BINARY FILE HANDLING

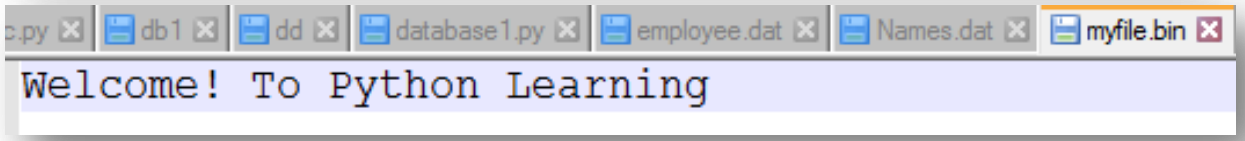
Handling the file in the way computer understands

Writing String to Binary file

- To store string in binary file, we must convert it to binary format either by prefixing the string with ‘b’ or using the encode() function.
- For e.g.

```
# Storing String in Binary File
msg="Welcome!"
f = open('myfile.bin','wb')
f.write(msg.encode())
f.write(b' To Python Learning')
f.close()
```

We can use ‘a’ in place of ‘w’ for append



Reading Binary file in String

```
# Reading String from Binary File
f=open('myfile.bin','rb')
msg=f.read()
print(msg)
print(msg.decode())
f.close()
```

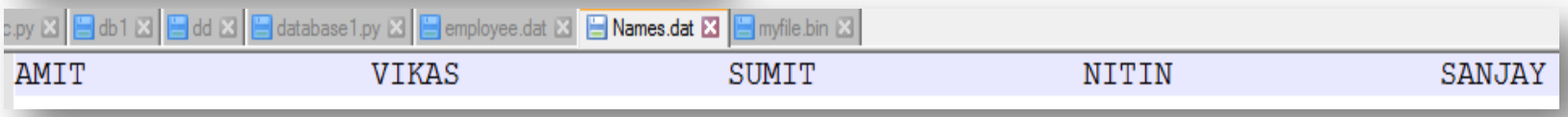
```
b'Welcome! To Python Learning'
Welcome! To Python Learning
```

We can observe, without decoding it will prefix text with 'b'

Program to create Binary file and store few records in it

```
# Program to Create Binary File and storing Few Names
size_of_rec = 20  #Each name will occupy 20 bytes
with open('Names.dat','wb') as f:
          ans='y'
          while ans.lower()=='y':
                    name = input("Enter Name :")
                    l = len(name)
                    name = name + (size_of_rec-l)*' '
                    # To add extra space with name to make it of length 20
                    name = name.encode()
                    f.write(name)
                    ans=input("Add More ?")
```

```
Enter Name :AMIT
Add More ?y
Enter Name :VIKAS
Add More ?y
Enter Name :SUMIT
Add More ?y
Enter Name :NITIN
Add More ?y
Enter Name :SANJAY
Add More ?n
```



Accessing record randomly from Binary File

```
# Program to Random Access any Name
# First Name will be at position 0, second will be at 20 and so on
size_of_rec = 20  #Each name will occupy 20 bytes
num = int(input("Enter Record Number :"))
with open('Names.dat','rb') as f:
        f.seek(size_of_rec*(num-1))    #Sending read pointer to desired position
        str = f.read(size_of_rec)
        if(len(str)==0):
                print("Incorrect Position ")
        else:
                print(str.decode())
```

```
Enter Record Number :3
SUMIT
>>>
== RESTART: C:\Users\vin\Deskt
Enter Record Number :10
Incorrect Position
```

```
Enter Name :AMIT
Add More ?y
Enter Name :VIKAS
Add More ?y
Enter Name :SUMIT
Add More ?y
Enter Name :NITIN
Add More ?y
Enter Name :SANJAY
Add More ?n
```

RECORD ORDER

Program to search for name in binary file and display record number

```
#Program to search for any name in file and display the record
#number that contains the name
import os
size_of_rec = 20
# Finding Size of File
size = os.path.getsize('Names.dat')
print("Size of file : ",size)

#Finding Number of Records
num_rec = int(size / size_of_rec)
print("Number of Records :",num_rec)

with open('Names.dat','rb') as f:
      n = input('Enter Name to Search ')
      n = n.encode()
      position = 0
      found = False
      for i in range(num_rec):
            f.seek(position)
            str = f.read(20)
            if n in str:
                  print('Found at Record # ',(i+1))
                  found=True
            position+=size_of_rec
      if not found:
            print('Name Not Found')
```

