# file_handling

August 21, 2024

### 0.0.1 How File I/O is done in most programming languages

- Open a file
- Read/Write data
- Close the file

### 0.0.2 Writing to a file

[24]:
```python
# case 1 - if the file is not present
f = open('sample.txt','w')
f.write('Hello world')
f.close()
# since file is close, this will not work
f.write('hello')
```

[32]:
```python
# write multiline strings
f = open('sample1.txt', 'w')
f.write('Hello world!!')
f.write('\nhow are you?')
f.close()
```

[34]:
```python
# case 2 - if the file is already present
f = open('sample.txt','w')
f.write('salman khan')
f.close()
```

[42]:
```python
# Problem with w mode
# introducing append mode(a)
f = open('sample1.txt', 'a')
f.write('I am fine')
f.close()
```

[46]:
```python
# write multiple lines in the existing directory
L=['Hey\n', 'How are you?\n', 'After a long time\n', 'It was nice meeting you!!
 ↪\n']
f = open('sample1.txt', 'w')
f.writelines(L)
f.close()
```

[5]:
```python
# reading from files
# -> using read()
f = open('sample1.txt', 'r')
s = f.read()
print(s)
f.close()
```

```
Hey
How are you?
After a long time
It was nice meeting you!!
```

[7]:
```python
# reading upto n chars
f = open('sample1.txt', 'r')
s = f.read(10)
print(s)
```

```
Hey
How ar
```

[13]:
```python
# readline() -> to read line by line
f = open('sample1.txt', 'r')
print(f.readline(), end= '')
print(f.readline(), end='')
f.close()
```

```
Hey
How are you?
```

[15]:
```python
# reading entire using readline
f = open('sample1.txt', 'r')
while True:
    data = f.readline()
    if data == '':
        break
    else:
        print(data,end='')

f.close()
```

```
Hey
How are you?
After a long time
It was nice meeting you!!
```

## 1 Using Context Manager (With)1.

It's a good idea to close a file after usage as it will free up the resources 2. If we dont close it, garbage collector would close i 3. t with keyword closes the file as soon as the usage is overver

[26]:
```
# with - shortcut for abpve code
# 'with' function automatically closes the file without the use of f.close()
with open('sample1.txt', 'w') as f:
    f.write('India is my country')
```

[34]:
```
# try f.read() now
with open('sample1.txt', 'r') as f:
    print(f.read())
```

```
India is my country
```

[42]:
```
# moving within a file -> 10 char then 10 char
with open('sample1.txt', 'r') as f:
    print(f.read(10))
    print(f.read(10))
```

```
India is m
y country
```

[58]:
```
# benefit? -> to load a big file in memory
big_L = ['hello world' for i in range(1000)]

with open('big_txt', 'w') as f:
    f.writelines(big_L)
```

