



Python Programming with Files

GCSE Student Booster

Starter

Which of these programs would need file storage

1. A times table creator
2. A shopping list builder
3. Displaying a calendar for the current month
4. Asking a user for their login



Key Information

- 1) Remember this booster is here to **help you**. Please consider your behaviour in the chat.
- 2) If you are in a room with a teacher/group, please login to the meeting. This is so we can mark your attendance. This information goes into a **prize draw**.
- 3) Make sure the name on the meeting is the **SAME** as the name on your Isaac account. We can't mark you present if they don't match.



Answers - Which of these programs would need file storage?

1. A times table creator **No**
This program can operate with just input from the user and create times tables at run time
2. A shopping list builder **Yes**
Storing the current list and then being able to open it and add items to the list later is an essential feature
3. Displaying a calendar for the current month **No**
This can be auto created dynamically from user input
4. Asking a user for their login **Yes**
All authentication systems would want to read a file of authorised users and write to register new users.



What's the point of the Starter?

Looking at the problem and thinking about the Algorithm features that might appear helps with planning code.

Algorithm features

If

Selecting between two choices

For

Looping a fixed number of times
(count controlled)

File Read/Write

For data that is stored/retrieved

Array

Multiple items of the same data
type to be manipulated

While

Looping a variable number of times
(condition controlled)



Learning Outcomes

That you can confidently:

- Explain the main features in Python to read and write text files
- Analyse code containing file reading and writing and make changes to the code
- Create code solutions using file handling
- Identify some of the common misconceptions about file handling



Check In

2 minutes

On a scale of 1-10 (1-low, 10-high) what is your confidence

1. Understanding/reading code that uses **file read/write**
2. Writing code that uses **file read/write** loops

and

3. What do you want to learn from this session?



Writing files



Definition – Writing to files

Files on a file system can be opened by Python, new contents written to, and closed.

filename

Stored in the same folder as the code

mode

"w" - overwrites the contents
"a" - adds to the end of the file

```
2 file_object = open("twinkle.txt", mode="w")
3 file_object.write("Twinkle, twinkle, little star,\n")
4 file_object.write("How I wonder what you are!\n")
5 file_object.write("Up above the world so high,\n")
6 file_object.write("Like a diamond in the sky.\n")
7 file_object.close()
```

[Ref: kaac- Handling text files](#)



File writing – check-in

2 minutes

```
1 itemsFile = open("ZooAnimals.txt", "w")
2 itemsFile.write("Giraffe\n")
3 itemsFile.write("Moose\n")
4 itemsFile.write("Tiger\n")
5 itemsFile.close()
6
7 itemsFile = open("ZooAnimals.txt", "a")
8 itemsFile.write("Elephant\n")
9 itemsFile.close()
```

Which of these will be the output when this code is run?

A Giraffe Moose Tiger	B Elephant
C Giraffe Moose Tiger Elephant	D Moose Tiger Elephant Elephant

[Ref: basic Handling text files](#)



Handout 1 – File Writing Predict, Run, Investigate Activity

8 minutes

Open the link and complete the tasks on pages 1-2

Handout 1: File Writing Programming Activity

Look at the code that has been shared with you – [Code Link](#)

```
1 studentsFile = open("students.txt", "a")
2 name = input("Please enter your name: ")
3 age = input("and your age: ")
4 studentsFile.write(name+",")
5 studentsFile.write(age+"\n")
6 studentsFile.close()
7
8 file = open("students.txt", "r")
9 for student in file:
10     print(student)
11 file.close()
```



Handout 1 Investigate Answers

```
1 studentsFile = open("students.txt", "a")
2 name = input("Please enter your name: ")
3 age = input("and your age: ")
4 studentsFile.write(name+",")
5 studentsFile.write(age+"\n")
6 studentsFile.close()
7
8 file = open("students.txt", "r")
9 for student in file:
10     print(student)
11 file.close()
```

1. Run the program. Take a look at the text file, now change Line from "a" to "w". run the program 3 times. Look at the text file what happens. **It only stores the last entered student details. All previous are lost**

2. Comment out Line 6, run the program. what is different and why? **Files are only written to secondary storage when they are closed. By not closing it gets lost**



Handout 1 – Activity Modify, Make

12 minutes

Now complete the activities in the modify section (page 3 of the handout) to alter the code you have been given.

