Computational Thinking and Programming - 2

Binary Files

XII

Syllabus

Binary files: basic operations on a binary file: open using file open modes (rb, rb+, wb, wb+, ab, ab+), close a binary file, import pickle module, dump() and load() method, read, write/create, search, append and update operations in a binary file.

Introduction

A binary file is just a file that contains information in the same format in which the information is held in memory i.e. the file content that is returned to you is raw (with no translations or no encoding).
In a binary file there is no delimiter for a line. As no translations occur in a binary file, these files are faster for the program to read and write than text files.

The contents in a binary file are not saved in readable sequence of characters when opened in any text editor. Binary files can range from image files like JPEGs or GIFs, audio files like MP3s or binary document formats like Word or PDF. The data stored as a collection (list, dictionary) or object or a class will also come under binary file, where the data other than string will be converted to its byte equivalent ascii/ unicode character before its is being saved.

What is a Binary File in Python?

BINARY FILE
CONTENT IN THE PROGRAM
[[1, 'Harish'], [2, 'Taruna'], [3, 'Neha']]
CONTENT IN THE FILE
Ä ï0]î(]î(KåHarishîe]î(K[1] åTarunaîe]î(KåNehaîee.

A binary file stores the data in the same format as stored in the memory. It is not readable when we open the file in a text editor.

Binary files are made up of non-human readable characters and symbols, which require specific programs to access its contents.

Opening a file in binary mode

In Python, files are opened in text mode by default and so to open files in binary mode, when specifying a mode, add ‘b’ to it.

For example, **read mode** will become rb, **write mode** will become wb, **append mode** will become ab and **read-write mode** will become rb+,ab+

Binary Files

In Python, binary files can be created using pickle, json (Java Script Object Notation) or cpickle modules.

pickle is used for serializing and de-serialising Python collections/ objects in memory to a byte stream that can be stored on disk or sent over a network. Later on, this character stream can be retrieved and de-serialised back to python object.

pickle is specific to Python module, thus, cross-language compatibility is not guaranteed.