[74]:
```
with open('big_txt', 'r') as f:
    chunk_size = 10
    while len(f.read(chunk_size)) > 0:
        print(f.read(chunk_size), end='***')
        f.read(chunk_size)
```

```
dhello wor***orldhello ***o worldhel***ello world***ldhello wo***worldhello***lo
worldhe***hello worl***rldhello w*** worldhell***llo worldh***dhello
wor***orldhello ***o worldhel***ello world***ldhello wo***worldhello***lo
worldhe***hello worl***rldhello w*** worldhell***llo worldh***dhello
wor***orldhello ***o worldhel***ello world***ldhello wo***worldhello***lo
worldhe***hello worl***rldhello w*** worldhell***llo worldh***dhello
wor***orldhello ***o worldhel***ello world***ldhello wo***worldhello***lo
worldhe***hello worl***rldhello w*** worldhell***llo worldh***dhello
wor***orldhello ***o worldhel***ello world***ldhello wo***worldhello***lo
worldhe***hello worl***rldhello w*** worldhell***llo worldh***dhello
wor***orldhello ***o worldhel***ello world***ldhello wo***worldhello***lo
worldhe***hello worl***rldhello w*** worldhell***llo worldh***dhello
wor***orldhello ***o worldhel***ello world***ldhello wo***worldhello***lo
```

```
worldhe***hello worl***rldhello w*** worldhell***llo worldh***dhello
wor***orldhello ***o worldhel***ello world***ldhello wo***worldhello***lo
worldhe***hello worl***rldhello w*** worldhell***llo worldh***dhello
wor***orldhello ***o worldhel***ello world***ldhello wo***worldhello***lo
worldhe***hello worl***rldhello w*** worldhell***llo worldh***dhello
wor***orldhello ***o worldhel***ello world***ldhello wo***worldhello***lo
worldhe***hello worl***rldhello w*** worldhell***llo worldh***dhello
wor***orldhello ***o worldhel***ello world***ldhello wo***worldhello***lo
worldhe***hello worl***rldhello w*** worldhell***llo worldh***dhello
wor***orldhello ***o worldhel***ello world***ldhello wo***worldhello***lo
worldhe***hello worl***rldhello w*** worldhell***llo worldh***dhello
wor***orldhello ***o worldhel***ello world***ldhello wo***worldhello***lo
worldhe***hello worl***rldhello w*** worldhell***llo worldh***dhello
wor***orldhello ***o worldhel***ello world***ldhello wo***worldhello***lo
worldhe***hello worl***rldhello w*** worldhell***llo worldh***dhello
wor***orldhello ***o worldhel***ello world***ldhello wo***worldhello***lo
worldhe***hello worl***rldhello w*** worldhell***llo worldh***dhello
wor***orldhello ***o worldhel***ello world***ldhello wo***worldhello***lo
worldhe***hello worl***rldhello w*** worldhell***llo worldh***dhello
wor***orldhello ***o worldhel***ello world***ldhello wo***worldhello***lo
worldhe***hello worl***rldhello w*** worldhell***llo worldh***dhello
wor***orldhello ***o worldhel***ello world***ldhello wo***worldhello***lo
worldhe***hello worl***rldhello w*** worldhell***llo worldh***dhello
wor***orldhello ***o worldhel***ello world***ldhello wo***worldhello***lo
worldhe***hello worl***rldhello w*** worldhell***llo worldh***dhello
wor***orldhello ***o worldhel***ello world***ldhello wo***worldhello***lo
worldhe***hello worl***rldhello w*** worldhell***llo worldh***dhello
wor***orldhello ***o worldhel***ello world***ldhello wo***worldhello***lo
worldhe***hello worl***rldhello w*** worldhell***llo worldh***dhello
wor***orldhello ***o worldhel***ello world***ldhello wo***worldhello***lo
worldhe***hello worl***rldhello w*** worldhell***llo worldh***dhello
wor***orldhello ***o worldhel***ello world***ldhello wo***worldhello***lo
worldhe***hello worl***rldhello w*** worldhell***llo worldh***dhello
wor***orldhello ***o worldhel***ello world***ldhello wo***worldhello***lo
worldhe***hello worl***rldhello w*** worldhell***llo worldh***dhello
wor***orldhello ***o worldhel***ello world***ldhello wo***worldhello***lo
worldhe***hello worl***rldhello w*** worldhell***llo worldh***dhello
wor***orldhello ***o worldhel***ello world***ldhello wo***worldhello***lo
worldhe***hello worl***rldhello w*** worldhell***llo worldh***dhello
wor***orldhello ***o worldhel***ello world***ldhello wo***worldhello***lo
worldhe***hello worl***rldhello w*** worldhell***llo worldh***dhello
wor***orldhello ***o worldhel***ello world***ldhello wo***worldhello***lo
worldhe***hello worl***rldhello w*** worldhell***llo worldh***dhello
wor***orldhello ***o worldhel***ello world***ldhello wo***worldhello***lo
worldhe***hello worl***rldhello w*** worldhell***llo worldh***dhello
wor***orldhello ***o worldhel***ello world***ldhello wo***worldhello***lo
worldhe***hello worl***rldhello w*** worldhell***llo worldh***dhello
wor***orldhello ***o worldhel***ello world***ldhello wo***worldhello***lo
```

```
worldhe***hello worl***rldhello w*** worldhell***llo worldh***dhello
wor***orldhello ***o worldhel***ello world***ldhello wo***worldhello***lo
worldhe***hello worl***rldhello w*** worldhell***llo worldh***dhello
wor***orldhello ***o worldhel***ello world***ldhello wo***worldhello***lo
worldhe***hello worl***rldhello w*** worldhell***llo worldh***dhello
wor***orldhello ***o worldhel***ello world***
```

[82]:
```python
# seek and tell function
with open('sample1.txt', 'r') as f:
    print(f.read(10))
    print(f.tell())
    f.seek(0)
    print(f.read(10))
    print(f.tell())
```

```
India is m
10
India is m
10
```

[5]:
```python
# seek during write
with open('sample1.txt', 'w') as f:
    f.write('helloz')
    f.seek(0)
    f.write('X')
```

### Problems with working in text mode

- can't work with binary files like images
- not good for other data types like int/float/list/tuples

[20]:
```python
# working with binary file, rb= read binary and wb= write binary
with open("C:\\Users\\Neelesh Dixit\\Desktop\\mountain.png", 'rb') as f:
    with open("C:\\Users\\Neelesh Dixit\\Desktop\\mountain_copy.png", 'wb') as␣
↪mf:
        mf.write(f.read())
```

