call that number “SongChosen”

open the “Artists” file

go to the “SongChosen” line

print the artist on that line

open the “SongNames” file

go to the “SongChosen” file

print the first letter of each word of the song, while showing how many letters there are in the song

allow player to guess by making an input called guess1

if they guess correctly then add 3 points to their “Score”

if they get it wrong give the player another chance by creating an input called guess2

if they guess correctly then add 1 point to their score

if wrong again then break loop otherwise keep looping

if the players “Score” is higher than a leaderboard score then replace the value on the leaderboard

prints a message to the player telling them their score and if its a new highscore

print the top 5 players on the leaderboard

Test design

* *Think of tests that you can carry out to see if your system works*
* *Remember to try and use normal, boundary and erroneous tests.*
* *If you wish to, you may add more tests to the table.*

**My tests:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test** | **What am I testing?** | **What data will I use?** | **Normal/Boundary/Erroneous?** | **Expected Result** |
| **1** | Authenticated Players | AuthenticatedPlayers file | Normal | If player in list continue, else display message |
| **2** | Random Song | SongChosen | Normal | When inputted correct song, program loops |
| **3** | Player Score | Score | Normal | Total score at the end of the game |
| **4** | Durability | All inputs | Boundary | If player enters something silly, program will not crash |
| **5** | Artists and Song match | Artist and song files | Normal | Song is matched with correct artist |

Development

* *Copy and paste your code into this section*
* *Remember to try and add comments to your code to make it more readable!*

**My program code:**

#// Imports //#

import random

import linecache

import time

import tkinter

import tkinter.font

#// Variables //#

points = 0

roundCounter = 0

artistsTxt = "Files\\Artists.txt"

authenticatedPlayersTxt = "Files\\AuthenticatedPlayers.txt"

doNotInputTxt = "Files\\DoNotInput.txt"

leaderboardTxt = "Files\\Leaderboard.txt"

lowercaseSongNamesTxt = "Files\\LowercaseSongNames.txt"

songNamesTxt = "Files\\SongNames.txt"

songNamesBlanksTxt = "Files\\SongNamesBlanks.txt"

licenseTxt = "Files\\Other Files\\LICENSE.txt"

changelogTxt = "Files\\Other Files\\Changelog.txt"

controlsTxt = "Files\\Other Files\\Controls.txt"

#// Functions //#

def playerAuthorisation():

#// Player Authorisation //#

global playerName

while True:

playerName = str(input("Please enter your name: "))

playerName = playerName.upper()

with open(doNotInputTxt, "r") as doNotInput:

if playerName in doNotInput.read():

print("Invalid input, try again.")

doNotInput.close()

else:

with open(authenticatedPlayersTxt, "r") as authenticatedPlayers:

if playerName in authenticatedPlayers.read():

print("Access Granted.\n")

break

else:

print("Access Denied.")

print("Closing...")

time.sleep(3)

quit()

def newSong():

#// New Song //#

global lineChosen

lineChosen = random.randint(3,12)

global songChosen

songChosen = linecache.getline(songNamesTxt, lineChosen)

global previousSong

previousSong = songChosen

def displayArtistAndSongBlanks():

#// Display Artist And Song //#

newSong()

print("\nMake sure to use grammar.")

print("\nThe artist of the song is:", (linecache.getline(artistsTxt,lineChosen)))

print(linecache.getline(songNamesBlanksTxt, lineChosen))

def openProgram():

#// Open Program //#

global inputControl

while True:

print("========== Main Menu ==========")

print("Play Game - Play the game.")

print("Controls - View controls.")

print("Leaderboard - View leaderboard.")

print("Changelog - View changelog.")

print("License - View license.")

print("Quit - Quit the program.\n")

inputControl = str(input("What would you like to do?\nInput: "))

inputControl = inputControl.upper()

if inputControl == "PLAY GAME":

playGame()

else:

if inputControl == "CONTROLS":

with open(controlsTxt, "r") as controls:

controlsData = controls.read()

print("\n", controlsData, "\n")

controls.close()

else:

if inputControl == "LEADERBOARD":

with open(leaderboardTxt, "r") as leaderboard:

leaderboardData = leaderboard.read()

print("\n", leaderboardData, "\n")

leaderboard.close()

else:

if inputControl == "CHANGELOG":

with open(changelogTxt, "r") as changelog:

changelogData = changelog.read()

print("\n", changelogData, "\n")

changelog.close()

else:

if inputControl == "LICENSE":

with open(licenseTxt, "r") as License:

licenseData = License.read()

print("\n", licenseData, "\n")

License.close()

else:

if inputControl == "QUIT":

print("Closing program...")

time.sleep(3)

quit()

else:

if inputControl == "":

print("No input detected, try again.")

else:

if inputControl == " ":

print("No input detected, try again.")

else:

print("That's not an option.\n")

def endGame():

global points

global roundCounter

print("Game Over...")