Program to search for name in binary file and display record number

```
#Program to search for any name in file and display the record
#number that contains the name
import os
size_of_rec = 20
# Finding Size of File
size = os.path.getsize('Names.dat')
print("Size of file : ",size)

#Finding Number of Records
num_rec = int(size / size_of_rec)
print("Number of Records :",num_rec)

with open('Names.dat','rb') as f:
        n = input('Enter Name to Search ')
        n = n.encode()
        position = 0
        found = False
        for i in range(num_rec):
                f.seek(position)
                str = f.read(20)
                if n in str:
                        print('Found at Record # ',(i+1))
                        found=True
                position+=size_of_rec
        if not found:
                print('Name Not Found')
```

```
Size of file :  100
Number of Records : 5
Enter Name to Search SUMIT
Found at Record #  3
>>>
=== RESTART: C:\Users\vin\Desktop\b
Size of file :  100
Number of Records : 5
Enter Name to Search RAJESH
Name Not Found
```

Program to update name in Binary File

```
# Program to Update Name in Binary File
import os
size_of_rec = 20
# Finding Size of File
size = os.path.getsize('Names.dat')
print("Size of file : ",size)

#Finding Number of Records
num_rec = int(size / size_of_rec)
print("Number of Records :",num_rec)

with open('Names.dat','r+b') as f:
        old_name = input('Enter Name :')
        old_name = old_name.encode()

        new_name = input('Enter New Name :')
        ln = len(new_name)
        new_name = new_name + (20-ln)* ' '
        new_name=new_name.encode()

        position = 0
        found = False
        for i in range(num_rec):
                f.seek(position)
                str = f.read(20)  #Read each name
                if old_name in str:
                        print('Updated Record No. ',(i+1))
                        found=True
                        f.seek(-20,1)  #sending cursor 20 bytes back for update
                        f.write(new_name)
                position+=size_of_rec
        if not found:
                print('Name Not Found')
```

```
Size of file :  100
Number of Records : 5
Enter Name :SUMIT
Enter New Name :SHIKHAR
Updated Record No.  3
```

```
Size of file :  100
Number of Records : 5
Enter Name :JAYANTILAL
Enter New Name :JETHALAL
Name Not Found
```

Program to update name in Binary File

```
# Program to Update Name in Binary File
import os
size_of_rec = 20
# Finding Size of File
size = os.path.getsize('Names.dat')
print("Size of file : ",size)

#Finding Number of Records
num_rec = int(size / size_of_rec)
print("Number of Records :",num_rec)

with open('Names.dat','r+b') as f:
            old_name = input('Enter Name :')
            old_name = old_name.encode()

            new_name = input('Enter New Name :')
            ln = len(new_name)
            new_name = new_name + (20-ln)* ' '
            new_name=new_name.encode()

            position = 0
            found = False
            for i in range(num_rec):
                        f.seek(position)
                        str = f.read(20)   #Read each name
                        if old_name in str:
                                    print('Updated Record No. ',(i+1))
                                    found=True
                                    f.seek(-20,1)   #sending cursor 20 bytes back for update
                                    f.write(new_name)
                        position+=size_of_rec
            if not found:
                        print('Name Not Found')
```

```
Size of file :  100
Number of Records : 5
Enter Name :SUMIT
Enter New Name :SHIKHAR
Updated Record No.  3
```

```
Size of file :  100
Number of Records : 5
Enter Name :JAYANTILAL
Enter New Name :JETHALAL
Name Not Found
```

py | db1 | dd | database1.py | employee.dat | Names.dat | myfile.bin

```
AMIT                VIKAS               SHIKHAR             NITIN               SANJAY
```

Program to delete name from binary file

```
# Program to Delete Name in Binary File
import os
size_of_rec = 20
# Finding Size of File
size = os.path.getsize('Names.dat')
print("Size of file : ",size)

#Finding Number of Records
num_rec = int(size / size_of_rec)
print("Number of Records :",num_rec)


f1 = open('Names.dat','rb')
f2 = open('Names2.dat','wb')

nm = input('Enter Name to be Deleted :')
l = len(nm)
nm= nm + (size_of_rec-l)*' '
nm=nm.encode()

position = 0
found = False
for i in range(num_rec):
            str = f1.read(size_of_rec)
            if(str!=nm):
                        f2.write(str)
print('Record Deleted ')
f1.close()
f2.close()
os.remove('Names.dat')
os.rename('Names2.dat','Names.dat')
```

```
Size of file :  100
Number of Records : 5
Enter Name to be Deleted :SHIKHAR
Record Deleted
```

