INFORMATICS INSTITUTE OF TECHNOLOGY

Foundation Certificate in Higher Education

Course:	Foundation Certificate Program
Module:	DOC334 (Computer Programming)
Module Leader:	Mr. Nishan Saliya Harankahawa
Type of Assignment:	Individual Coursework
Submission Date:	01/ December/2024
Student Name:	W.D.N.Nimsara
Student ID:	20240281

1. Abstract

The objective is to build MathsBro, a Python-based console game designed to enhance mathematical skills through gamified learning. The program offers four difficulty levels Demo, Easy, Medium, and Hard and provides real-time feedback, session summaries, and data storage in text and HTML formats. The project applies advanced programming techniques, including modular design, error handling, and dynamic file management.

2. Acknowledgement

I am profoundly grateful to the IIT management for organizing the foundation program and creating a helpful environment for us to improve our academic and general skills under the direction of our reputable lecturers.

First and foremost, I would like to express my gratitude to Mr. Nishan Saliya, Module leader, for providing us with guidance in building a simple game and developing it step by step. I would also like to extend my appreciation to Ms. Bihini Amarasinghe, lecturer of the module "DOC334: Computer Programming", for her constructive feedback and encouragement throughout this project semester. Her guidance and input have been crucial in shaping the direction and improving the quality of our work. Finally, I want to express my special thanks to my batchmates and friends, for providing technical advice to me and sharing the knowledge they have. Which have been fundamental in helping me to reach this milestone.

3. Table of contents

4. List of Figures

5. List of Tables

6. Introduction

6.1 Brief explanation

MathsBro is a Python-based console game designed for single-player, which focuses on improving math skills through an entertaining process of playing. When playing users solve randomly generated arithmetic questions in different difficulty modes(Demo, Easy, Medium, and Hard) with questions ranging from basic addition to multiplication. And the game dynamically changes question complexity according to the selected mode. And it follows several features, quick feedback, gameplay tracking and multiple rounds per session.

6.2 Tasks to complete

1. Develop a Python 3.x based console game using appropriate constructs, including packages, modules, functions, variables, and data structures.
2. Enable single human player to choose game mode(demo mode, easy mode, medium mode and hard mode) and play multiple game plays per each session(Mode). And each session consists of several arithmetic questions according to the game mode.
3. After the player completes all questions, return feedback and store records of correctness or incorrectness with correct answers.
4. Store and Display correctly answered question count and percentage with total question count and selected mode.
5. Each session can be played multiple rounds, if the player prefers.
6. Save game history in text files with .TXT extension.
7. Save game history in HTML file with .HTML extension.

7. problem statement

The objective is to create a console-based Python program called MathsBro, a single-player math game. The game allows four difficulty modes (Demo, Easy, Medium, and Hard) each provide a range of random math questions. This program should allows the players to play multiple rounds, identify correctness or incorrectness with percentages for each round. Furthermore, this program's history should save to both TXT and HTML files.

8. Developed Salutation

1. Game initialization: Set up a prompt for player to enter mathbro.py and game mode on his command prompt otherwise game start as Demo mode.
2. Game mode: Create different modules for each mode such as demo.py, easy.py, medium.py and hard.py.
3. Game play: Create an algorithm to generate questions according to the game mode player selects and player must answer the given prompt.
4. Game result: Once the user answered, instantly got a comment about his answer-correct or incorrect. After the player has answered every question, the program shows the result sheet with the correct answers.
5. Game summary: The program then shows the session summary in total questions, correct answers, percentages, and game mode after the result sheet.
6. Game loop: In this case, after the game round, the player receives a question like "Do you want to try again?", and if the player wants to play more, his answer should be 'yes', then the game loops.
7. Game history: History will be automatically recorded in a text file and an HTML file after each round by the player.
8. Error handling: Add an error handling mechanism to detect any kind of issues with the user's input and give error messages to the user and ignore that error.

9. Algorithm

1. Start
2. Type mathbro.py and game mode.
3. Click enter.
4. Carefully read the question.
5. Type the answer.
6. Click enter.
7. After answering all questions, read the result sheet.
8. Read summary sheet.
9. Read the question given below.
10. If you want to try again type ‘yes’ and enter. it will go to step 4.
11. If you want to stop type ‘no’.
12. End

10. Source code

10.1 mathbro.py – Main program

```
#import modules
import random
from datetime import datetime
import demo
import easy
import medium
import hard
import sys

#creat variables/File Object
FileObject=None
filename=None
HtmlObject=None
htmlname=None
mode=None
session=0
intell=None
GameData=None
OtherData=None
now=None
nowtime=None
nowdate=None
text0=None
text1=None
Qnumber=None
question=None
answer=None
correct=None
accuracy=None
text2=None
tryagain=None
argulength=None
SavedText=None
details=None

#get current time and date
now=datetime.now()
nowtime=now.strftime("%H:%M") #convert to 24-hour format
nowdate=datetime.now().date()
```

```
#creating filename
filename = now.strftime(f"{now.date()}_{now.time().strftime('%H%M')}_{random.randint(100,
999)}.txt")
htmlname = now.strftime(f"{now.date()}_{now.time().strftime('%H%M')}_{random.randint(100,
999)}.html")

#open file object
FileObject=open(filename,"w",encoding="utf-8")

#store Time and date details
text0=f"Date : {nowdate} \nTime : {nowtime}\n"

FileObject.write(text0)
#--------------------------------------------------Main program-------------------------------------------------
-
#getting commands from CMD
print("\nWellcome to Mathbro...")
if len(sys.argv)==2:
   mode=str(sys.argv[1])
else:
   mode=str(sys.argv[0])

#creat loop for multiple rounds
while True:
   #Creat directions
   if mode=="-e":
      intell=easy.easygame()
   elif mode=="-m":
      intell=medium.mediumgame()
   elif mode=="-h":
      intell=hard.hardgame()
   else:
      mode="demo"
      intell=demo.demogame()

   #seperating return arguments
   GameData=intell[0]
   OtherData=intell[1]

   #session count
   session+=1

   #-----------------------File handling PART------------------------------------------------------------------
   text1=f"\nSession {session}\nResult sheet\n"
   FileObject.write(text1)

   #store Game data to file
```

```
    for x in range(len(GameData)):
        Qnumber=x+1
        question=GameData[x]['Question']
        answer=GameData[x]['user']
        correct=GameData[x]['com']
        accuracy=GameData[x]['sta']

        if answer==correct:
            text2= f"✓ {question} = {answer}"
        else:
            text2= f"✕ {question} = {answer} correct answer is {correct}"

        FileObject.write(text2+"\n")

    #store overoll data
    TotalQuestions=OtherData[0]['total']
    CorrectQuestions=OtherData[0]['corr']
    Marks=OtherData[0]['pre']
    Level=OtherData[0]['mod']

    text3=f"\nTotal questions : {TotalQuestions}\nCorrect questions : {CorrectQuestions}\nMarks : {Marks}\nLevel : {Level}"

    FileObject.write(text3+"\n")

    tryagain=input("\nDo you want to Try Again (Yes/No)?").lower()

    # Chance to escape from loop
    if (tryagain == str("no")):
        print("\nThank you for playing mathbro...")
        FileObject.close()
        break

    elif(tryagain == str("yes")):
        continue

    else:
        # detecting value error and ignore
        print("\nAssume you want to try again.")
        continue

#---------------------------HTML file----------------------------------------------------------------------------
#open Html file
HtmlObject=open(htmlname,"w",encoding="utf-8")

#inport from text file
with open(filename, "r", encoding="utf-8") as SavedText:
    details= SavedText.read()

#Writing HTML file without harm to the structure
HtmlObject.write("<!DOCTYPE html>\n")
HtmlObject.write("<html lang='en'>\n")
HtmlObject.write("<head>\n")
HtmlObject.write("</head>\n")
HtmlObject.write("<body>\n")
HtmlObject.write("<pre>\n")          # Preserve structure of the main text file
HtmlObject.write(details)            # Write copyed details
HtmlObject.write("</pre>\n")
HtmlObject.write("</body>\n")
HtmlObject.write("</html>\n")

#closed Html file
HtmlObject.close()
```

10.2 demo.py – Demo mode module

```
#importing modules
import random
import time

def demogame( ):
    "This function is will work as Demo Mode for mathbro"
    #creating variables
    mode=None
    arithmeticoperator={}
    numbers=[]
    pro_question=''
    real_answer=''
    user_answer=''
    roundcount=None
    chances=None
    count=None
    status=None
    answercount=None
    presentage=None
    gameData=[]
    otherData=[]
    resulttext=None
    Question=None
    Uanswer=None
    Canswer=None

    #assigning as specification
    mode='Demo'
    arithmeticoperator={'operaton':['+',]}
    numbers=[0,1,2,3,4,5]
    roundcount=int(1)
    answercount=int(0)
    chances=3
    count=0

    # print Mode
    print("\n",mode," Mode")

    #creat loop for generate questions
    while count<chances:

        # Generating questions
        num1=(random.choice(numbers))
        operator=random.choice(arithmeticoperator['operaton'])
        num2=int(random.choice(numbers))

        #print round count
        time.sleep(1)
        print (f"\nQ {roundcount})")

        #Display question
        pro_question=f"{num1} {operator} {num2}"
        print("Question : ",pro_question,'?')

        #real answer to question
        real_answer=eval(pro_question)

        #getting answer from user
        #detect value errors and repert untill correct format enters
        while user_answer!=ValueError:
            try:
                user_answer=int(input("Your answer :"))
                break
            except:
                print("\nInvalid format! Try again.")

        time.sleep(1)

        #Find is answer correct or not and print correct or not with correct answer
        if real_answer==user_answer:
            print("correct")
            status="Correct"

            #Recode for answer count
            answercount=answercount+1
        else:
            print("Incorrect")
            status="Incorrect"

        # store Game Data
        gameData.append({'Question':pro_question,'user':user_answer,'com':real_answer,'sta':status,})

        #chance to break form loop
        count+=1
        roundcount+=1

    #Print Game data
    print("\nResult sheet")
    for y in range(len(gameData)):
        Question=gameData[y]["Question"]
        Uanswer=gameData[y]["user"]
        Canswer=gameData[y]["com"]

        if Uanswer==Canswer:
            resulttext = f"✓ {Question} = {Uanswer}"
        else:
            resulttext = f"✕ {Question} = {Uanswer} correct answer is {Canswer}"

        print(resulttext)

    #calculation presentage
    presentage=format(answercount/chances*100,'.1f')+"%"

    time.sleep(1)

    #print other details
    print("\nTotal Questions : ",chances)
    print("Correct questions : ",answercount)
    print("Presentage : ",presentage)
    print("Level :",mode)

    otherData.append({"total":chances,'corr':answercount,'pre':presentage,'mod':mode})
    return gameData,otherData
```