Then go on to the make activity to create a brand-new application using what you have learnt.



Handout 1 solution

```
Handout 1 final solution

def menu():
    print("----Welcome to the shopping list program----")
    print("1) Create New List")
    print("2) Add to your existing list")
    print("3) Add to a new list of your choice")
    print("4) Quit")
    return input()

while True:
    choice = menu()
    if choice == "1" or choice == "2":
        if choice == "1":
            file = open("shopping.txt", "w")
        else:
            file = open("shopping.txt", "a")
        while True:
            shop = input("Please enter your item: ")
            if shop == "stop":
                break
            file.write(shop + "\n")
        file.close()
    if choice == "3":
        f = input("What is the name of the file? ")
        file = open(f, "w")
        shop = input("Please enter your item: ")
        file.write(shop + "\n")
        file.close()
    if choice == "4":
        break
```



Reading files



Definition – Reading files

Files on a file system can be opened by Python and their contents read into a program

filename

Stored in the same folder as the code

mode

Open to read use "r"
Open to write use "w"
Open to append use "a"

```
2 file_object = open("bridge.txt", mode="r")
3 rhyme = file_object.read()
4 file_object.close()
5 print(rhyme)
```

read operations

read() - the whole file
read(5) - a number of chars.
readline() - a whole line

Ref: Isaac-Handling-text-files



Reading files with for loops

```
3 playerFile = open("player_file.txt", "r")
4 for player in playerFile:
5     print(player)
```

Each time round the loop the next line of the file is read into the **player** variable

The loop ends automatically after it has read to the end of the file

Ref: Isaac-Handling-text-files



File reading – check-in

2 minutes

```
1 itemsFile = open("animals.txt", "r")
2
3 for item in itemsFile:
4     print(item)
5
6 print(itemsFile.readline())
7
8 itemsFile.close()
```

animals.txt

1	Lion
2	Tiger
3	Wolf
4	Zebra

What will be the output when this code is run (A,B,C, or D)?

A	B
Lion	Lion
Tiger	Tiger
Wolf	Wolf
Zebra	Zebra
	Lion
C	D
Lion	Lion
Tiger	Tiger
Wolf	Wolf
Zebra	Zebra



Handout 2 – File Reading 9 minutes

Predict, Run, Investigate

Open the link and complete the tasks on pages 1-2

Handout 2: File Reading Programming Activity

Look at the code that has been shared with you – [Code link](#)

```
studentsFile = open("scores.txt", "a")
newScore = int(input("Enter your score: "))

highScores=[]
file = open("scores.txt", "r")
for score in file:
    highScores.append(score.strip("\n"))
file.close()
print(highScores)

if int(highScores[0])>newScore:
    print("Sorry, the highscore is still:", int(highScores[0]))
else:
    print("Congratulations! You have a new highscore")
```

File reading Activity

Investigate **Answers**

```
studentsFile = open("scores.txt", "r")
newScore = int(input("Enter your score: "))

highScores=[]
file = open("scores.txt", "r")
for score in file:
    highScores.append(score.strip("\n"))
file.close()
print(highScores)

if int(highScores[0])>newScore:
    print("Sorry, the highscore is still:", int(highScores[0]))
else:
    print("Congratulations! You have a new highscore")
```

1. Look at the values in the text files. Try typing in a higher score than those values. What happens? **It displays Congratulations, you have a new high score**
2. What if the value is lower than the high score? Or equal? **Same message as question 1. If it is lower you get the "sorry the highscore is still" message**

File reading Activity

Investigate **Answers**

```
studentsFile = open("scores.txt", "r")
newScore = int(input("Enter your score: "))

highScores=[]
file = open("scores.txt", "r")
for score in file:
    highScores.append(score.strip("\n"))
file.close()
print(highScores)

if int(highScores[0])>newScore:
    print("Sorry, the highscore is still:", int(highScores[0]))
else:
    print("Congratulations! You have a new highscore")
```

3. What does item.strip("\n") do on Line7? **removes the newline character from the end of the string**
4. On Line5 change "r" to "a". run the program - what happens? **There is a Python interpreter error because you cannot open an item for append and then read from it**

File Reading – Activity

Modify, Make

12 minutes

Complete the activities in the modify section to alter the code you have been given.

