27/1/22

Assignment No. 2

1. Define class, object and attributes. Demonstrate the concept of functions and modifiers with code.

Ans **class** - A user-defined prototype for an object that defines a set of attributes that characterize any object of the class.

Object - A unique instance of a data structure that is defined by its class. An object comprises both datamembers and methods

Attributes - Assigning values to named elements of an object. These elements are called attributes

example: >>> blank.x = 3.0

>>> blank.y = 4.0

Functions and modifiers

```
class Time:
        time = Time()
        time.hour = 11
        time.minute = 59
        time.seconds = 30
        time1 = Time()
        time1.hour = 2
        time1.minutes = 30
        time1.seconds = 40
    done = add_time(time, time1)
    print_time(done)
def add_time(t1, t2):
    sum = Time()
    sum.hour = t1.hour + t2.hour
    sum.minute = t1.minute + t2.minute
    sum.seconds = t1.seconds + t2.seconds
    return sum
```

output:

13:89:70

using modifiers the function can written in such a way that proper time or valid time is displayed.

```
def add_time(t1, t2):
    sum = Time()
    sum.hour = t1.hour + t2.hour
    sum.minute = t1.minute + t2.minute
    sum.seconds = t1.seconds + t2.seconds

    if sum.seconds >= 60: sum.seconds -= 60
        sum.minute += 1

    if sum.minute >= 60: sum.minute -= 60
        sum.hour += 1

    return sum

done = add_time(time, time1)
print_time(done)
```

output:

14:30:10

---

9. Ans Explain __init__ and __str__ methods with an example?

__init__ method

The init method is a special method that gets invoked when an object is instantiated.

Ex:

```
def class Time:
    def __init__(self, hour=0, minute=0, second=0):
        self.hour = hour
        self.minute = minute
        self.second = second
```

```
time= Time()
time.print_time()
```

output:

```
00:00:00
```

The --str-- method:

This method returns a string representation of an object. when we print an object, python invokes the str method.

Ex:

```
class Time:
    def--str--(self):
    return '%.2d:%.2d:%.2d' %(self.hour,
    self.minute, self.second)
time= Time(9,54)
print(time)
```

output:

```
09:54:00
```

---

3. Explain operator overloading and polymorphism with examples

Ans operator overloading:

Python allows programmers to redefine the meaning of operators when they operate on class objects. This feature is called operator overloading.

+	__add__
-	__sub__
*	__mul__
/	__truediv__
**	__pow__
%	__mod__
>>	__rshift__
&	__and__

Example:

```
class Time:
    def time_to_int(time):
    minutes = time.hour * 60 + time.minute
    seconds = minutes * 60 + time.seconds
    return seconds
    def int_to_time(seconds):
        time = Time()
    minutes, time.second = divmod(seconds, 60)
    time.hour, time.minute = divmod(minute,
                                    60)
    return time

    def add_time(t1, t2):
    seconds = time_to_int(t1) + time_to_int(t2)
    return int_to_time(seconds)

    t1 = Time(9, 45)
    t2 = Time(1, 35)
    print(t1 + t2)
```

output

```
11:20:00
```

polymorphism:

- polymorphism refers to having several different forms. It enables the programmers to assign a different meaning or usage to a variable, function, or an object in different contexts
- when polymorphism is applied to a function or method depending on the given parameters, a particular form of the function can be selected for execution. method overriding in one way of implementing polymorphism

Example:

```
class India():
    def capital(self):
        print("New Delhi")

    def language(self):
        print(" Hindi and English")

class USA():
    def capital(self):
        print("Washington, D.C ")

    def language(self):
        print("English")

obj_ind = India()
obj_usa = USA()

for country in (obj_ind, obj_usa):
    country.capital()
    country.language()
```

Output:

```
New Delhi
Hindi and English
Washington, D.C
English
```

---

4. Illustrate the concept of inheritance with example code

Ans **Inheritance**

The process of creating the new class by extending

existing class is called inheritance or the process of inheriting the features of base case is called as inheritance

Example:

```
class Person(object):

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

    def getName(self):
        return self.name

    def isEmployee(self):
        return False

class Employee(person):

    def isEmployee(self):
        return True

emp = Person("Anu")
print(emp.getName(), emp.isEmployee())

emp = Employee("Abi")
print(emp.getNAME(), emp.isEmployee())
```

Output:

```
Anu  False
Abi  True
```

5. Explain the process of downloading files from the web with request module and also saving downloaded files to harddrive with example?

Ans Downloading a web page with the requests module

- The requests module lets you easily download files from the web without having to worry about complicated issues such as network errors, connection problems, and data compression.
- The requests.get() function takes a string of a URL to download. By calling type() one requests.get()'s return value, you can see that it returns a Response object. which contains the response that the web server gave for your request

Example:

```
>>> import requests
>>> res = requests.get('http://www.gutenberg.org/cache/epub/11121/pg1112.txt')
>>> type(res)
<class 'requests.models.Response'>
>>> res.status_code == requests.codes.ok
True
>>> len(res.text)
178981
>>> print(res.text[:250])
The project Gutenberg Ebook of Romeo and Juliet, by William Shakespare
```

- The URL goes to a text webpage for the entire play of Romeo and Juliet, provided by project Gutenberg. You can tell that the request for this webpage succeeded by checking status_code attribute of the Response object
- If it is equal to value of requests.codes.ok, the request is succeeded, the downloaded web page is stored as a string in the Response object's text variable, and we can display how many characters we want by

Specifying index

Saving downloaded files to the hard drive

We can save the web pages to a file on your hard drive with the standard open() function and write() method. We should open the file in write binary mode.