10.3 easy.py – Easy mode module

```
#importing modules
import random
import time

def easygame( ):
    "This function is will work as Easy Mode for mathbro"
    #creating variables
    mode=None
    arithmeticoperator={}
    numbers=[]
    pro_question=''
    real_answer=''
    user_answer=''
    roundcount=None
    chances=None
    count=None
    status=None
    answercount=None
    presentage=None
    gameData=[]
    otherData=[]
    resulttext=None
    Question=None
    Uanswer=None
    Canswer=None

    #assigning as specification
    mode='Easy'
    arithmeticoperator={'operaton':['+','-',]}
    numbers=[0,1,2,3,4,5,6,7,8,9,10]
    roundcount=int(1)
    answercount=int(0)
    chances=5
    count=0

    # print Mode
    print("\n",mode," Mode")

    #creat loop for generate questions
    while count<chances:

        # Generating questions
        num1=(random.choice(numbers))
        operator=random.choice(arithmeticoperator['operaton'])
    num2=int(random.choice(numbers))

    #print round count
    time.sleep(1)
    print (f"\nQ {roundcount})")

    #Display question
    pro_question=f"{num1} {operator} {num2}"
    print("Question : ",pro_question,'?')

    #real answer to question
    real_answer=eval(pro_question)

    #getting answer from user
    #detect value errors and repert untill correct format enters
    while user_answer!=ValueError:
        try:
            user_answer=int(input("Your answer :"))
            break
        except:
            print("\nInvalid format! Try again.")

    time.sleep(1)

    #Find is answer correct or not and print correct or not with correct answer
    if real_answer==user_answer:
        print("correct")
        status="Correct"

        #Recode for answer count
        answercount=answercount+1
    else:
        print("Incorrect")
        status="Incorrect"

    # store Game Data
    gameData.append({'Question':pro_question,'user':user_answer,'com':real_answer,'sta':status,})

    #chance to break form loop
    count+=1
    roundcount+=1

#Print Game data
print("\nResult sheet")
for y in range(len(gameData)):
    Question=gameData[y]["Question"]
    Uanswer=gameData[y]["user"]
    Canswer=gameData[y]["com"]

    if Uanswer==Canswer:
        resulttext = f"✓ {Question} = {Uanswer}"
    else:
        resulttext = f"✕ {Question} = {Uanswer} correct answer is {Canswer}"

    print(resulttext)

#calculation presentage
presentage=format(answercount/chances*100,'.1f')+"%"

time.sleep(1)

#print other details
print("\nTotal Questions : ",chances)
print("Correct questions : ",answercount)
print("Presentage : ",presentage)
print("Level :",mode)

otherData.append({"total":chances,'corr':answercount,'pre':presentage,'mod':mode})
return gameData,otherData
```

10.4 medium.py – Medium mode module

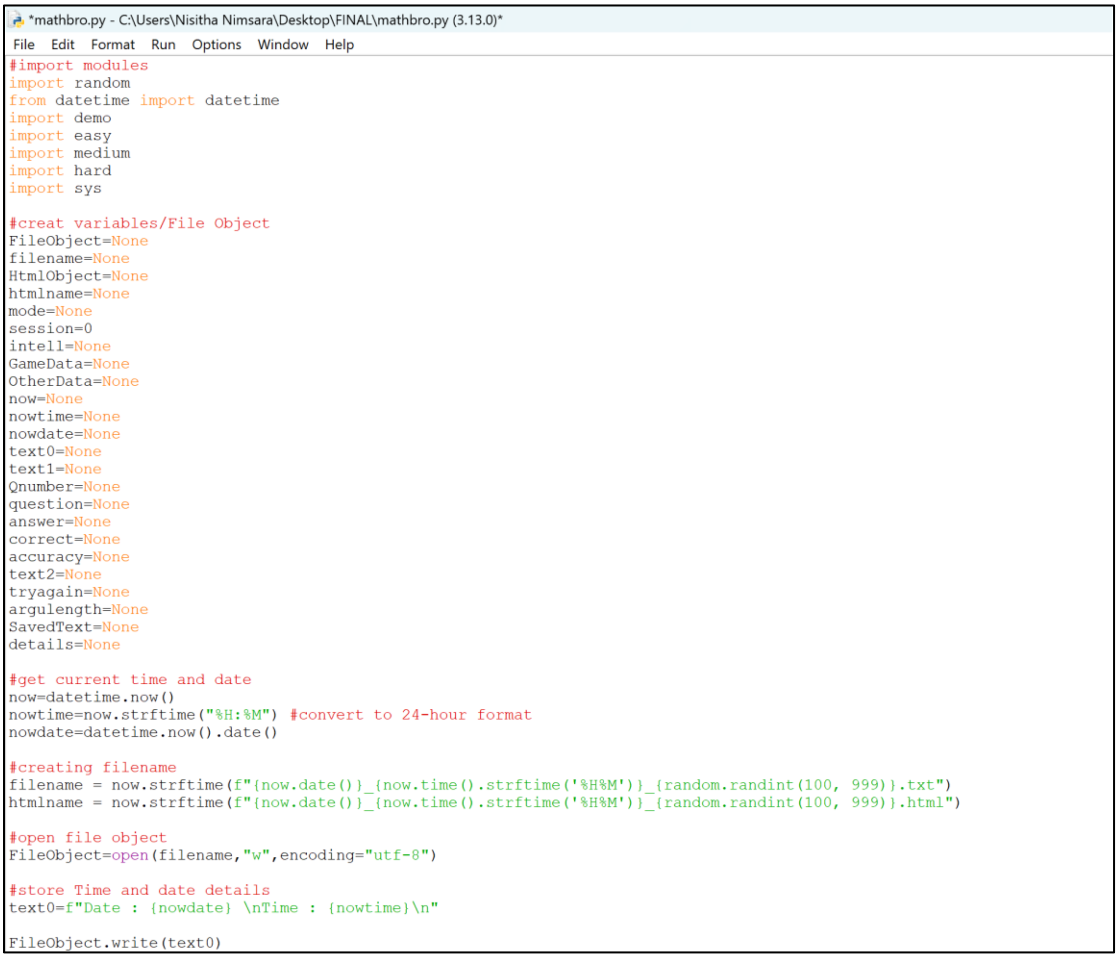
```
#importing modules
import random
import time

def mediumgame( ):
    "This function is will work as Medium Mode for mathbro"
    #creating variables
    mode=None
    arithmeticoperator={}
    numbers=[]
    pro_question=''
    real_answer=''
    user_answer=''
    roundcount=None
    chances=None
    count=None
    status=None
    answercount=None
    presentage=None
    gameData=[]
    otherData=[]
    resulttext=None
    Question=None
    Uanswer=None
    Canswer=None

    #assigning as specification
    mode='Medium'
    arithmeticoperator={'operaton':['+','-',]}
    numbers=[0,1,2,3,4,5,6,7,8,9,10]
    roundcount=int(1)
    answercount=int(0)
    chances=10
    count=0

    # print Mode
    print("\n",mode," Mode")

    #creat loop for generate questions
    while count<chances:

        # Generating questions
        num1=(random.choice(numbers))
        operator=random.choice(arithmeticoperator['operaton'])
    num2=int(random.choice(numbers))

    #print round count
    time.sleep(1)
    print (f"\nQ {roundcount})")

    #Display question
    pro_question=f"{num1} {operator} {num2}"
    print("Question : ",pro_question,'?')

    #real answer to question
    real_answer=eval(pro_question)

    #getting answer from user
    #detect value errors and repert untill correct format enters
    while user_answer!=ValueError:
        try:
            user_answer=int(input("Your answer :"))
            break
        except:
            print("\nInvalid format! Try again.")

    time.sleep(1)

    #Find is answer correct or not and print correct or not with correct answer
    if real_answer==user_answer:
        print("correct")
        status="Correct"

        #Recode for answer count
        answercount=answercount+1
    else:
        print("Incorrect")
        status="Incorrect"

    # store Game Data
    gameData.append({'Question':pro_question,'user':user_answer,'com':real_answer,'sta':status,})

    #chance to break form loop
    count+=1
    roundcount+=1

#Print Game data
print("\nResult sheet")
for y in range(len(gameData)):
    Question=gameData[y]["Question"]
    Uanswer=gameData[y]["user"]
    Canswer=gameData[y]["com"]

    if Uanswer==Canswer:
        resulttext = f"✓ {Question} = {Uanswer}"
    else:
        resulttext = f"✕ {Question} = {Uanswer} correct answer is {Canswer}"

    print(resulttext)

#calculation presentage
presentage=format(answercount/chances*100,'.1f')+"%"

time.sleep(1)

#print other details
print("\nTotal Questions : ",chances)
print("Correct questions : ",answercount)
print("Presentage : ",presentage)
print("Level :",mode)

otherData.append({"total":chances,'corr':answercount,'pre':presentage,'mod':mode})
return gameData,otherData
```

10.5 hard.py - Hard mode module

```
#importing modules
import random
import time

def hardgame( ):
    "This function is will work as Hard Mode for mathbro"
    #creating variables
    mode=None
    arithmeticoperator={}
    numbers=[]
    pro_question="
    real_answer="
    user_answer="
    roundcount=None
    chances=None
    count=None
    status=None
    answercount=None
    presentage=None
    gameData=[]
    otherData=[]
    resulttext=None
    Question=None
    Uanswer=None
    Canswer=None

    #assigning as specification
    mode='Hard'
    arithmeticoperator={'operaton':['+','-','*']}
    numbers=[0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20]
    roundcount=int(1)
    answercount=int(0)
    chances=10
    count=0

    # print Mode
    print("\n",mode," Mode")

    #creat loop for generate questions
    while count<chances:

        # Generating questions
        num1=(random.choice(numbers))
        operator=random.choice(arithmeticoperator['operaton'])
    num2=int(random.choice(numbers))

    #print round count
    time.sleep(1)
    print (f"\nQ {roundcount})")

    #Display question
    pro_question=f"{num1} {operator} {num2}"
    print("Question : ",pro_question,'?')

    #real answer to question
    real_answer=eval(pro_question)

    #getting answer from user
    #detect value errors and repert untill correct format enters
    while user_answer!=ValueError:
        try:
            user_answer=int(input("Your answer :"))
            break
        except:
            print("\nInvalid format! Try again.")

    time.sleep(1)

    #Find is answer correct or not and print correct or not with correct answer
    if real_answer==user_answer:
        print("correct")
        status="Correct"

        #Recode for answer count
        answercount=answercount+1
    else:
        print("Incorrect")
        status="Incorrect"

    # store Game Data
    gameData.append({'Question':pro_question,'user':user_answer,'com':real_answer,'sta':status,})

    #chance to break form loop
    count+=1
    roundcount+=1

#Print Game data
print("\nResult sheet")
for y in range(len(gameData)):
    Question=gameData[y]["Question"]
    Uanswer=gameData[y]["user"]
    Canswer=gameData[y]["com"]

    if Uanswer==Canswer:
        resulttext = f"✓ {Question} = {Uanswer}"
    else:
        resulttext = f"✕ {Question} = {Uanswer} correct answer is {Canswer}"

    print(resulttext)

#calculation presentage
presentage=format(answercount/chances*100,'.1f')+"%"

time.sleep(1)

#print other details
print("\nTotal Questions : ",chances)
print("Correct questions : ",answercount)
print("Presentage : ",presentage)
print("Level :",mode)

otherData.append({"total":chances,'corr':answercount,'pre':presentage,'mod':mode})
return gameData,otherData
```

