```
# Name: Jorge Barrueta, Alejandro Lopez
# Date: 12/02/19
# Course: COSC 2316 Fall 2019 (Dr. Shebaro)
# Program Description: Spanish learning program

######### Algorithm/Psuedocode ########

##<<<<<<<< Updated upstream
# 1. define a Student Class, with the needed methods (Ex: average)
# 2. write a toString of the class to write into the text file
# 2.1. Create function that reads a file and returns list of lines
# 3. when the program begins, read the text files that exist (users.txt, learn.txt)
# 4. for each user in user.txt create a new instance of a Student. Save each into a users list.
# 5. save each translation into a dictionary - learnDict MUST BE GLOBAL
# 6. start loop for menu until exit is selected
# 7. begin the menu, which includes Learn, Test, Leaderboard
# 8. if Learn is pressed, step though the learn dictionary and present the key. Once the user presses
#         enter show the value, the next enter will go to the next key.
# 9. if Test is pressed, the user will be asked to either create a new Student instance or sign into a previous one
# 10. a function will randomly select 5 keys from the dictionary, the user will be displayed the key and must enter
#     the equivalent to the value.
# 11. if correct, the instance correct will go up as well as total, if wrong, wrong will go up one as well as total
# 12. Leaderboard will sort the list of users based on total correct, and display the top five.
# 13. Exit ends program, save the changes in users to the text file. (format: average correct wrong total name\n)
#         (ex: from the list users -> users[0].__str__() will write it in the correct format)


############## Python Code #############
import random
import itertools


class Student:
    def __init__(self, name, correct=0, wrong=0, total=0):
        self.name = name
        self.correct = correct
        self.wrong = wrong
        self.total = total
        self.average = 0 if self.total <= 0 else (self.correct / self.total) * 100
```

```
    def __str__(self):
        return str(round(self.average, 2)) + " " + str(self.correct) + " " + str(self.wrong) + " "
+ str(
            self.total) + " " \
              + self.name

    def getCorrect(self):
        return int(self.correct)

    def tCorrect(self):
        self.correct += 1
        self.total += 1

    def tWrong(self):
        self.wrong += 1
        self.total += 1

    def tAverage(self):
        self.average = 0 if self.total <= 0 else (self.correct / self.total) * 100

    def fomatToString(self):
        self.tAverage()
        print("Name:", self.name, "ll Average:", self.average, "ll Correct:", self.correct, "ll
Wrong:", self.wrong,
              "ll Total:", self.total)


# Function Description: readFile function will read a file and save it into a list, if text file
doesn't exits,
#                       it will create it
# Precondition: readFile will receive a text file
# Postcondition: will return a list of file lines
def readFile(txtFile):
    txtList = []
    try:
        with open(txtFile, "r") as inFile:
            txtList = inFile.readlines()
    except IOError:
        createdFile = open(txtFile, "w")
        createdFile.close()
    return txtList


# Function Description: extract users from the text file users.txt
# Precondition: must receive a list of users from the text file, a user by line
# Postcondition: will return a list of Student instances
def extractUsers(list1):
```

```
    tempUsers = []
    count = 0
    for user in list1:
        splitUser = user.split()
        try:
            avg = float(splitUser[0])
            correct = int(splitUser[1])
            wrong = int(splitUser[2])
            total = int(splitUser[3])
            name = " ".join(splitUser[4:])
            tempUsers.append(Student(name.strip(), correct, wrong, total))
        except ValueError:
            count += 1
    if count > 0:
        print("WARNING:", count, "could not be read and were skipped, please fix text file
to be in the correct format")
    return tempUsers


# Function Description: testOption will redirect student to either create a new user or
open old user, then call testing
# Precondition: will receive nothing
# Postcondition: will return nothing, will make changes to student's progress
def testOption():
    usrInput = 0
    n = 0
    while usrInput != 5:
        try:
            usrInput = int(input("\n\n1. Create new user
2. Open old user
3. Delete old user
4. List out the users
5. Return to main menu
>>> "))
            if usrInput == 1:
                usrName = input("\n\nPlease enter your name: ")
                users.append((Student(usrName.strip())))
                n = -1
                testing(n)
            elif usrInput == 2:
                usrName = input("\n\nPlease enter your name (enter exactly as you did
before): ")
                found = False
                for index in range(len(users)):
                    sName = users[index].name
                    if str(sName).strip().lower() == usrName.strip().lower():
                        found = True
```

```
                    n = index
                    break
            if not found:
                print("Could not find your user, it is now being created")
                users.append(Student(usrName.strip()))
                n = -1
            testing(n)
        elif usrInput == 3:
            delUser = input("Please enter the name of the user you would like to delete: ")
            for index in range(len(users)):
                sName = users[index].name
                found = False
                if str(sName).strip().lower() == delUser.strip().lower():
                    users.pop(index)
                    found = True
                    print("Successfully deleted the user")
                    break
            if not found:
                print("Could not find the user you wish to delete")
        elif usrInput == 4:
            print("\n" * 20)
            for user in users:
                user.fomatToString()
        elif usrInput < 1 or usrInput > 5:
            print("Please enter a valid option")
    except ValueError:
        print("Please enter a number")


# Function Description: testing will carry out the actual test and make the changes to
Student
# Precondition: will receive the index of the current student testing
# Postcondition: will return nothing
def testing(i):
    input("\n\nHello " + users[i].name + "! \nPress enter to start the exam...")