[22]:
```python
# working with other data types
with open('sample.txt','w') as f:
  f.write(5)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[22], line 3
      1 # working with other data types
      2 with open('sample.txt','w') as f:
----> 3   f.write(5)
```

```
TypeError: write() argument must be str, not int
```

[24]:
```python
with open('sample.txt','w') as f:
  f.write('5')
```

[40]:
```python
# more complex data
d = {
    'name':'neelesh',
    'age' : 27,
    'gender' : 'Male'
}

with open('sample1.txt', 'r') as f:
    print(f.read())
    print(type(f.read()))
```

```
{'name': 'neelesh', 'age': 27, 'gender': 'Male'}
<class 'str'>
```

### 1.0.1 Serialization and Deserialization

- **Serialization** - process of converting python data types to JSON format
- **Deserialization** - process of converting JSON to python data types

**What is JSON?**

[18]:
```python
# serialization using json module
# list
import json

L = [1,2,3,4]

with open('demo.json', 'w') as f:
    json.dump(L,f)
```

[24]:
```python
# dict
d = {
    'name':'neelesh',
    'age' : 27,
    'gender' : 'Male'
}
with open('demo.json', 'w')as f:
    json.dump(d,f,indent=4)
```

[28]:
```python
# deserialization
import json
```

```
with open('demo.json', 'r') as f:
    d = json.load(f)
    print(d)
    print(type(d))
```

```
{'name': 'neelesh', 'age': 27, 'gender': 'Male'}
<class 'dict'>
```

[42]:
```
# serialize and deserialize tuple
import json

t = (1,2,3,4,5)
with open('demo.json', 'w') as f:
    json.dump(t, f)
```

[44]:
```
# serialize and deserialize a nested dict

d = {
    'student':'nitish',
     'marks':[23,14,34,45,56]
}

with open('demo.json','w') as f:
  json.dump(d,f)
```

### 1.0.2 Serializing and Deserializing custom objects

[53]:
```
class Person:
    def __init__(self,fname,lname,age,gender):
        self.fname = fname
        self.lname = lname
        self.age = age
        self.gender = gender

# format to printed in
# -> Nitish Singh age -> 33 gender -> male
```

[55]:
```
person = Person('Neelesh', 'Dixit',33,'male')
```

[57]:
```
# As a string
import json

def show_object(person):
    if isinstance(person, Person):
        return "{} {} age -> {} gender -> {}".format(person.fname,person.
 ↪lname,person.age,person.gender)
```

```
with open('demo.json', 'w')as f:
    json.dump(person,f,default=show_object)
```

[61]:
```
#As a dict
import json

def show_object(person):
    if isinstance(person, Person):
        return {'name':person.fname + '' + person.lname, 'age':person.
 ↪age,'gender':person.gender}
with open('demo.json', 'w')as f:
    json.dump(person,f,default=show_object,indent=4)
```

[63]:
```
# deserializing
import json

with open('demo.json','r') as f:
  d = json.load(f)
  print(d)
  print(type(d))
```

```
{'name': 'NeeleshDixit', 'age': 33, 'gender': 'male'}
<class 'dict'>
```

### 1.0.3 Pickling

`Pickling` is the process whereby a Python object hierarchy is converted into a byte stream, and `unpickling` is the inverse operation, whereby a byte stream (from a binary file or bytes-like object) is converted back into an object hierarchy.

[84]:
```
class Person:

  def __init__(self,name,age):
    self.name = name
    self.age = age

  def display_info(self):
    print('Hi my name is',self.name,'and I am ',self.age,'years old')
```

[86]:
```
p = Person('Neelesh', 28)
```

[92]:
```
#pickle dump
import pickle
with open('person.pkl', 'wb') as f:
    pickle.dump(p,f)
```

[94]:
```
#pickle load
import pickle
```

```
with open('person.pkl', 'rb') as f:
    pickle.load(f)
p.display_info()
```

```
Hi my name is Neelesh and I am  28 years old
```

```
[ ]:
```

```
[ ]:
```

```
[ ]:
```

```
[ ]:
```

```
[ ]:
```

```
[ ]:
```

```
[ ]:
```

```
[ ]:
```

```
[ ]:
```

```
[ ]:
```

```
[ ]:
```

```
[ ]:
```

```
[ ]:
```

```
[ ]:
```

```
[ ]:
```

```
[ ]:
```

```
[ ]:
```

```
[ ]:
```

```
[ ]:
```

```
[ ]:
```

```
[ ]:
```

```
[ ]:
```

```
[ ]:
```