11. Screenshots of the code and explanation

11.1 mathbro (Main) part 1

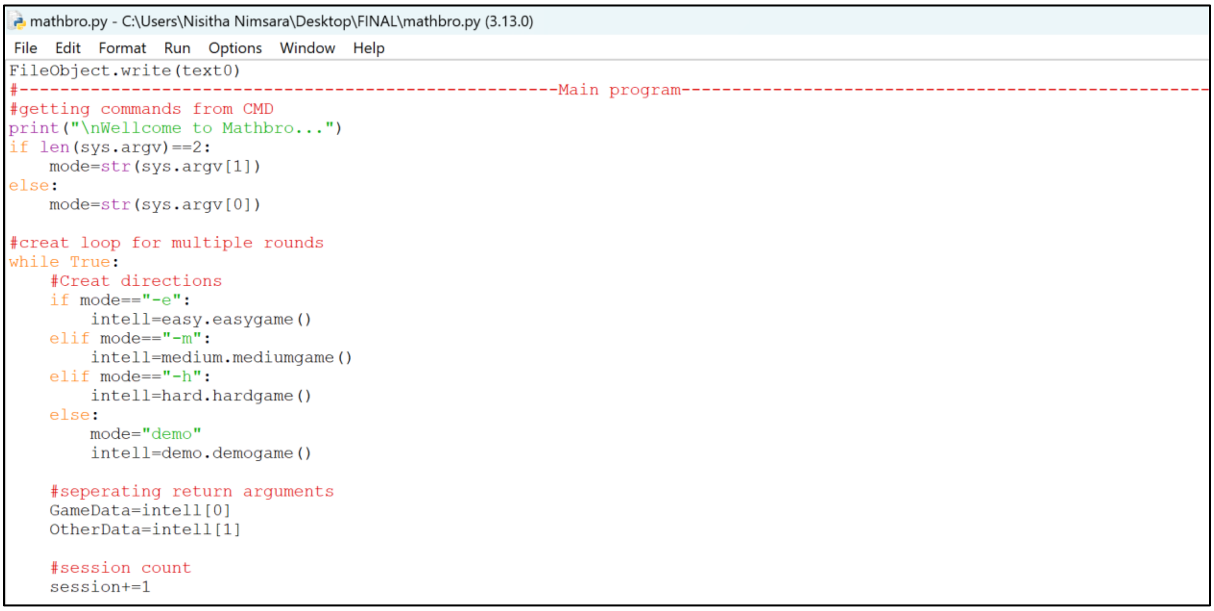

mathbro.py - C:\Users\Nisitha Nimsara\Desktop\FINAL\mathbro.py (3.13.0)
File Edit Format Run Options Window Help

```
#import modules
import random
from datetime import datetime
import demo
import easy
import medium
import hard
import sys

#creat variables/File Object
FileObject=None
filename=None
HtmlObject=None
htmlname=None
mode=None
session=0
intell=None
GameData=None
OtherData=None
now=None
nowtime=None
nowdate=None
text0=None
text1=None
Qnumber=None
question=None
answer=None
correct=None
accuracy=None
text2=None
tryagain=None
argulength=None
SavedText=None
details=None

#get current time and date
now=datetime.now()
nowtime=now.strftime("%H:%M") #convert to 24-hour format
nowdate=datetime.now().date()

#creating filename
filename = now.strftime(f"{now.date()}_{now.time().strftime('%H%M')}_{random.randint(100, 999)}.txt")
htmlname = now.strftime(f"{now.date()}_{now.time().strftime('%H%M')}_{random.randint(100, 999)}.html")

#open file object
FileObject=open(filename,"w",encoding="utf-8")

#store Time and date details
text0=f"Date : {nowdate} \nTime : {nowtime}\n"

FileObject.write(text0)
```

Figure 1 mathbro.py code part 1

Starting the main program by importing built-in modules (random,datetime,sys) and user-built modules (demo,easy,medium,hard). After that created all variables and assigning value to None or 0. After that I get the time and convert in into 24-hour format and get date. Then I created the file names for TXT format and HTML format. Then open the text file object(UTF-8 is a unicode character encoding method) and write date and time.

11.2 mathbro (Main) part 2

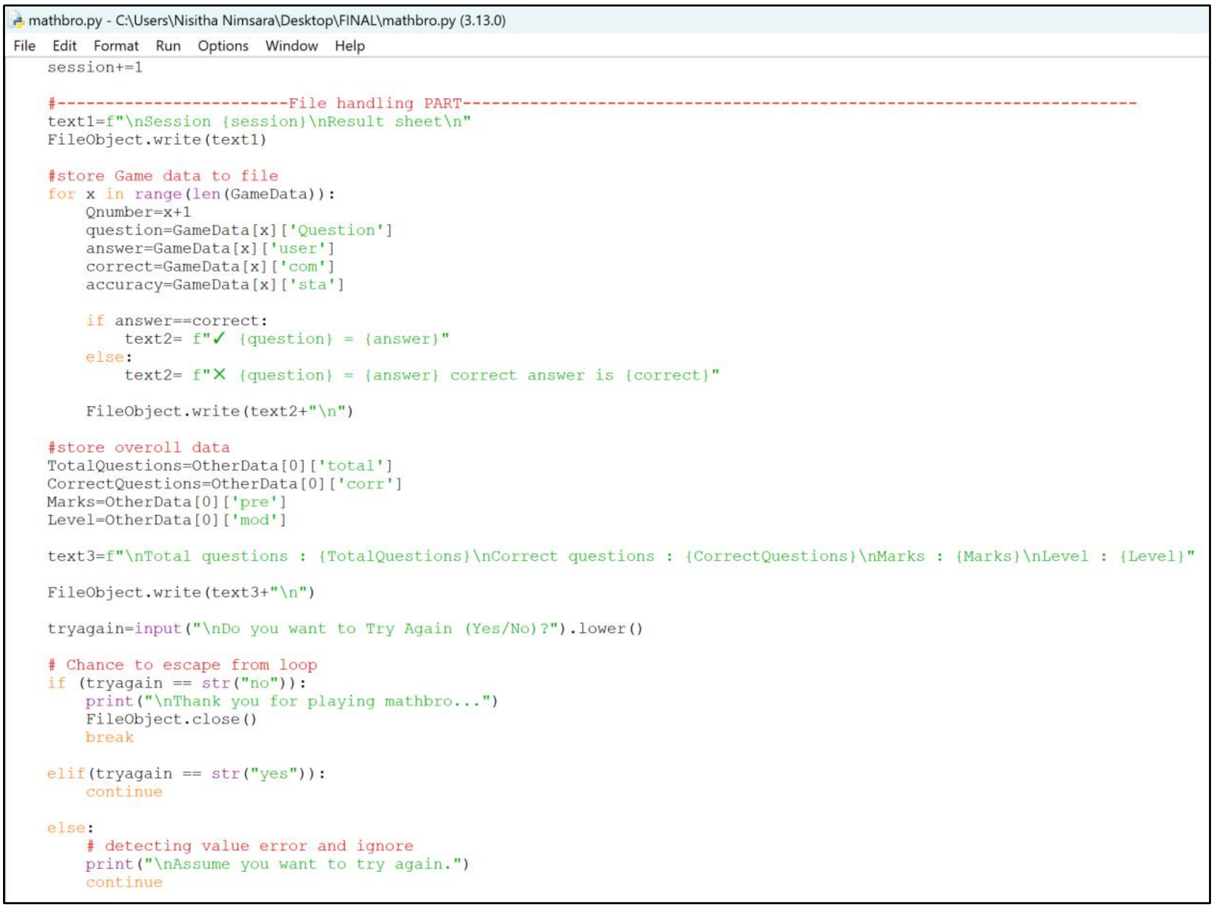

mathbro.py - C:\Users\Nisitha Nimsara\Desktop\FINAL\mathbro.py (3.13.0)
File
Edit
Format
Run
Options
Window
Help

```
FileObject.write(text0)
#-----------------------------------------------------Main program----------------------------------------------------
#getting commands from CMD
print("\nWellcome to Mathbro...")
if len(sys.argv)==2:
    mode=str(sys.argv[1])
else:
    mode=str(sys.argv[0])

#creat loop for multiple rounds
while True:
    #Creat directions
    if mode=="-e":
        intell=easy.easygame()
    elif mode=="-m":
        intell=medium.mediumgame()
    elif mode=="-h":
        intell=hard.hardgame()
    else:
        mode="demo"
        intell=demo.demogame()

    #seperating return arguments
    GameData=intell[0]
    OtherData=intell[1]

    #session count
    session+=1
```

Figure 2 mathbro.py code part 2

After writing date and time in text file, I started the main program. As first it will display the welcome massages to user in command prompt. Then compare the length of the argument. If the length 2, argument index 1 value assign to the mode variable otherwise argument index 0 value assign to the mode variable. Then I open while true repetition function to repeat main program infinity. Then I placed the if function to choose mode according to the user inputs. And call the suitable function. And turn the user to suitable mode(each mode will explain later). And at the same time, I put the intel variable to copy values than return from each function(each function returns 2 arguments). After that I divided it into 2 parts as game data and other data.