Program to delete name from binary file

```
# Program to Delete Name in Binary File
import os
size_of_rec = 20
# Finding Size of File
size = os.path.getsize('Names.dat')
print("Size of file : ",size)

#Finding Number of Records
num_rec = int(size / size_of_rec)
print("Number of Records :",num_rec)


f1 = open('Names.dat','rb')
f2 = open('Names2.dat','wb')

nm = input('Enter Name to be Deleted :')
l = len(nm)
nm= nm + (size_of_rec-l)*' '
nm=nm.encode()

position = 0
found = False
for i in range(num_rec):
            str = f1.read(size_of_rec)
            if(str!=nm):
                        f2.write(str)
print('Record Deleted ')
f1.close()
f2.close()
os.remove('Names.dat')
os.rename('Names2.dat','Names.dat')
```

```
Size of file :  100
Number of Records : 5
Enter Name to be Deleted :SHIKHAR
Record Deleted
```

py | db1 | dd | database1.py | employee.dat | Names.dat | myfile.bin

```
AMIT                VIKAS               NITIN               SANJAY
```

Pickling – Storing employee details in binary file

```
import pickle
emp=[]
f = open('employee.dat','wb')
ans='y'
while ans=='y':
          eno = int(input("Enter Employee Number :"))
          name= input("Enter Employee Name ")
          salary=int(input("Enter Salary "))
          emp.append([eno,name,salary])
          ans=input("Add More record ")
pickle.dump(emp,f)
f.close()
```

```
Enter Employee Number :1
Enter Employee Name AMIT
Enter Salary 8000
Add More record y
Enter Employee Number :2
Enter Employee Name SUMIT
Enter Salary 9000
Add More record y
Enter Employee Number :3
Enter Employee Name NITIN
Enter Salary 45000
Add More record y
Enter Employee Number :4
Enter Employee Name DINKAR
Enter Salary 5500
Add More record n
```

Un-Pickling – Reading and Display Record

```
import pickle
emp=[]
f = open('employee.dat','rb')
ans='y'
while True:
        try:
                emp = pickle.load(f) # loading data in emp list
        except EOFError:
                break
for e in emp:
        print(e)
f.close()
```

```
[1, 'AMIT', 8000]
[2, 'SUMIT', 9000]
[3, 'NITIN', 45000]
[4, 'DINKAR', 5500]
```

Un-Pickling – Display Record (Formatted Output)

```
import pickle
emp=[]
f = open('employee.dat','rb')
ans='y'
while True:
        try:
                emp = pickle.load(f) # loading data in emp list
        except EOFError:
                break
print("%10s"%"EMP NO ","%20s"%"EMP NAME ","%10s"%"EMP SALARY")
print("**************************************************")
for e in emp:
        print("%10s"%e[0],"%20s"%e[1],"%10s"%e[2])
f.close()
```

```
   EMP NO               EMP NAME   EMP SALARY
**************************************************
         1                   AMIT         8000
         2                  SUMIT         9000
         3                  NITIN        45000
         4                 DINKAR         5500
```

Searching in Binary File

```
import pickle
emp=[]
f = open('employee.dat','rb')
ans='y'
print("********************* EMPLOYEE SEARCH FORM **********************")
en = int(input("Enter Employee Number to Search :"))
fount=False
while True:
        try:
                emp = pickle.load(f) # loading data in emp list
        except EOFError:
                break
print("%10s"%"EMP NO ","%20s"%"EMP NAME ","%10s"%"EMP SALARY")
print("*************************************************")
for e in emp:
        if (e[0]==en):
                print("%10s"%e[0],"%20s"%e[1],"%10s"%e[2])
                fount=True
                break
if fount==False:
        print("## SORRY EMPLOYEE NUMBER NOT FOUND ##")
f.close()
```

```
******************** EMPLOYEE SEARCH FORM *********************
Enter Employee Number to Search :3
   EMP NO            EMP NAME  EMP SALARY
***************************************************
         3               NITIN      45000
```

```
******************** EMPLOYEE SEARCH FORM *********************
Enter Employee Number to Search :5
   EMP NO            EMP NAME  EMP SALARY
***************************************************
## SORRY EMPLOYEE NUMBER NOT FOUND ##
```