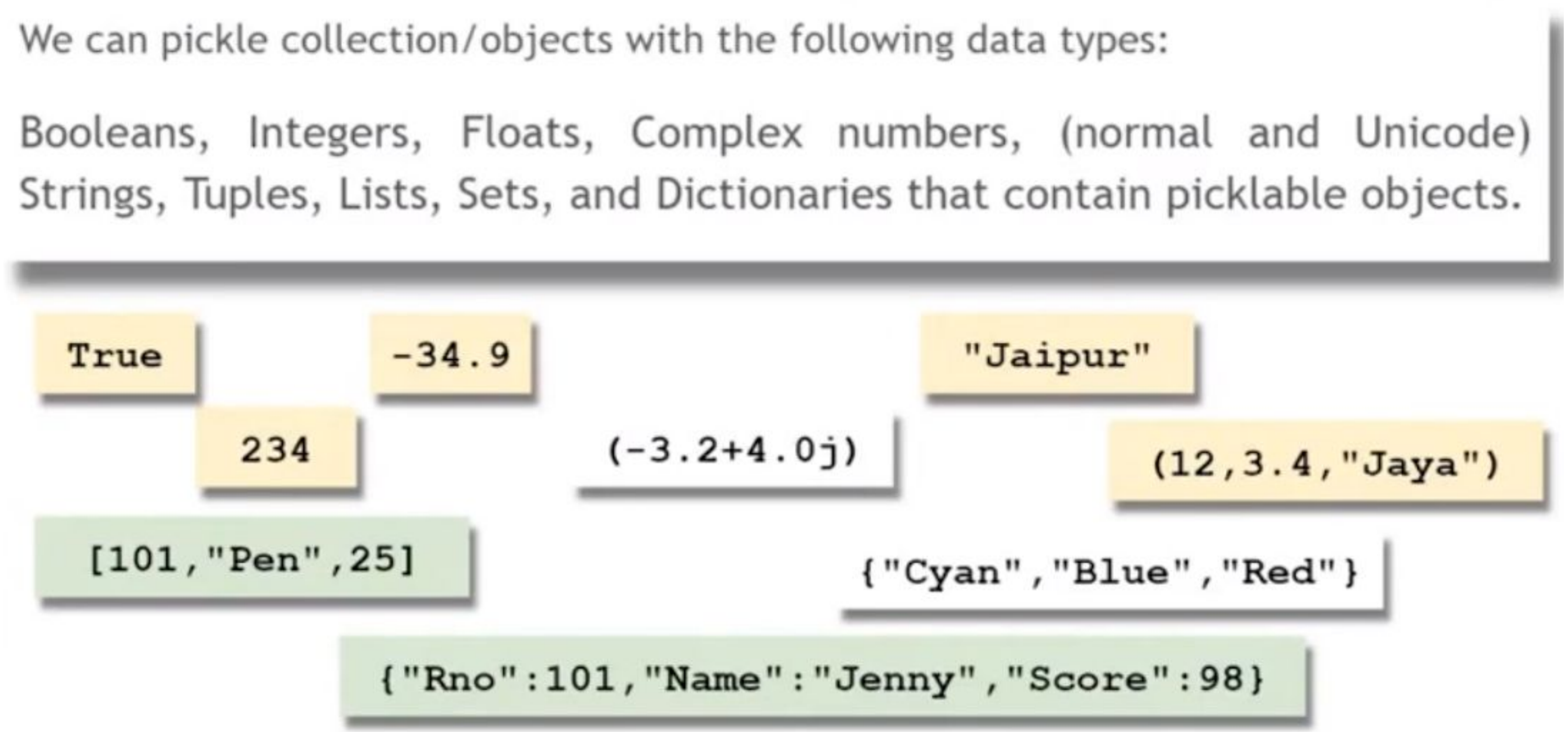
Relative and Absolute paths

Absolute File Path is specified by a drive label and/or folders along with leading forward slash (folder separated by/ in respective order). For example , /New.Project/File.Txt or c:/Myproject/File.Txt

An absolute path describes how to access a given file or folder/directory, starting from the root of the file system. A file path is also called pathname. It is the complete address of the file.

Relative File Path is specified by directly referring to the file/folder without a leading forward slash. For File.Txt a relative path is interpreted from the perspective of current working directory. If you use a relative file path from the wrong directory/ folder, then the path will refer to a different file than you intend, or it will refer to no file at all.

What can be pickled?



We can pickle collection/objects with the following data types:

Booleans, Integers, Floats, Complex numbers, (normal and Unicode) Strings, Tuples, Lists, Sets, and Dictionaries that contain picklable objects.

```
True
-34.9
"Jaipur"
234
(-3.2+4.0j)
(12,3.4,"Jaya")
[101,"Pen",25]
{"Cyan","Blue","Red"}
{"Rno":101,"Name":"Jenny","Score":98}
```

Pickle module

pickle.dump : It is used to write the pickled representation of the collection/ object to the file. Syntax :

```
pickle.dump(<Collection/Object>,<FileObject>)
```

pickle.load : It is used to read the pickled representation of collection/ object from the opened file object and return the reconstituted object hierarchy specified therein. This is equivalent to unpickle.

```
<Collection/Object> = pickle.load(<FileObject>)
```

Pickle and Unpickle - Code 1 (list)

CREATE FILE - create.py

```
import pickle
with open ("item.dat","wb") as F:
    pickle.dump([100,"Pencil"],F)
```

DISPLAY FILE - display.py

```
import pickle
with open("item.dat","rb") as F:
    print(pickle.load(F))
```

```
[100, 'Pencil']
```

item.dat

```
Äï]î(KdåPencilîe
```

```
[100, 'Pencil']
```

Pickle and Unpickle - Code 2 (dictionary)

CREATE FILE - create.py

```
import pickle
with open ("item.dat","wb") as F:
    pickle.dump({"INo":100,"IName":"Pencil"},F)
```