11.3 mathbro (Main) part 3


mathbro.py - C:\Users\Nisitha Nimsara\Desktop\FINAL\mathbro.py (3.13.0)
File Edit Format Run Options Window Help

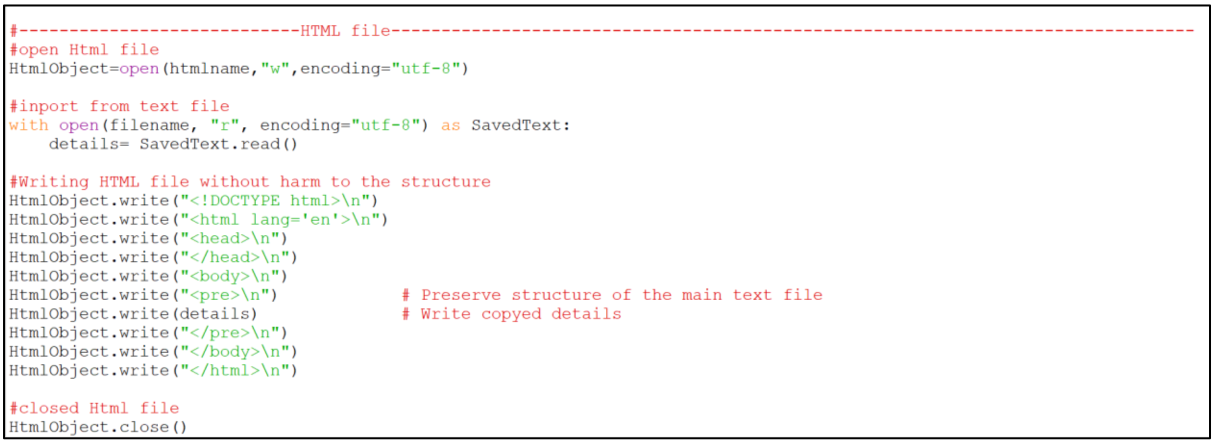
```
    session+=1

    #------------------------File handling PART----------------------------------------------------------------------
    text1=f"\nSession {session}\nResult sheet\n"
    FileObject.write(text1)

    #store Game data to file
    for x in range(len(GameData)):
        Qnumber=x+1
        question=GameData[x]['Question']
        answer=GameData[x]['user']
        correct=GameData[x]['com']
        accuracy=GameData[x]['sta']

        if answer==correct:
            text2= f"✓ {question} = {answer}"
        else:
            text2= f"✕ {question} = {answer} correct answer is {correct}"

        FileObject.write(text2+"\n")

    #store overoll data
    TotalQuestions=OtherData[0]['total']
    CorrectQuestions=OtherData[0]['corr']
    Marks=OtherData[0]['pre']
    Level=OtherData[0]['mod']

    text3=f"\nTotal questions : {TotalQuestions}\nCorrect questions : {CorrectQuestions}\nMarks : {Marks}\nLevel : {Level}"

    FileObject.write(text3+"\n")

    tryagain=input("\nDo you want to Try Again (Yes/No)?").lower()

    # Chance to escape from loop
    if (tryagain == str("no")):
        print("\nThank you for playing mathbro...")
        FileObject.close()
        break

    elif(tryagain == str("yes")):
        continue

    else:
        # detecting value error and ignore
        print("\nAssume you want to try again.")
        continue
```

Figure 3 mathbro.py code part 3

After that adding session variable value to 1 and start the text file handling part. In this part I filter the Question that user received, user’s answer, correct answer and the state of the answer. And filter it into correct answer or not. And store them in txt file one by one. After that total question count, correct question count, percentage and game mode also write inside that file.

After that the player receive and option to paly again or not. If the player prefers to play again, type yes or any button. then it will go back to the selected mode and repeat this whole process again. If the player prefers to stop, then the players receive an appreciation for playing and break then loop and close that previously open text file object.

11.4 mathbro (Main) part 4

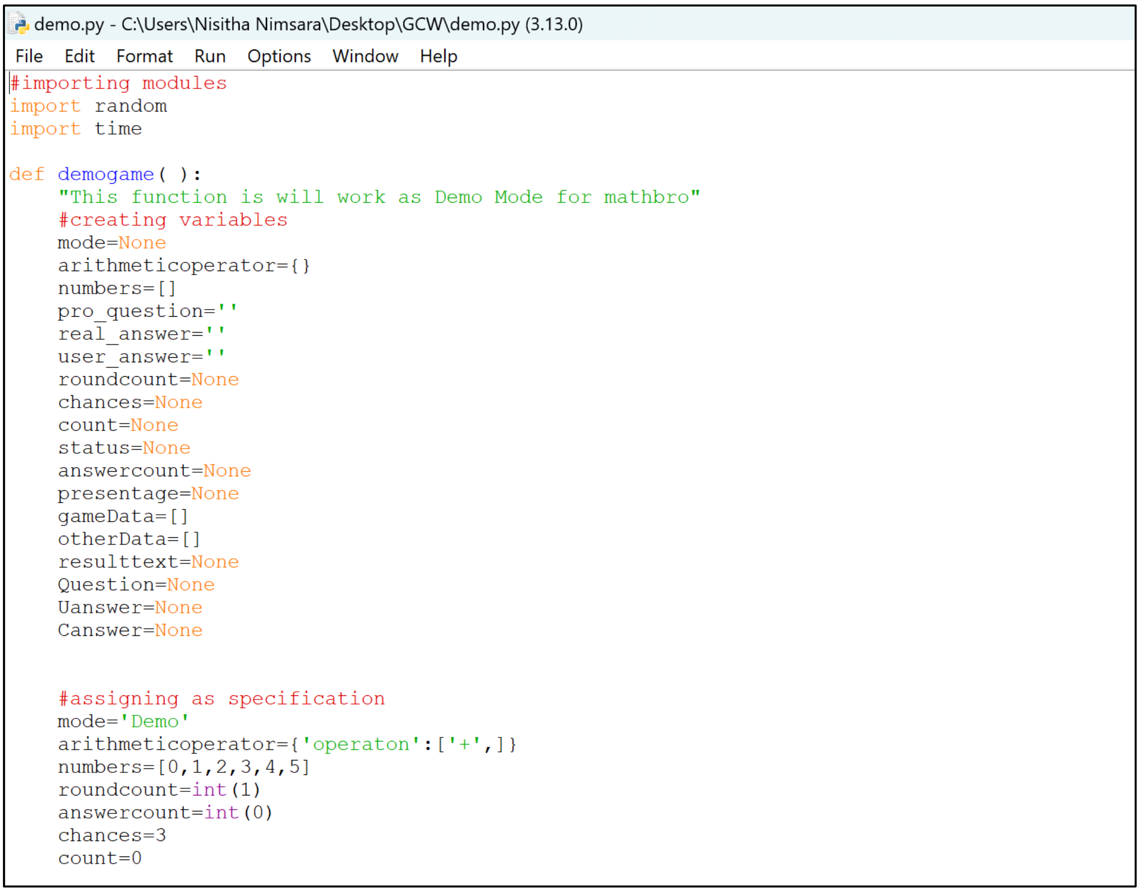


```
#----------------------------HTML file--------------------------------------------------------------------------------
#open Html file
HtmlObject=open(htmlname,"w",encoding="utf-8")

#inport from text file
with open(filename, "r", encoding="utf-8") as SavedText:
    details= SavedText.read()

#Writing HTML file without harm to the structure
HtmlObject.write("<!DOCTYPE html>\n")
HtmlObject.write("<html lang='en'>\n")
HtmlObject.write("<head>\n")
HtmlObject.write("</head>\n")
HtmlObject.write("<body>\n")
HtmlObject.write("<pre>\n")            # Preserve structure of the main text file
HtmlObject.write(details)              # Write copyed details
HtmlObject.write("</pre>\n")
HtmlObject.write("</body>\n")
HtmlObject.write("</html>\n")

#closed Html file
HtmlObject.close()
```

Figure 4 mathbro.py code part 4

After the closing of the text file object, html file object will open. And copy everything to the details variable. Then step by step html structure building and put everything that copies from text file into html body part. After that Html file object closes.

11.5 Module part 1

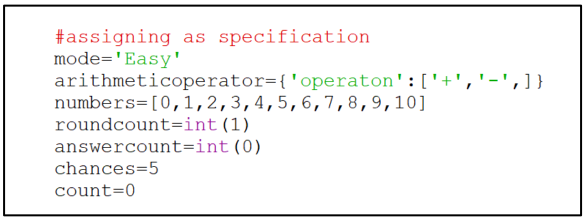

demo.py - C:\Users\Nisitha Nimsara\Desktop\GCW\demo.py (3.13.0)
File
Edit
Format
Run
Options
Window
Help

```
#importing modules
import random
import time

def demogame( ):
    "This function is will work as Demo Mode for mathbro"
    #creating variables
    mode=None
    arithmeticoperator={}
    numbers=[]
    pro_question=''
    real_answer=''
    user_answer=''
    roundcount=None
    chances=None
    count=None
    status=None
    answercount=None
    presentage=None
    gameData=[]
    otherData=[]
    resulttext=None
    Question=None
    Uanswer=None
    Canswer=None


    #assigning as specification
    mode='Demo'
    arithmeticoperator={'operaton':['+',]}
    numbers=[0,1,2,3,4,5]
    roundcount=int(1)
    answercount=int(0)
    chances=3
    count=0
```

Figure 5 demo.py code part 1

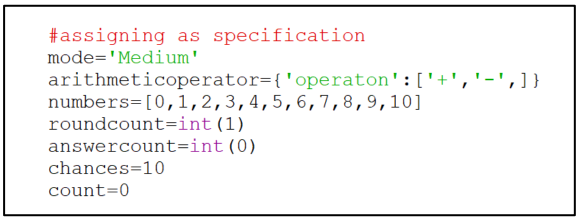
As first, module file import random and time building functions. Then start the mode function and creating variables and lists then assign values to then as none or empty.

Then assign values according to the game mode instructions. In this picture, this is the structure of demo mode, these values unique to only this mode. In other modes, all structures are same but only this ware different.