Then go on to the make activity to create a brand new application using what you have learnt.



Handout 2 solution

```

16: newScore = int(input("Enter your score: "))
17:
18: highScores=[]
19:
20: #file = open('scores.txt','r+')
21: #for score in file:
22: #    if score!=" " and score!="\n":
23: #        highScores.append(int(score.strip()))
24: #file.close()
25:
26: if int(highScores[0])>newScore:
27:     print("Sorry, the highscore is still!", int(highScores[0]))
28: else:
29:     print("Congratulations! You have a new highscore")
30:
31: #extension 1 = average
32: ave = sum(highScores)/len(highScores)
33: print("The average score is ", round(ave, 2))
34: if ave<newScore:
35:     print("You got above average")
36:
37: #challenge 2 = top 3
38: print("Top 3 scores")
39: for i in range(1,4):
40:     print("In position {}: is the score {}".format(i+1, str(h
41:
42: #challenge
43: highScores.append(newScore)
44: highScores = sorted(highScores)
45: for i in range(1, len(highScores)):
46:     print("In position {}: is the score {}".format(i+1, str(h
47:
48: #huge challenge
49: highScores.append(newScore)
50: highScores = sorted(highScores)

```



Files

Common misconceptions

1. You can read and write to a file at the same time.
You can only open it in one mode at a time
2. When you have read to the end of the file it will go back to the start. **You have to close/re-open or reset to start from the beginning again**
3. Opening for write adds to the file. **You must use append instead**
4. Writing a line of text to a file adds a new line automatically **It does not you need to add a \n**



Check in

2 minutes

1. What are the three ways to open a file?
2. What character do you use for each of these?
3. What does `read(5)` do?
4. What is the command to add items to the end of a file?
5. If you wanted to erase the contents of a file when you opened it what mode would you use?
6. What does `\n` do?





Learning with Isaac

Isaac provides all the knowledge and lots of practice to get confident with for and while loops in programming and exams

- [Concepts - Handling Text Files](#)
- Questions - examples [A day at the Zoo](#), [Shopping List](#), and more with [Question finder](#)
- Code snippets in Pseudocode/Python/c#



Isaac Gameboard practice

If you want more file handling practice, then try this gameboard.

You will need to sign in to **Isaac Computer Science** or register for a free account if not done already.

GCSE Booster - File handling

Is it true?	Programming Concepts / File handling	100%
File mode	Programming Concepts / File handling	100%
What's in the file?	Programming Concepts / File handling	100%
Appending the contents of a file	Programming Concepts / File handling	100%
End of the line	Programming Concepts / File handling	100%
The charity shop system	Programming Concepts / File handling	100%
CSV Downloads	Programming Concepts / File handling	100%
Adding students to the list	Programming Concepts / File handling	100%

nccse.io/isc-file



Check for more ISAAC boosters

Keep an eye out for more student booster events

Learning Aims

That you can confidently:

1. Explain the main features in Python to read and write text files
2. Analyse code containing file reading and writing and make changes to the code
3. Create code solutions using file handling
4. Identify some of the common misconceptions about file handling

Next Steps

1. Get an Isaac account, practice
2. Practice, practice, practice - programming (there are lots of platforms)
3. Use online Python resources (e.g. W3 schools - avoid ~~stackoverflow~~)

Next Steps

4. Past paper questions (search on your exam board website)
5. Ask your teacher to get an Isaac Teacher Account



"Everyone in this country should learn how to program because it teaches you how to think"

Steve Jobs



And...

Checkout all the other Student Boosters we have coming up



Search "Isaac Computer Science events"



Thank you