{'INo': 100, 'IName': 'Pencil'}

item.dat

```
Ä
ï-}î(åINoîKdåINameîåPencilîu.
```

{'INo': 100, 'IName': 'Pencil'}

DISPLAY FILE - display.py

```
import pickle
with open("item.dat","rb") as F:
    print(pickle.load(F))
```

Option 1

Load and Dump into the Binary File individually

Add content to a pickled file - Option 1

```
import pickle
def createfile():
    file=open("student.dat","wb")
    ans="y"
    while ans =="y":
        rno=int(input("Enter roll no : "))
        name=input("Enter name : ")
        ans=input("Want to enter more (y/n) - ")
        pickle.dump([rno,name],file)
    file.close()
```

OPTION 1

```
Enter roll no : 1
Enter name : Harish Gupta
Want to enter more (y/n) - y
Enter roll no : 2
Enter name : Sangeeta Sareen
Want to enter more (y/n) - y
Enter roll no : 3
Enter name : Nidhi Goel
Want to enter more (y/n) - n
```

Read contents from the binary file - Option 1

```
def readfile():
    try:
        file=open("student.dat","rb")
        while True:
            try:
                stu=pickle.load(file)
                print(stu)
            except:
                file.close()
                break
    except:
        print("File does not exist")
```

OUTPUT

```
[1, 'Harish Gupta']
[2, 'Sangeeta Sareen']
[3, 'Nidhi Goel']
```

Searching in a binary file - Option 1

```
def searchfile():
    rno=int(input("Enter the roll number you would like to search : "))
    try:
        file=open("student.dat","rb")
        while True:
            try:
                stu=pickle.load(file)
                if rno == stu[0]:
                    print(stu[0], " has name ",stu[1])
                    break
            except:
                print(rno, " not found in the file")
                file.close()
                break
    except:
      print("File does not exist")
```

Assume that the contents of the file are:

```
[1, 'Harish Gupta']
[2, 'Sangeeta Sareen']
[3, 'Nidhi Goel']
Enter the roll number you would like to search : 3
3  has name  Nidhi Goel
```

Appending in a binary file - Option 1

```
def appendfile():
    file=open("student.dat","ab")
    ans="y"
    while ans =="y":
        rno=int(input("Enter roll no : "
        name=input("Enter name : ")
        ans=input("Want to enter more (y
        pickle.dump([rno,name],file)
    file.close()
```

Contents of file:

```
[1, 'Harish Gupta']
[2, 'Sangeeta Sareen']
[3, 'Nidhi Goel']
```

Append File:

```
Enter roll no : 4
Enter name : Chitra Chatterjee
Want to enter more (y/n) - y
Enter roll no : 5
Enter name : Shambhavi Savarkar
Want to enter more (y/n) - n
```

New Contents:

```
[1, 'Harish Gupta']
[2, 'Sangeeta Sareen']
[3, 'Nidhi Goel']
[4, 'Chitra Chatterjee']
[5, 'Shambhavi Savarkar']
```

Option 2

Load and Dump all records into the Binary File at one go

Add content to a pickled file - Option 2

```
import pickle
def createfile():
    file=open("student.dat","wb")
    ans="y"
    L=[]
    while ans =="y":
        rno=int(input("Enter roll no : "))
        name=input("Enter name : ")
        L.append([rno,name])
        ans=input("Want to enter more (y/n) - ")
    pickle.dump(L,file)
    file.close()
```

```
Enter roll no : 1
Enter name : Harsh Gupta
Want to enter more (y/n) - y
Enter roll no : 2
Enter name : Sangeeta Sareen
Want to enter more (y/n) - y
Enter roll no : 3
Enter name : Nidhi Goel
Want to enter more (y/n) - n
```

Read contents from the binary file - Option 2

```
def readfile():
    try:
        file=open("student.dat","rb")
        L=pickle.load(file)
        for i in L:
            print("Roll Number : ", i[0], " Name : ",i[1])
        file.close()
    except:
       print("File does not exist")
```

```
Roll Number :  1  Name :  Harsh Gupta
Roll Number :  2  Name :  Sangeeta Sareen
Roll Number :  3  Name :  Nidhi Goel
```