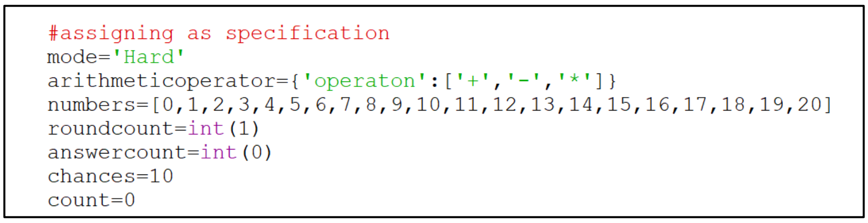
```
#assigning as specification
mode='Easy'
arithmeticoperator={'operaton':['+','-',]}
numbers=[0,1,2,3,4,5,6,7,8,9,10]
roundcount=int(1)
answercount=int(0)
chances=5
count=0
```

Figure 6 easy mode specification



```
#assigning as specification
mode='Medium'
arithmeticoperator={'operaton':['+','-',]}
numbers=[0,1,2,3,4,5,6,7,8,9,10]
roundcount=int(1)
answercount=int(0)
chances=10
count=0
```

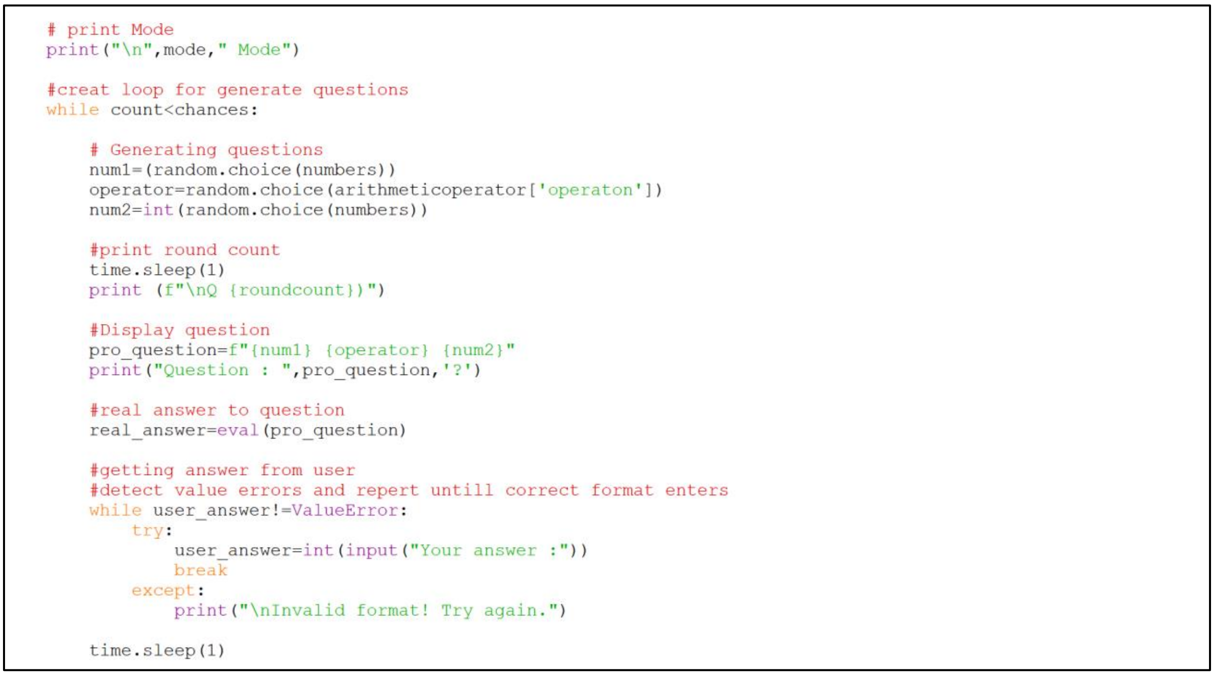
Figure 7 medium mode specification



```
#assigning as specification
mode='Hard'
arithmeticoperator={'operaton':['+','-','*']}
numbers=[0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20]
roundcount=int(1)
answercount=int(0)
chances=10
count=0
```

Figure 8 Hard mode specification

11.6 Module part 2

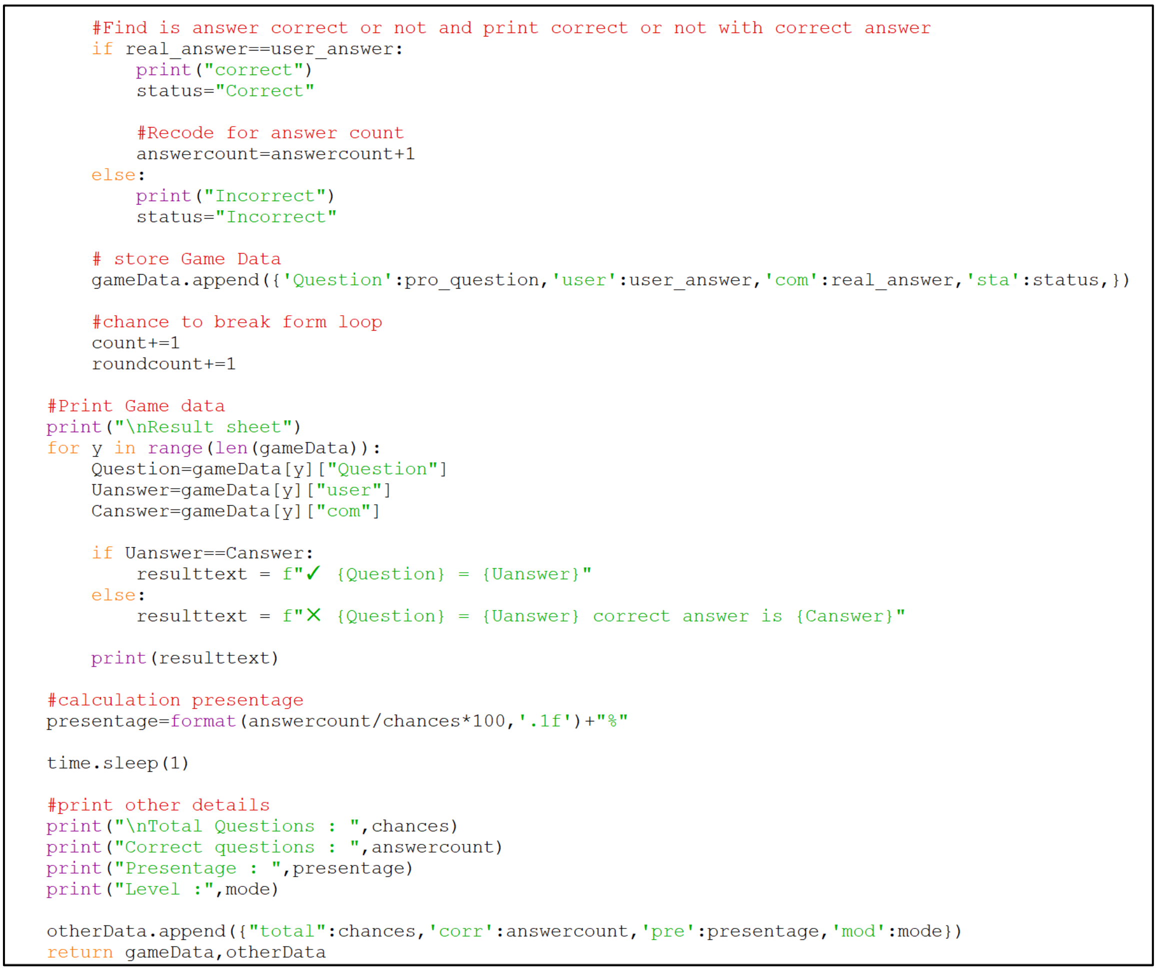


```
# print Mode
print("\n",mode," Mode")

#creat loop for generate questions
while count<chances:

    # Generating questions
    num1=(random.choice(numbers))
    operator=random.choice(arithmeticoperator['operaton'])
    num2=int(random.choice(numbers))

    #print round count
    time.sleep(1)
    print (f"\nQ {roundcount})")

    #Display question
    pro_question=f"{num1} {operator} {num2}"
    print("Question : ",pro_question,'?')

    #real answer to question
    real_answer=eval(pro_question)

    #getting answer from user
    #detect value errors and repert untill correct format enters
    while user_answer!=ValueError:
        try:
            user_answer=int(input("Your answer :"))
            break
        except:
            print("\nInvalid format! Try again.")

    time.sleep(1)
```

Figure 9 demo.py code part 2

After assigning values, set a print function to display the mode to the user. Then the question generation part will start, and the player receives a number of questions according to the specification. When question generating I uses random.choice function to choose form list. And also after every question player received 1secound of break (time.seek(1)) for more interacting. Somehow if player input wrong input format, then that error will catch from try except method and give an massage to player and continue from last ended place.

11.7 Module part 3



```
        #Find is answer correct or not and print correct or not with correct answer
        if real_answer==user_answer:
            print("correct")
            status="Correct"

            #Recode for answer count
            answercount=answercount+1
        else:
            print("Incorrect")
            status="Incorrect"

        # store Game Data
        gameData.append({'Question':pro_question,'user':user_answer,'com':real_answer,'sta':status,})

        #chance to break form loop
        count+=1
        roundcount+=1

    #Print Game data
    print("\nResult sheet")
    for y in range(len(gameData)):
        Question=gameData[y]["Question"]
        Uanswer=gameData[y]["user"]
        Canswer=gameData[y]["com"]

        if Uanswer==Canswer:
            resulttext = f"✓ {Question} = {Uanswer}"
        else:
            resulttext = f"✕ {Question} = {Uanswer} correct answer is {Canswer}"

        print(resulttext)

    #calculation presentage
    presentage=format(answercount/chances*100,'.1f')+"%"

    time.sleep(1)

    #print other details
    print("\nTotal Questions : ",chances)
    print("Correct questions : ",answercount)
    print("Presentage : ",presentage)
    print("Level :",mode)

    otherData.append({"total":chances,'corr':answercount,'pre':presentage,'mod':mode})
    return gameData,otherData
```

Figure 10 demo.py code part 3

After the player gives the answer, the program will filter whether the answer is correct or not using if function. Then if it is correct, it will show “correct” and if it is wrong, program will show “incorrect”. After that all data related to that question will stored to list(append) using dictionary format. Then count variable and round count variable modified by 1(addition 1).

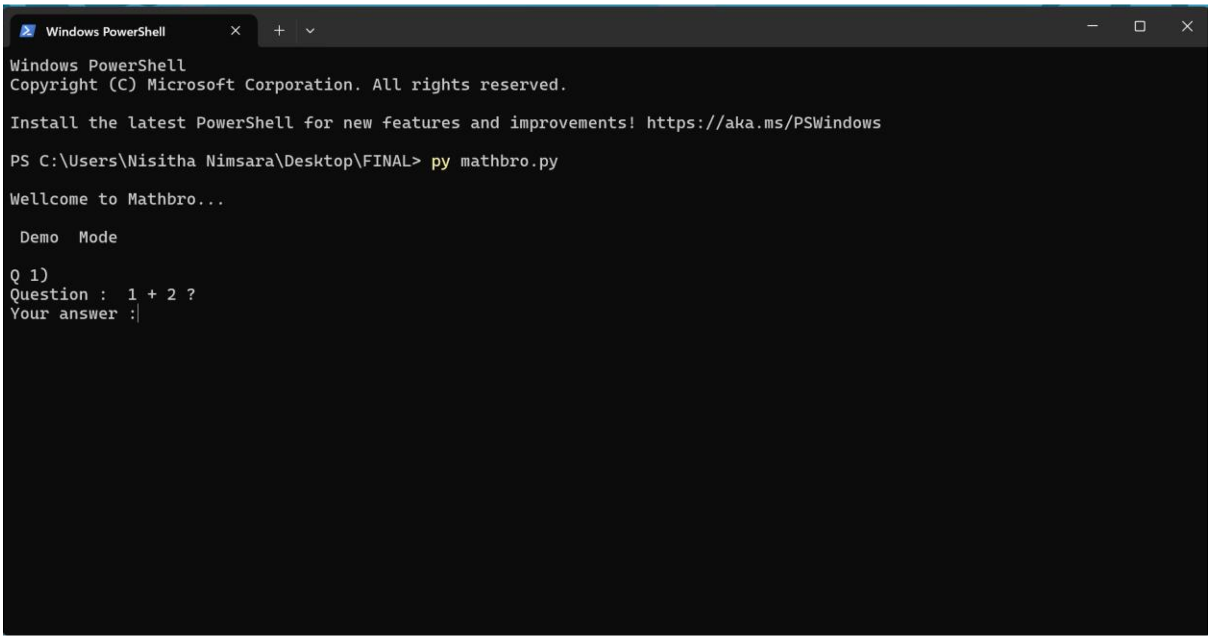
After storing all question details, result sheet will display to player using for loop and it will extract all data one by one. If user answer is correct, ✓ mark, question and user answer, otherwise ✕ mart, question ,user answer and correct answer will display. After this 1 secound break and then total question count, correct answer count, percentage and mode also display to the player. For finally these data also stored to other data list and return index10 as game data and index 1 as other data to the main program.