Finding Number of Record in Binary File

```
import pickle
import os
emp=[]
f = open('employee.dat','rb')
emp = pickle.load(f)
l = len(emp)          # Count the number of emp object in file
# This part is used to display record, you can skip
while True:
          try:
                    emp = pickle.load(f) # loading data in emp list
          except EOFError:
                    break
print("%10s"%"EMP NO ","%20s"%"EMP NAME ","%10s"%"EMP SALARY")
print("*****************************************************")
for e in emp:
          print("%10s"%e[0],"%20s"%e[1],"%10s"%e[2])
# Display part Ends here
print("*****************************************************")
print("Total Records are :",l)
```

```
   EMP NO               EMP NAME   EMP SALARY
*****************************************************
         1                   AMIT         8000
         2                  SUMIT         9000
         3                  NITIN        45000
         4                 DINKAR         5500
*****************************************************
Total Records are : 4
```

Updating Employee Record

```
import pickle
emp=[]
f = open('employee.dat','rb')
emp = pickle.load(f) # loading data in emp list

print("## EMPLOYEE RECORDS ##")
print(emp)
print('-----------------------------------------------------------')
f.close()
f = open('employee.dat','wb')
found=False
en = int(input("Enter Employee Number to update :"))
for i in range(len(emp)):
          if emp[i][0]==en:
                    sl = int(input("Enter New Salary :"))
                    emp[i][2]=sl
                    found=True
                    print("## Record Updated ##")
if not found:
          print("## No Such Employee Number ##")
pickle.dump(emp,f)
f.close()
f=open('employee.dat','rb')
emp=pickle.load(f)
print("## EMPLOYEE RECORDS AFTER UPDATE ##")
print(emp)
print('-----------------------------------------------------------')
```

Updating Employee Record

```
import pickle
emp=[]
f = open('employee.dat','rb')
emp = pickle.load(f) # loading data in emp list

print("## EMPLOYEE RECORDS ##")
print(emp)
print('-------------------------------------------------------------')
f.close()
f = open('employee.dat','wb')
found=False
en = int(input("Enter Employee Number to update :"))
for i in range(len(emp)):
            if emp[i][0]==en:
                        sl = int(input("Enter New Salary :"))
                        emp[i][2]=sl
                        found=True
                        print("## Record Updated ##")
if not found:
            print("## No Such Employee Number ##")
pickle.dump(emp,f)
f.close()
f=open('employee.dat','rb')
emp=pickle.load(f)
print("## EMPLOYEE RECORDS AFTER UPDATE ##")
print(emp)
print('-------------------------------------------------------------')
```

```
## EMPLOYEE RECORDS ##
[[1, 'amit', 99999], [2, 'nitin', 8000], [3, 'hhh', 5500]]
-------------------------------------------------------------
Enter Employee Number to update :2
Enter New Salary :9000
## Record Updated ##
## EMPLOYEE RECORDS AFTER UPDATE ##
[[1, 'amit', 99999], [2, 'nitin', 9000], [3, 'hhh', 5500]]
-------------------------------------------------------------
```

Deleting Employee Record

```
import pickle
emp=[]
f = open('employee.dat','rb')
emp = pickle.load(f) # loading data in emp list

print("## EMPLOYEE RECORDS ##")
print(emp)
print('-----------------------------------------------------------')
f.close()
f = open('employee.dat','wb')
found=False
en = int(input("Enter Employee Number to Delete :"))
emp2=[]
for i in range(len(emp)):
          if emp[i][0]!=en:
                    emp2.append(emp[i])

pickle.dump(emp2,f)
f.close()
f=open('employee.dat','rb')
emp=pickle.load(f)
print("## EMPLOYEE RECORDS AFTER DELETE ##")
print(emp)
print('-----------------------------------------------------------')
```

Deleting Employee Record

```
import pickle
emp=[]
f = open('employee.dat','rb')
emp = pickle.load(f) # loading data in emp list

print("## EMPLOYEE RECORDS ##")
print(emp)
print('-----------------------------------------------------------')
f.close()
f = open('employee.dat','wb')
found=False
en = int(input("Enter Employee Number to Delete :"))
emp2=[]
for i in range(len(emp)):
            if emp[i][0]!=en:
                        emp2.append(emp[i])

pickle.dump(emp2,f)
f.close()
f=open('employee.dat','rb')
emp=pickle.load(f)
print("## EMPLOYEE RECORDS AFTER DELETE ##")
print(emp)
print('-----------------------------------------------------------')
```

```
## EMPLOYEE RECORDS ##
[[1, 'AMAN', 5000], [2, 'BIPIN', 9000], [3, 'CHANDU', 7800], [4, 'DINKAR', 9900]]
-----------------------------------------------------------
Enter Employee Number to Delete :3
## EMPLOYEE RECORDS AFTER DELETE ##
[[1, 'AMAN', 5000], [2, 'BIPIN', 9000], [4, 'DINKAR', 9900]]
-----------------------------------------------------------
```