Searching in a binary file - Option 2

```
def searchfile():
    rno=int(input("Enter the roll number you would like to search : "))
    try:
        file=open("student.dat","rb")
        L=pickle.load(file)
        for i in L:
            if rno == i[0]:
                print("Roll Number : ", i[0], " Name : ",i[1])
                break
        else:
            print("Roll Number", rno, " does not exist")
        file.close()
    except:
      print("File does not exist")
```

Searching in a binary file - Output (Option 2)

```
Enter a choice : (C/D/S/A/E) s
Enter the roll number you would like to search : 2
Roll Number :  2  Name :  Sangeeta Sareen
1. Create a file
2. Display from a file
3. Search in a file
4. Append in a file
5. Exit the program
Enter a choice : (C/D/S/A/E) s
Enter the roll number you would like to search : 5
Roll Number 5  does not exist
```

Appending into a binary file - Option 2

```
def appendfile():
    file=open("student.dat","rb+")
    ans="y"
    file.seek(0)
    L=pickle.load(file)
    while ans =="y":
        rno=int(input("Enter roll no : "))
        name=input("Enter name : ")
        L.append([rno,name])
        ans=input("Want to enter more (y/n) - ")
    file.seek(0)
    pickle.dump(L,file)
    file.close()
```

Appending into a binary file - Option 2

```
Enter a choice : (C/D/S/A/E) d
Roll Number :  1  Name :  Harish Gupta
Roll Number :  2  Name :  Sangeeta Sareen
Roll Number :  3  Name :  Nidhi Goel
1. Create a file
2. Display from a file
3. Search in a file
4. Append in a file
5. Exit the program
Enter a choice : (C/D/S/A/E) a
Enter roll no : 4
Enter name : Chitra Chatterjee
Want to enter more (y/n) - y
Enter roll no : 5
Enter name : Shambhavi Savarkar
Want to enter more (y/n) - n
```

```
Enter a choice : (C/D/S/A/E) d
Roll Number :  1  Name :  Harish Gupta
Roll Number :  2  Name :  Sangeeta Sareen
Roll Number :  3  Name :  Nidhi Goel
Roll Number :  4  Name :  Chitra Chatterjee
Roll Number :  5  Name :  Shambhavi Savarkar
```

Modifying the names in an already existing file - Option 2

```
def modifyfile():
    file=open("student.dat","rb+")
    ans="y"
    file.seek(0)
    L=pickle.load(file)
    for rec in L:
        print("Roll Number : ",rec[0], "  Name : ",rec[1])
        choice = input("Do you wish to change the name of the 
roll number ")
        if choice in 'Yy':
            rec[1]= input("Enter new name : ")
    file.seek(0)
    pickle.dump(L,file)
    file.close()
```

Create a file - containing Dictionary

```
import pickle
def createfile():
    file=open("student.dat","wb")
    ans="y"
    while ans =="y":
        d={}
        rno=int(input("Enter roll no :  "))
        name=input("Enter name :  ")
        ans=input("Want to enter more (y/n)")
        d={'RNo':rno,'name':name}
        pickle.dump(d,file)
    file.close()
```

Display file - containing Dictionary

```
def readfile():
    try:
        file=open("student.dat","rb")
        while True:
            try:
                stu=pickle.load(file)
                print("Roll Number : ",stu['RNo'], " Name : ",stu['name'])
            except:
                file.close()
                break
    except:
       print("File does not exist")
```

Search file - containing Dictionary

```python
def searchfile():
    rno=int(input("Enter the roll number you would like to search : "))
    try:
        file=open("student.dat","rb")
        while True:
            try:
                stu=pickle.load(file)
                if rno == stu['RNo']:
                    print(stu['RNo'], " has name ",stu['name'])
                    break
            except:
                print(rno, " not found in the file")
                file.close()
                break
    except:
      print("File does not exist")
```