12. Test cases

Case No	Input	Expected Output	Remarks
1)	Input only mathbro.py	Enter to the Demo mode	Pass
2)	Input mathbro.py and -m	Enter to the medium mode	Pass
3)	Enter correct answer	Display correct	Pass
4)	Enter wrong answer	Display Incorrect	Pass
5)	Enter Invalid answer	Display Error message and resume from last place	Pass
6)	Complete all 10 questions (medium mode)	Display result sheet and round summary	Pass
7)	Enter Try again = ‘yes’	Start next round in same mode	Pass
8)	Enter Try again = ‘no’	Display appreciation massage	Pass
9)	Save in text file	Store result sheet with summary	Pass
10)	Save in Html file	Store result sheet with summary	Pass

Table 1 Test cases

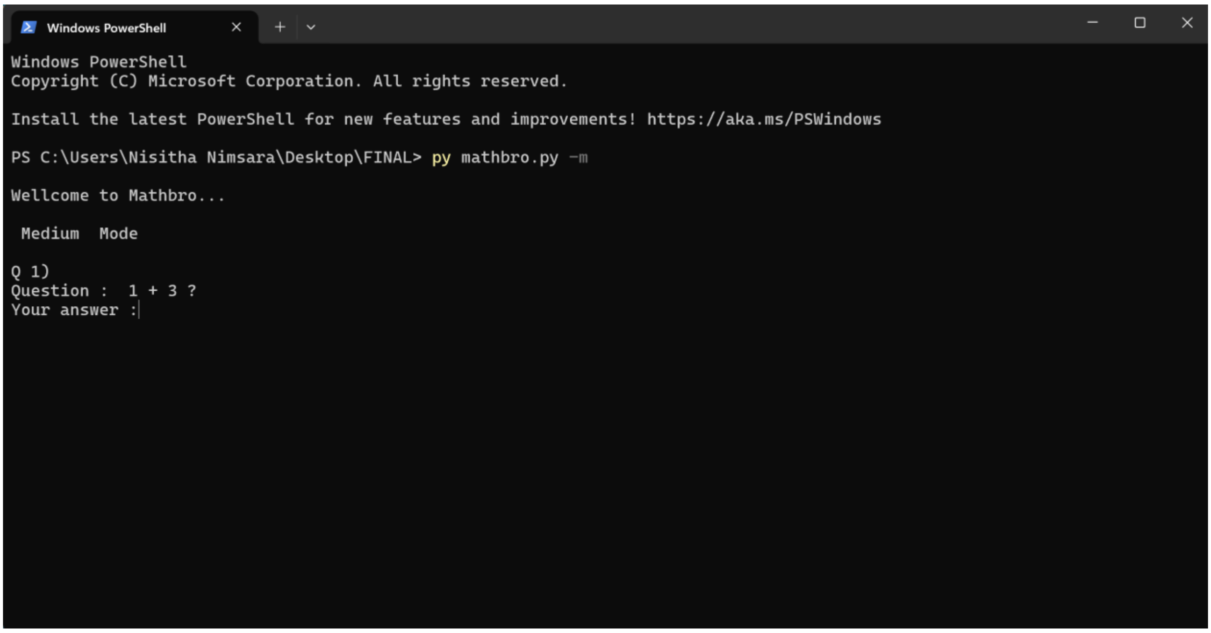
12.1 Test Case 1



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\Nisitha Nimsara\Desktop\FINAL> py mathbro.py

Wellcome to Mathbro...

 Demo  Mode

Q 1)
Question :  1 + 2 ?
Your answer :
```

Figure 11 Test case 1

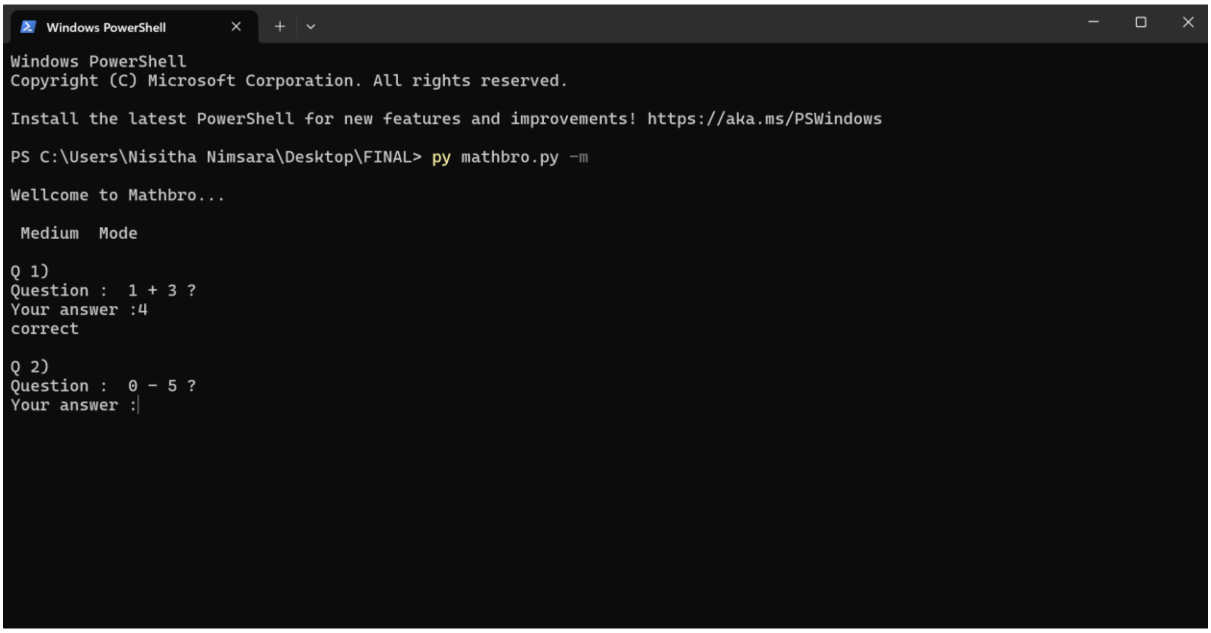
12.2 Test Case 2



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\Nisitha Nimsara\Desktop\FINAL> py mathbro.py -m

Wellcome to Mathbro...

 Medium  Mode

Q 1)
Question :  1 + 3 ?
Your answer :
```

Figure 12 Test case 2

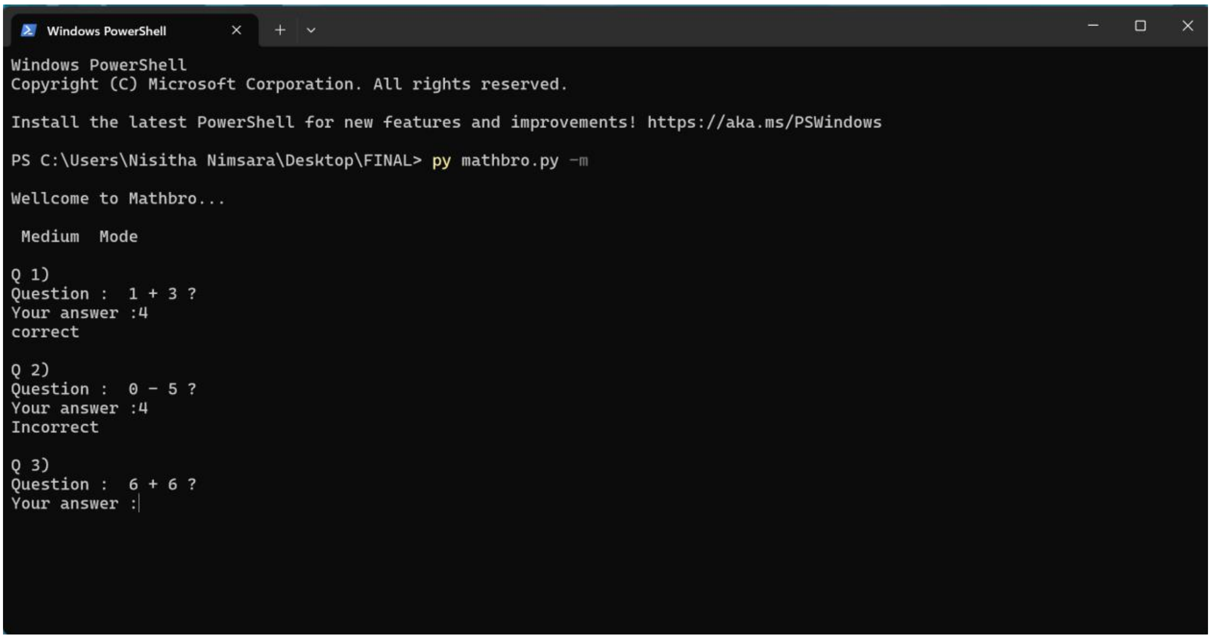
12.3 Test Case 3



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\Nisitha Nimsara\Desktop\FINAL> py mathbro.py -m

Wellcome to Mathbro...