with open(lowercaseSongNamesTxt, "r") as lowercaseSongNames:

lowercaseSongChosen = linecache.getline(lowercaseSongNamesTxt, lineChosen)

lowercaseSongNames.close()

print("The correct answer was:", lowercaseSongChosen)

if roundCounter == 1:

print("You played 1 round.")

else:

print("You played ", roundCounter, "rounds.")

if points == 1:

print("You got 1 point.")

else:

print("You got",points,"points.\n")

addToLeaderboard()

points = 0

roundCounter = 0

while True:

print("Play again?")

print("Yes / No")

optionInput = str(input("Input: "))

optionInput = optionInput.upper()

if optionInput == "YES":

playGame()

else:

if optionInput == "NO":

print("")

openProgram()

else:

print("That's not an option.")

def playGame():

#// Play Game //#

global guess1

global guess2

global points

global roundCounter

global songChosen

global songNamesTxt

while True:

newSong()

displayArtistAndSongBlanks()

guess1 = str(input("Guess the song: "))

guess1 = guess1.upper()

controlsGuess1()

noInputGuess1()

invalidInputGuess1()

with open(songNamesTxt) as songNames:

if guess1 in songChosen:

points += 3

roundCounter +=1

print("Correct!")

songNames.close()

else:

guess2 = str(input("Incorrect! Try again: "))

guess2 = guess2.upper()

controlsGuess2()

noInputGuess2()

invalidInputGuess2()

if guess2 in songChosen:

points += 1

roundCounter += 1

print("Correct!")

else:

songNames.close()

endGame()

def controlsGuess1():

#// Controls Guess 1 //#

global guess1

if guess1 == "STOP":

endGame()

if guess1 == "SKIP":

print("Skipping song...")

playGame()

def controlsGuess2():

#// Controls Guess 2 //#

global guess2

if guess2 == "STOP":

endGame()

if guess2 == "SKIP":

print("Skipping song...")

playGame()

def noInputGuess1():

#// No Input Guess 1 //#

global guess1

global noInputMessage

noInputMessage = "No input detected, try again."

if guess1 == "":

print(noInputMessage)

else:

if guess1 == " ":

print(noInputMessage)

def noInputGuess2():

#// No Input Guess 2 //#

global guess2

global noInputMessage

if guess2 == "":

print(noInputMessage)

else:

if guess2 == " ":

print(noInputMessage)

def invalidInputGuess1():

#// Invalid Input Guess 1 //#

global guess1

global invalidInputMessage

invalidInputMessage = "Invalid input, try again."

with open(doNotInputTxt, "r") as doNotInput:

if guess1 in doNotInput:

print(invalidInputMessage)

doNotInput.close()

def invalidInputGuess2():

#// Invalid Input Guess 2 //#

global guess2

global invalidInputMessage

with open(doNotInputTxt, "r") as doNotInput:

if guess2 in doNotInput:

print(invalidInputMessage)

doNotInput.close()

def addToLeaderboard():

#// Update Leaderboard //#

global points

global playerName

with open(leaderboardTxt, "a") as leaderboard:

points = str(points)

leaderboard.write("\n\nName: ")

leaderboard.write(playerName.capitalize())

leaderboard.write("\n")

leaderboard.write("Points: ")

leaderboard.write(points)

leaderboard.close()

with open(leaderboardTxt, "r") as leaderboard:

leaderboardData = leaderboard.read()

print(leaderboardData, "\n")

leaderboard.close()

#// Function Calling //

playerAuthorisation()

openProgram()

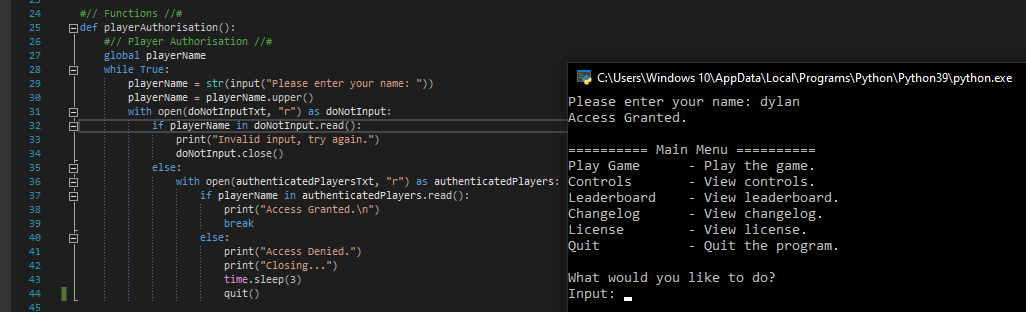
Testing

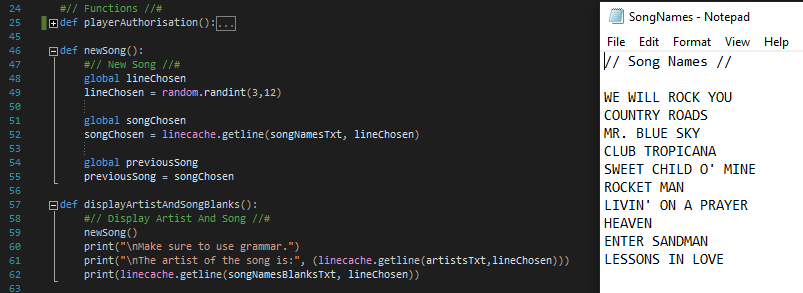
* *Show you have completed the tests you thought of*
* *Identify if you needed to make changes to your program*
* *Include the screenshots of the tests*

