ITSE 1302

Travis Doporto

Final Exam

Again you can ignore the commented section. Those were basically notes for myself and code that I was not ready to delete.

This program was, like you said very similar to the review thankfully, mostly I held them side by side and edited a few of the words and had to resort to google to help define some of the operators used such as the entries and the list next to them. I did not use my own time and date standard I just copied yours and kept running it making sure it mirrored your example. At first the code was fairly simple and easy to reproduce but once the advanced portion came into play that’s where it got a little tricky. You almost had to do the complete version in order to achieve a desirable look which again required lots of back tracking and googling what my errors meant. I did utilize a few friends in the web design world who have backgrounds in coding but to my uncanny luck none of them knew Python so I had to google their logic in Pythons terms which is why a lot of seems like I am using different verbiage throughout. I will say that I thought “INDENTING” was Pythons Achilles heel but maybe it is “EXCEPT” operators. That is what held me back during a lot of this as I was able to do the basic score no problem but once I got to the full is where those really slowed me down. Overall this was fun. I do not know how I will apply this to my personal life but that is up to me to find an application.

Output Screenshot:

A computer screen shot of a program

Description automatically generatedA computer screen shot of a program

Description automatically generated

Source Code:

from datetime import datetime

import csv

def get\_current\_time():

    return datetime.now().strftime('%m/%d/%y %I:%M:%S %p')

def read\_file(unitExam):

    try:

        with open('exam\_3\_test\_file.txt', 'r') as f:

            for line in f:

                newLine = line.strip().split('|')

                unitExam.append(newLine)

    except FileNotFoundError:

        print("The file 'unit\_3\_exam.txt' was not found.")

    except Exception as e:

        print(f"An error occurred: {e}")

def budgetContent(unitExam):

    print(f"{'Month':<30}{'Amount':<20}")

    print("=" \* 30)

    for entry in unitExam:

        if len(entry) >= 2:

            month = entry[0]

            amount = entry[1]

            print(f"{month:<30}{amount:<20}")

def write\_csv(unitExam):

    fileName = input("Enter your filename: ")

    with open(f"{fileName}.csv", "w", newline="") as f:

        writer = csv.writer(f)

        writer.writerows(unitExam)

    # try:

        # with open(fileName, 'w', '') as csvFile:

        #     csvWriter = csv.writer(csvFile)

        #     csvWriter.writerow(['Month', 'Amount'])

        #     for entry in unitExam:

        #         if len(entry) >= 2:

        #             csvWriter.writerow([entry[0], entry[1]])

        print(f"Congrats! Filename: {fileName} created!")

    # except Exception as e:

    #     print(f"An error has occured: {e}")

def editDirectory(unitExam):

    while True:

        try:

            for index, entry in enumerate(unitExam, start=1):

                print(f"{index}. {entry[0]} | {entry[1]}")

            userSelection = int(input("Enter the entry you want to edit (or -1 to exit): "))

            if userSelection == -1:

                break

            elif userSelection < 1 or userSelection > len(unitExam):

                print("Error: Invalid selection. Please choose a number between 1 and", len(unitExam))

            selected\_entry = unitExam[userSelection - 1]

            print(f"\nYou selected: {selected\_entry[0]} | {selected\_entry[1]}")

            new\_month = input(f"Enter a new Month (current: {selected\_entry[0]}): ").strip()

            new\_amount = input(f"Enter a new Amount (current: {selected\_entry[1]}): ").strip()

            if new\_month:

                selected\_entry[0] = new\_month

            if new\_amount:

                selected\_entry[1] = new\_amount

            print(f"Updated entry: {selected\_entry[0]} | {selected\_entry[1]}")

            return unitExam

        except ValueError:

            print("Error: Invalid input. Please enter a valid number for selection.")

        except Exception as e:

            print(f"An unexpected error occurred: {e}")

def deleteDirectory(unitExam):

    while True:

        try:

            for index, entry in enumerate(unitExam, start=1):

                print(f"{index}. {entry[0]} | {entry[1]}")

            selection = int(input("Enter the number you want to delete (1-3 or -1 for no change): "))

            if selection == -1:

                print("Exiting with no changes")

                return

            elif selection < 1 or selection > len(unitExam):

                print("Error, please enter a valid option.")

                return

            selectedEntry = unitExam[selection - 1]

            del unitExam[selection - 1]

            print(f"\nRemoved entry: {selectedEntry[0]} | {selectedEntry[1]}")

            return unitExam

        except ValueError:

            print("Error: Invalid input. Please enter a valid number for selection.")

        except Exception as e:

            print(f"An unexpected error occurred: {e}")

def listData(unitExam):

    try:

        for index, entry in enumerate(unitExam, start=1):

            print(f"{index}. {entry[0]} | {entry[1]}")

    except Exception as e:

        print(f"An unexpected error occurred: {e}")

def main():

    print("Budget Editing Program")

    print("==="\*15)

    print(f"Starting time: {get\_current\_time()}")

    unitExam = []

    read\_file(unitExam)

    budgetContent(unitExam)

    write\_csv(unitExam)

    listData(unitExam)

    while True:

        try:

            print("/nWhat operation would you like to perform?")

            print("1. Edit a value")

            print("2. Delete a value")

            print("3. Write data to csv and close program")

            print("-1. Close program without operating")

            userChoice = int(input("Enter your operation choice: "))

            if userChoice == 1:

                unitExam = editDirectory(unitExam)

            elif userChoice == 2:

                unitExam = deleteDirectory(unitExam)

            elif userChoice == 3:

                unitExam = write\_csv(unitExam)

                break

            elif userChoice == -1:

                print("Exiting without operating")

                break

            else:

                raise ValueError("Please select a valid option!")

            listData(unitExam)

        except ValueError as e:

                print(f"Error: {e}. Please enter a valid number.")

        except Exception as e:

                print(f"An unexpected error occurred: {e}")

        print(f"Ending time: {get\_current\_time()}")

if \_\_name\_\_ == "\_\_main\_\_":

    main()