Example:
```
>>> import requests
>>> res = requests.get('http://www.gutenberg.org/
        cache/epub/1112/pg1112.txt')
>>> res.raise_for_status()
>>> playFile = open('Romeo And Juliet.txt', 'wb')
>>> for chunk in
        res.iter_content(100000):
            playFile.write(chunk)
100000
78981
>>> playFile.close()
```

- The iter_content() method returns chunks of the content on each iteration through the loop. Each chunk is of the bytes data type, and you get to specify how many bytes each chunk will contain.
- The write() method returns the number of bytes written to the file.

process for downloading & saving.

1. Call requests.get() to download the file
2. Call open() with 'wb' to create a new file in write binary mode.
3. Loop over the Response objects iter_content() method
4. Call write() on each iteration to write the content to the file
5. Call close() to close the file

6. Describe the getText() function used for getting full text from a .docx file with example code.

Ans If we want only the text, not the styling information in the word document, you can use getText() function. It accepts a filename of a .docx file and returns a single string value of its text.

Example:

```
#! python 3
import docx
def getText(filename):
    doc = docx.Document(filename)
    fullText = []
    for para in doc.paragraphs:
        fullText.append(para.text)
    return '\n'.join(fullText)
```

- Save this file as readDocx.py
- The getText() function opens the word document, loops over all the paragraph objects in the paragraphs list, and then appends their text to the list in fullText. After the loop, the strings in fullText are joined together with newline characters.
- The readDocx.py program can be imported like any other module.

```
>>> import readDocx
>>> print(readDocx.getText('demo.docx'))
Document Title
A plain paragraph with some bold and
some italic Heading, level 1
Intense quote
first item in unordered
list
```

- You can also adjust getText() to modify the string before

returning it

Example:

To indent each paragraph,

fullText.append(' ' + para.text)

To add a double space in between paragraphs

return '\n\n'.join(full_Text)

7. What is JSON? Explain the json module of python with sample code?

Ans JSon

- JavaScript Object Notation is a popular way to format data as a single human-readable string. JSON is the native way that JavaScript programs write their data structure and usually resembles with python's pprint() function would produce.

The json module

- Python's json module handles all the details of translating between a string with JSON data and python values for the json.loads() and json.dumps() function. JSON can't store every kind of python value. It can contain values of on the following datatypes: strings, integers, floats, Booleans, lists, dictionaries, and None Type.

Reading JSON with the loads() function (load string)

- To translate a string containing JSON data into a python value, pass it to the json.loads() function

Example:

```
>>> stringOfJsonData = '{"name": "Zophie", "isCat":
    true, "miceCaught": 0, "felineIQ": null}'
>>> import json
>>> jsonDataAsPythonValue =
    json.loads(stringOfJsonData)
>>> jsonDataAsPythonValue
```

{'isCat': True, 'miceCaught': 0, 'name': 'Zophie', 'felineIQ': None}

Writing JSON with the dumps() function (dump string)

- The json.dumps() function will translate a python value into a string of JSON-formatted data

Example:

```
>>> pythonValue = {'isCat': True, 'miceCaught': 0, 'name':
                   'Zophie', 'felineIQ': None}
>>> import json
>>> stringOfJsonData = json.dumps(pythonValue)
>>> stringOfJsonData
'{"isCat": true, "felineIQ": null, "miceCaught": 0, "name":
                   "Zophie"}'
```

---

8. Discuss the creation, encryption and decryption of a pdf?

Ans Creating PDF's

- PyPDF2's counterpart to PdfFileReader objects is PdfFileWriter objects, which can create new PDF files. But PyPDF2 cannot write arbitrary text to a PDF like python can do with plaintext files. Instead, PyPDF2's PDF writing capabilities are limited to copying pages from other PDF's, rotating pages, overlaying pages and encrypting files.
- PyPDF2 doesn't allow you to directly edit a PDF. Instead you have to create a new PDF and then copy content over from an existing document.

The general approach

1. Open one or more existing PDFs into PdfFileReader objects
2. Create a new PdfFileWriter object
3. Copy pages from the PdfFileReader objects into the PdfFileWriter object

9. Finally, use the pdfFileWriter object to write the output PDF

Encrypting PDFs

A pdfFileWriter object can also add encryption to a PDF document.

Example:

```
>>> import PyPDF2
>>> pdfFile = open('meetingminutes.pdf', 'rb')
>>> pdfReader = PyPDF2.PdfFileReader(pdfFile)
>>> pdfWriter = PyPDF2.PdfFileWriter()
>>> for pageNum in range(PdfReader.numPages):
        pdfWriter.addPage(pdfReader.getPage(PageNum))

>>> pdfWriter.encrypt('swordfish')
>>> resultPdf = open('encryptedminutes.pdf', 'wb')
>>> pdfWriter.write(resultPdf)
>>> resultPdf.close()
```

Decrypting PDFs

Some PDF documents have an encryption feature that will keep them from being read until whoever is opening the document provides a password

Example:

```
>>> import PyPDF2
>>> pdfReader = PyPDF.PdfFileReader(open('encrypted.pdf',
                                        'rb'))

>>> pdfReader.isEncrypted
True
>>> pdfReader.decrypt('swordfish')
1
>>> pageobj = pdfReader.getPage(0)
```