    print("\n" * 20)
    countC = 0
    countW = 0
    randomKeys = random.sample(list(learnDict), 5)
    for key in randomKeys:
        usrAns = input("Write the Spanish translation.\n" + key + ": ")
        if usrAns == learnDict[key]:
            users[i].tCorrect()
            users[i].tAverage()
            print("Good job!!!!")
            countC += 1
        else:
```

```
            users[i].tWrong()
            users[i].tAverage()
            print("Wrong! Spanish word was...", learnDict[key])
            countW += 1
    print("\n" * 20)
    saveUsers()
    print(str(users[i].name).upper() + "'s RESULTS:\n\nTotal Correct:", countC, "\nTotal
Wrong:", countW, "\nAverage:",
          round((countC / 5) * 100, 2))


# def learn(i):
# Function Description: writes user information to file
# Precondition: will receive the student information
# Postcondition: will write the student information in users.txt
def saveUsers():
    with open("users.txt", "w") as uFile:
        for i in users:
            uFile.write(str(i) + "\n")


# Function Description: creates dictionary for test
# Precondition: receive nothing
# Postcondition: will return dictionary of questions
def testWords():
    with open("learn.txt", "r") as tWordsFile:
        listEng = []
        listSpan = []
        # var [list of striped words]
        tempList = [elem.strip() for elem in tWordsFile.readlines()]
        for i in range(0, len(tempList)):
            if i == 0 or i % 2 == 0:
                # english list
                listEng.append(tempList[i])
            else:
                # spanish list
                listSpan.append(tempList[i])
        eng2Span = dict(zip(listEng, listSpan))
        return eng2Span


# Function Description: learn will give the user a flash Card-esqe testing
# Precondition: receives nothing
# Postcondition: will display one card of information to learn
def learn():
    learnMat = testWords()
    front = list(learnMat.keys())
```

```
    back = list(learnMat.values())
    for i in range(len(front)):
        if i != len(front):
            print("\n*****" + front[i] + "*****")
            learnInput = input("\n\nHit enter to see Translation:")
            print("\n#####" + back[i] + "#####")
            learnInput = input("\n\nHit enter to get next word ('x' to end):")
            if learnInput == 'x':
                break


# Function Description: Display the Top 5 leaderboard sorted by Total Correct
# Precondition: receives nothing
# Postcondition: Displays Top 5 leaderBoard
def leaderBoard():
    sortlist = [("ll Total Correct: ", student.correct, " ll Student: " + str(student.name)) for
student in users]
    lBoard = sorted(sortlist, reverse=True)
    lBoard = [(i[0],str(i[1]),i[2]) for i in lBoard]
    for person in itertools.islice(lBoard, 5):
        print("".join(person))


# leaderBoard()
# Function Description: Returns a number to be used in menu
# Precondition: User enters a positive integer
# Postcondition: returns a number
def menuInput():
    print("\n\n\nLanguage Learning Program"
          "\n1 LeaderBoard"
          "\n2-Test"
          "\n3-Learn"
          "\n4-exit")
    try:
        menuNum = int(input("Enter a Number  "))
        return menuNum
    except ValueError:
        print("please enter a valid input")


# Function Description: Gives the Function Called
# Precondition: User enters a positive integer
# Postcondition: Function is called or closes menu
def menu():
    loop = 0
    menuOp = menuInput()
    while loop == 0:
```

```
        if menuOp == 1:
            loop = 1
            print("Top 5 Leaderboard :")
            leaderBoard()
            input("\n\n\npress enter to return to menu")
            menu()
        elif menuOp == 2:
            loop = 2
            print("Testing Profile")
            testOption()
            input("\n\n\npress enter to return to menu")
            menu()
        elif menuOp == 3:
            loop = 3
            input("\npress enter to begin")
            print("\n\n\nStarting....")
            learn()
            input("\nAll Cards finished \nHit enter to go back to menu")
            menu()
        elif menuOp == 4:
            print("Thank you for learning with us Today")
            break
            loop = 4
        else:
            loop = 1
            input("\n\n\npress enter to return to menu")
            menu()


# Driver Program
usersFile = readFile("users.txt")
users = extractUsers(usersFile)
learnDict = testWords()
menu()
```