 Medium  Mode

Q 1)
Question :  1 + 3 ?
Your answer :4
correct

Q 2)
Question :  0 - 5 ?
Your answer :
```

Figure 13 Test case 3

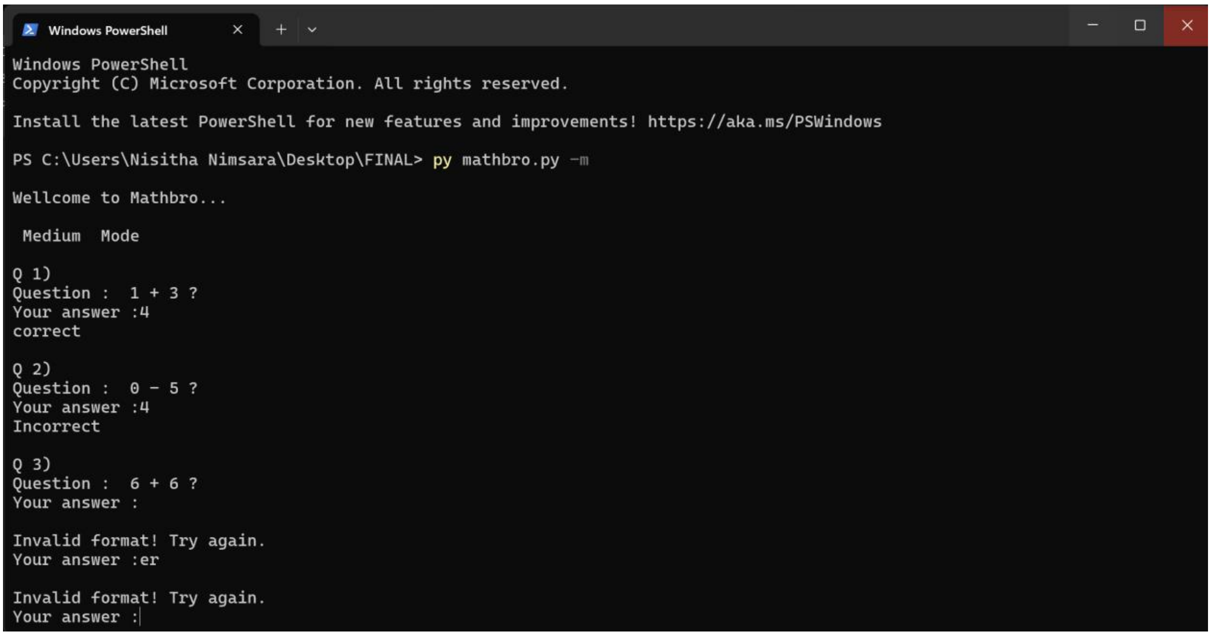
12.4 Test Case 4



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\Nisitha Nimsara\Desktop\FINAL> py mathbro.py -m

Wellcome to Mathbro...

 Medium  Mode

Q 1)
Question :  1 + 3 ?
Your answer :4
correct

Q 2)
Question :  0 - 5 ?
Your answer :4
Incorrect

Q 3)
Question :  6 + 6 ?
Your answer :
```

Figure 14 Test case 4

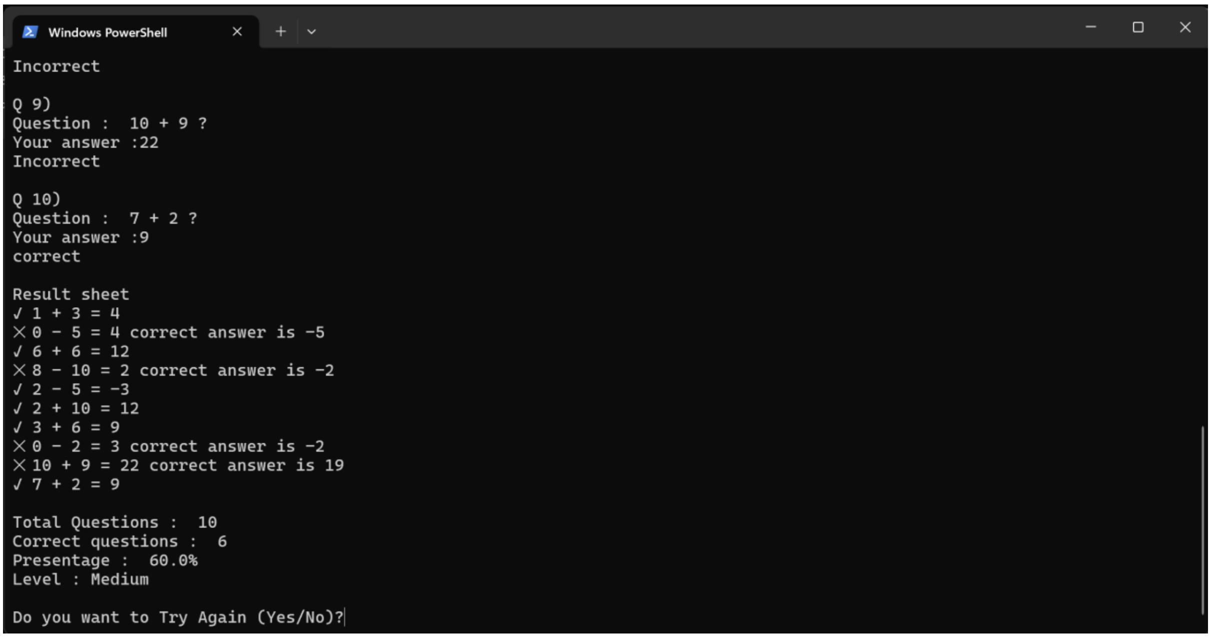
12.5 Test Case 5



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\Nisitha Nimsara\Desktop\FINAL> py mathbro.py -m

Wellcome to Mathbro...

 Medium  Mode

Q 1)
Question :  1 + 3 ?
Your answer :4
correct

Q 2)
Question :  0 - 5 ?
Your answer :4
Incorrect

Q 3)
Question :  6 + 6 ?
Your answer :

Invalid format! Try again.
Your answer :er

Invalid format! Try again.
Your answer :
```

Figure 15 Test case 5

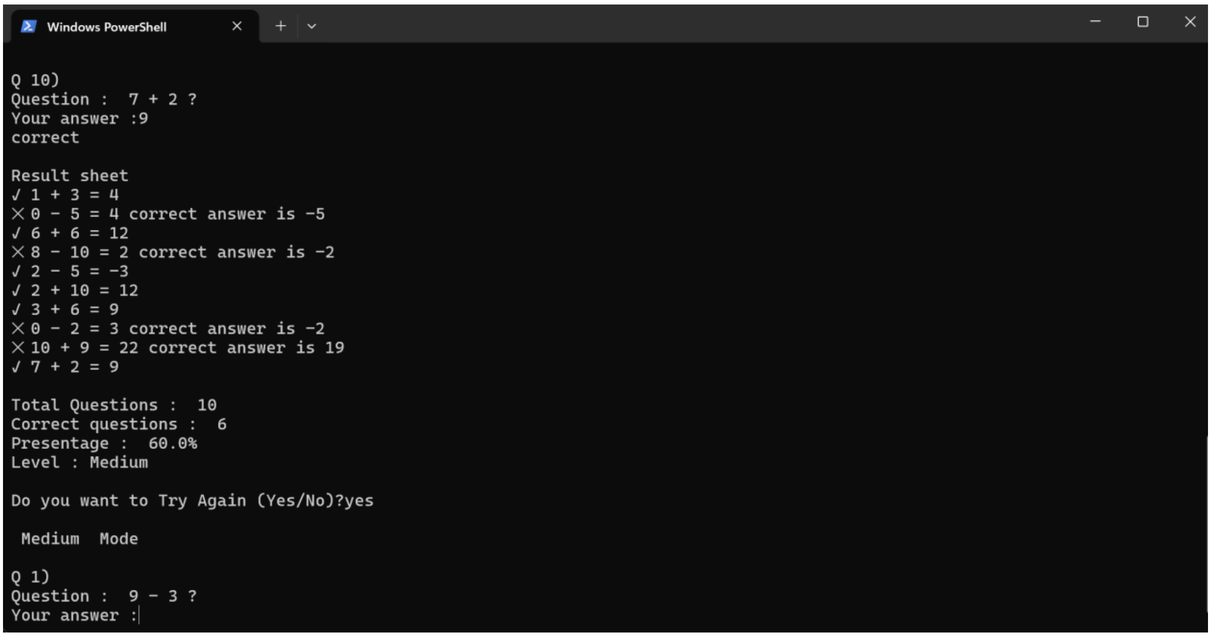
12.6 Test Case 6



```
Incorrect

Q 9)
Question :  10 + 9 ?
Your answer :22
Incorrect

Q 10)
Question :  7 + 2 ?
Your answer :9
correct

Result sheet
√ 1 + 3 = 4
× 0 - 5 = 4 correct answer is -5
√ 6 + 6 = 12
× 8 - 10 = 2 correct answer is -2
√ 2 - 5 = -3
√ 2 + 10 = 12
√ 3 + 6 = 9
× 0 - 2 = 3 correct answer is -2
× 10 + 9 = 22 correct answer is 19
√ 7 + 2 = 9

Total Questions :  10
Correct questions :  6
Presentage :  60.0%
Level : Medium

Do you want to Try Again (Yes/No)?
```

Figure 16 Test case 6

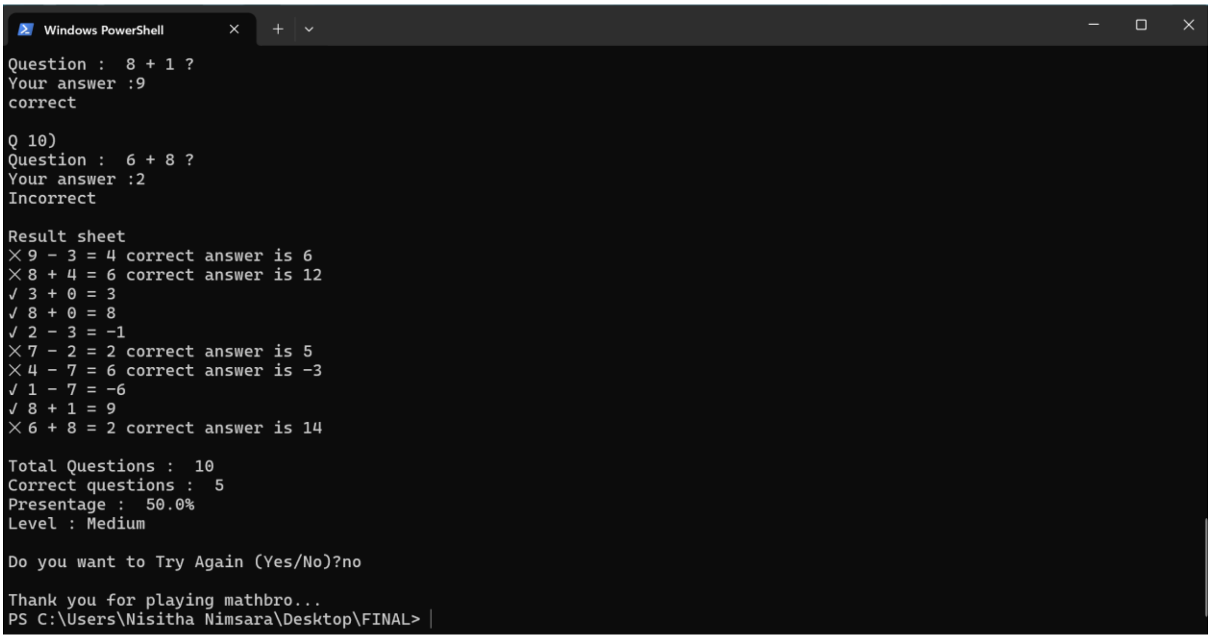
12.7 Test Case 7