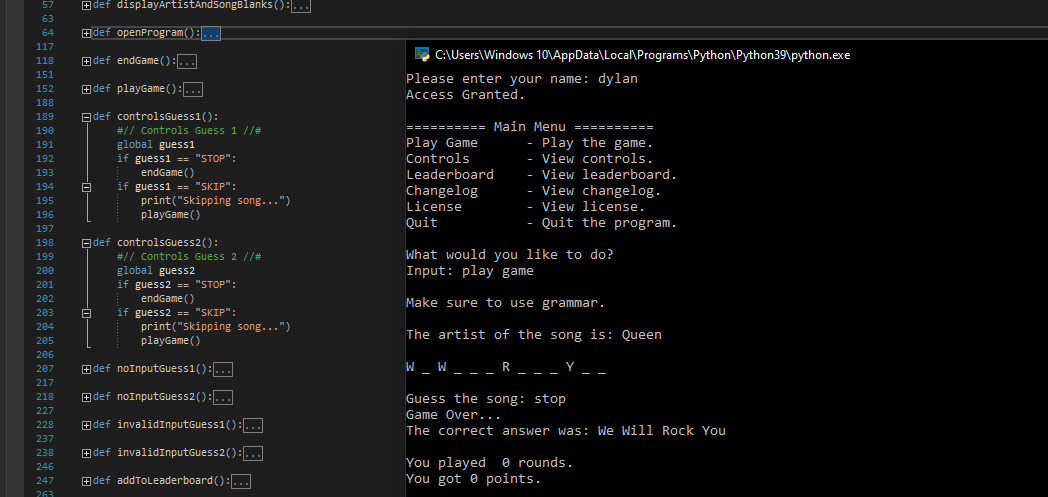
**My tests:**

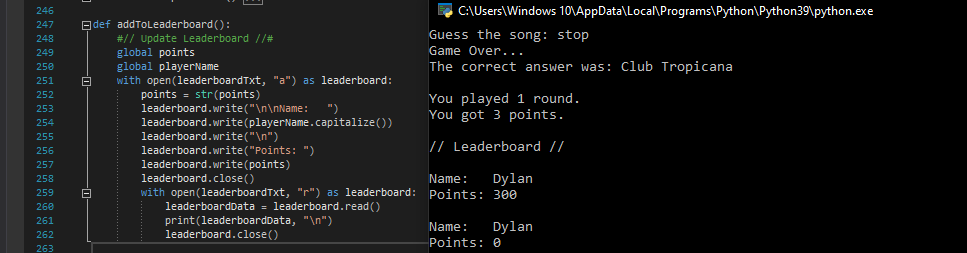
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test** | **What am I testing?** | **Expected result** | **Pass/Fail** | **Do I need to change my program? If so, how?** |
| **1** | Authenticating Players | Game will store player name | Pass | No changes needed. |
| **2** | Storing new song in variable | Game stores song in variable | Pass | No changes needed. |
| **3** | Controls | Game stops when player inputs “STOP” | Pass | No changes needed. |
| **4** | Controls | Game gets new song when player inputs “SKIP” | Pass | No changes needed. |
| **5** | Leaderboard | Game prints leaderboard when the game ends | Pass | No changes needed. |

**My test screenshots:**









Evaluation

* Evaluate how successful your program was. You should like your evaluation to your testing results.
* You should reflect on any new skills you have developed

This section should be approximately 200-500 words.

**How successful was my program?**

My program overall was successful as it came out better than I had expected. At first I was still rusty as I hadn’t done much programming for a long while but after an hour or so it was just as easy as I had remembered. The only significant problems I had was when the computer would pick a song, print out the song (for testing purposes) and the computer would actually have chosen the song after the one it had originally picked.

**What new skills have I developed?**

I have developed a wide variety of new skills such as:

- File Handling

Handling files in Python when starting was a bit strange with reading and getting the program to look for a specific thing in a file. But after a bit of research it was fairly easy.

- Variable Name Consistency

When I was programming I always had a thing of keeping my code as neat as I could and making it as easy to read as possible. One of the major things to help with this would be keeping my variable names consistent throughout my program.

- Organising Code

Like I mentioned above, I always have a thing with keeping my code as tidy as possible and easy to read and edit. I had learned from making this program that I had to make sure it was easy to read and efficient, I knew that there would be a lot of code involved which is why I had to try to make it as tidy as possible.