Append a file - containing Dictionary

```
def appendfile():
    file=open("student.dat","ab")
    ans="y"
    while ans =="y":
        d={}
        rno=int(input("Enter roll no :  "))
        name=input("Enter name :  ")
        ans=input("Want to enter more (y/n) - ")
        d={'RNo':rno,'name':name}
        pickle.dump(d,file)
    file.close()
```

Menu driven - navigating through the options

```
while True:
    print("1. Create a file")
    print("2. Display from a file")
    print("3. Search in a file")
    print("4. Append in a file")
    print("5. Exit the program")
    choice = input("Enter a choice : (C/D/S/A/E) ")
    if choice in "cC":
        createfile()
    elif choice in "dD":
        readfile()
    elif choice in "sS":
        searchfile()
    elif choice in "aA":
        appendfile()
    else:
        break
```

File modes:

FILE MODES	OPERATION
rb	Read mode : Read contents from an existing binary file, raises an exception, FileNotFoundError, if the file does not exist.
rb+	Update mode: Opens the file both for reading and writing the binary file. The file pointer/cursor is placed at the beginning of the binary file.
wb	Write mode : Opens a write-only file in binary mode. In case it is existing, the contents are overwritten.
wb+	Write and read mode : Opens a file for both writing and reading in binary format. Overwrites the existing file if the file exists. If the file does not exist, it creates a new file for reading and writing.

File modes:

FILE MODES	OPERATION
ab	**Append mode :** It opens the file to write the content at the end of an existing binary file. The file cursor is at the end of the file in case the file exists. In case the file is not existing, a new file gets created.
ab+	Opens a file for both **appending** and **reading** in binary mode. The file cursor gets placed at the end of the existing file and can be moved to the beginning, if needed.

Practice Programs

1. Write a function, **createfile()** to create a binary file and **readfile()** which reads data from the file “employee.dat” containing data in dictionary format given below :-

   ```
   {'Emp Code' : 101 , 'Emp Name' : 'Rohan' , 'Designation' :
    'Clerk' , 'Salary' : 4500 }.
   ```

   Include a function **countrec()** to display and count number of employees whose salary is greater than 5000 and designation is `Clerk`.

2. Write a program to write data (as shown below) to file “books.dat” in List format as long as the user desires and then reads data from the same file. Display and count number of books with name ‘Python’. Store the data in following format in file Books.dat : (sample list format data given):

   ```
   ['101', 'Python', 'David', 400.0]
   ['102', 'MySQL', 'Pramiti', 390.0]
   ```

Practice Programs

3. Write a function CountVehicle() to read data of Vehilces from a data file "VEHICLE.DAT" and counts and displays the details of all Vehicles whose name is "Wagon R" and cost is greater than 600000. Also write the same to another file, called "Vehi.Dat".

```
VEHICLE={"Vehicle Code":11,"Vehicle Name":"Wagon R","Cost": 604000 }
```

4. Write a program to display the details of the Shop that has the minimum Sales in a month. The data is available in a binary file named "SHOPS.DAT", and the data is stored in given format.

```
SHOP ={"Code" :101,"Address": "23-Delhi","Sales":34000 }
```

Case Study

Assuming an organization maintains the following data about its employees in a binary file, "EMP.DAT":

1. EMPNo - Employee number of the employee
2. EMPName - Employee name
3. DeptNo - Department number
4. DeptName - Department name
5. Age - age of the employee
6. Salary - basic salary of the employee
7. Total Salary - total salary of the employee (after adding an allowance of 10% and making a deduction of 5 %)

Case Study

Write a Python program with the following options using user-defined functions:

1. To add a new employee in the existing file
2. To Search for a given employee using the EMPNo
3. To Display all the employees based on their department name(input by the user)
4. To Display all the employees based on their department number, input by the user
5. To count the number of employees in a particular department number
6. To transfer the details of all employees in a particular department into another file called "NEWEMP.DAT"
7. To modify the salary of an employee based on the EMPNo
8. To count the number of employees in each department

THANK YOU!

HAPPY LEARNING!!!