```
Q 10)
Question :  7 + 2 ?
Your answer :9
correct

Result sheet
✓ 1 + 3 = 4
✕ 0 - 5 = 4 correct answer is -5
✓ 6 + 6 = 12
✕ 8 - 10 = 2 correct answer is -2
✓ 2 - 5 = -3
✓ 2 + 10 = 12
✓ 3 + 6 = 9
✕ 0 - 2 = 3 correct answer is -2
✕ 10 + 9 = 22 correct answer is 19
✓ 7 + 2 = 9

Total Questions :  10
Correct questions :  6
Presentage :  60.0%
Level : Medium

Do you want to Try Again (Yes/No)?yes

 Medium  Mode

Q 1)
Question :  9 - 3 ?
Your answer :
```

Figure 17 Test case 7

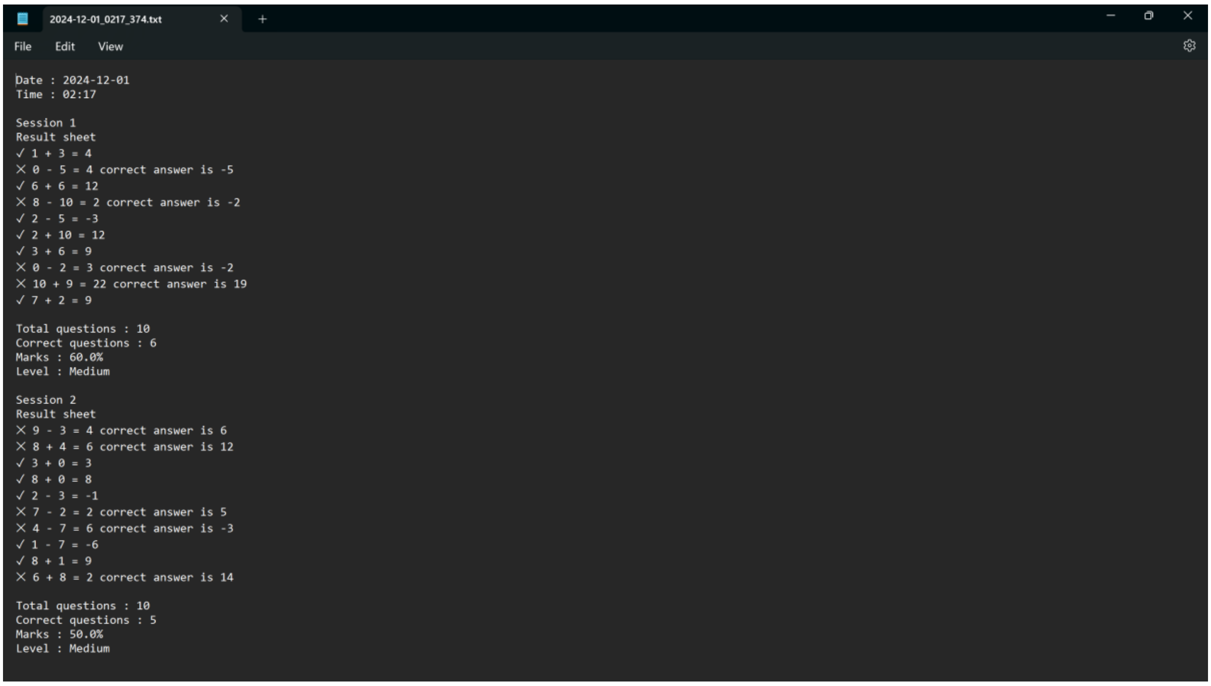
12.8 Test Case 8



```
Question :  8 + 1 ?
Your answer :9
correct

Q 10)
Question :  6 + 8 ?
Your answer :2
Incorrect

Result sheet
✕ 9 - 3 = 4 correct answer is 6
✕ 8 + 4 = 6 correct answer is 12
✓ 3 + 0 = 3
✓ 8 + 0 = 8
✓ 2 - 3 = -1
✕ 7 - 2 = 2 correct answer is 5
✕ 4 - 7 = 6 correct answer is -3
✓ 1 - 7 = -6
✓ 8 + 1 = 9
✕ 6 + 8 = 2 correct answer is 14

Total Questions :  10
Correct questions :  5
Presentage :  50.0%
Level : Medium

Do you want to Try Again (Yes/No)?no

Thank you for playing mathbro...
PS C:\Users\Nisitha Nimsara\Desktop\FINAL>
```

Figure 18 Test case 8

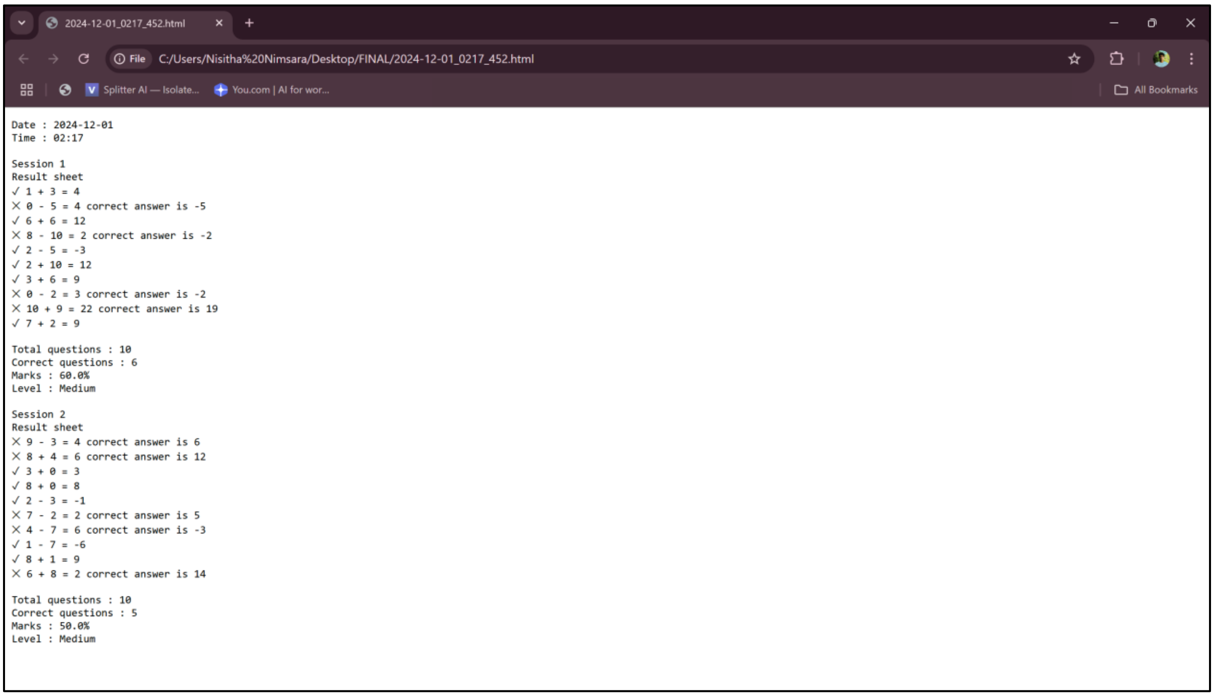
12.9 Test Case 9



```
Date : 2024-12-01
Time : 02:17

Session 1
Result sheet
✓ 1 + 3 = 4
✕ 0 - 5 = 4 correct answer is -5
✓ 6 + 6 = 12
✕ 8 - 10 = 2 correct answer is -2
✓ 2 - 5 = -3
✓ 2 + 10 = 12
✓ 3 + 6 = 9
✕ 0 - 2 = 3 correct answer is -2
✕ 10 + 9 = 22 correct answer is 19
✓ 7 + 2 = 9

Total questions : 10
Correct questions : 6
Marks : 60.0%
Level : Medium

Session 2
Result sheet
✕ 9 - 3 = 4 correct answer is 6
✕ 8 + 4 = 6 correct answer is 12
✓ 3 + 0 = 3
✓ 8 + 0 = 8
✓ 2 - 3 = -1
✕ 7 - 2 = 2 correct answer is 5
✕ 4 - 7 = 6 correct answer is -3
✓ 1 - 7 = -6
✓ 8 + 1 = 9
✕ 6 + 8 = 2 correct answer is 14

Total questions : 10
Correct questions : 5
Marks : 50.0%
Level : Medium
```

Figure 19 Test case 9

12.10 Test Case 10



```
Date : 2024-12-01
Time : 02:17

Session 1
Result sheet
✓ 1 + 3 = 4
✕ 0 - 5 = 4 correct answer is -5
✓ 6 + 6 = 12
✕ 8 - 10 = 2 correct answer is -2
✓ 2 - 5 = -3
✓ 2 + 10 = 12
✓ 3 + 6 = 9
✕ 0 - 2 = 3 correct answer is -2
✕ 10 + 9 = 22 correct answer is 19
✓ 7 + 2 = 9

Total questions : 10
Correct questions : 6
Marks : 60.0%
Level : Medium

Session 2
Result sheet
✕ 9 - 3 = 4 correct answer is 6
✕ 8 + 4 = 6 correct answer is 12
✓ 3 + 0 = 3
✓ 8 + 0 = 8
✓ 2 - 3 = -1
✕ 7 - 2 = 2 correct answer is 5
✕ 4 - 7 = 6 correct answer is -3
✓ 1 - 7 = -6
✓ 8 + 1 = 9
✕ 6 + 8 = 2 correct answer is 14

Total questions : 10
Correct questions : 5
Marks : 50.0%
Level : Medium
```

Figure 20 Test case 10

13. Conclusion

This project successfully creates a python-based console game that aims to generate questions according to the difficulty level. It facilitates real time feedback, comprehensive session summaries, session history storage in both .txt and .html formats with continuous game plays and error-handling mechanisms for further enhances. The project showcases a strong Python programming techniques, including different file types handling and interaction loops.

Overall , it highlights strong Python programming abilities and effective user input and file management.