**1s**

class Document:

def \_\_init\_\_(self,title,author):

self.\_title = title

self.\_author =author

def display\_info(self):

return self.\_author,self.\_title

disply = Document("Turab","pasha")

disply.display\_info()

**2:**

class Document:

def \_\_init\_\_(self, title, author):

self.\_title = title

self.\_author = author

def display\_info(self):

return self.\_author, self.\_title

class Book(Document):

def \_\_init\_\_(self, title, author, genre, pages):

super().\_\_init\_\_(title, author)

self.\_genre = genre

self.\_pages = pages

def display\_info(self):

return f"{self.\_genre}, {self.\_pages}, {super().display\_info()}"

disply = Book("Turab", "Pasha", "Fiction", 223)

print(disply.display\_info())

**3:**

class Document:

def \_\_init\_\_(self, title, author):

self.\_title = title

self.\_author = author

def display\_info(self):

return f"Title is {self.\_title} and author is {self.\_author}"

class Book(Document):

def \_\_init\_\_(self, title, author, genre, pages):

super().\_\_init\_\_(title, author)

self.\_genre = genre

self.\_pages = pages

def display\_info(self):

return f"{self.\_genre} book with {self.\_pages} pages, {super().display\_info()}"

class Article(Book):

def \_\_init\_\_(self, title, author, genre, pages, journal, DOI):

super().\_\_init\_\_(title, author, genre, pages)

self.\_journal = journal

self.\_DOI = DOI

def display\_info(self):

return f"{self.\_genre} article with {self.\_pages} pages, published in {self.\_journal} with DOI {self.\_DOI}, {super().display\_info()}"

article = Article("pasha ", "Turab", "Science", 10, "Science ", "123-456-789")

print(article.display\_info())

**4:**

class Document:

def \_\_init\_\_(self, title, author):

self.\_title = title

self.\_author = author

def display\_info(self):

return f"Title is {self.\_title} and author is {self.\_author}"

class Book(Document):

def \_\_init\_\_(self, title, author, genre=None, pages=None):

super().\_\_init\_\_(title, author)

self.\_genre = genre

self.\_pages = pages

if genre is None and pages is None:

self.\_genre = "bio"

self.\_pages = 0

def display\_info(self):

return f"{self.\_genre} book with {self.\_pages} pages, {super().display\_info()}"

class Article(Book):

def \_\_init\_\_(self, title, author, genre, pages, journal=None, DOI= None):

super().\_\_init\_\_(title, author, genre, pages)

self.\_journal = journal

self.\_DOI = DOI

if journal is None and DOI is None:

self.\_journal = "journal"

self.\_DOI = "DOI"

def display\_info(self):

return f"{self.\_genre} article with {self.\_pages} pages, published in {self.\_journal} with DOI {self.\_DOI}, {super().display\_info()}"

book = Book("Turab", "pasha")

articleWithFull = Article("pasha ", "Turab", "Science", 10, "Science ", "123-456-789")

articleWithOutFull = Article("pasha ", "Turab", "Science", 10)

print(book.display\_info())

print(articleWithFull.display\_info())

print(articleWithOutFull.display\_info())

**5:**

import csv

import os.path

class Document:

def \_\_init\_\_(self, title, author):

self.\_title = title

self.\_author = author

def display\_info(self):

return f"Title is {self.\_title} and author is {self.\_author}"

class Book(Document):

def \_\_init\_\_(self, title, author, genre=None, pages=None):

super().\_\_init\_\_(title, author)

self.\_genre = genre

self.\_pages = pages

if genre is None and pages is None:

self.\_genre = "bio"

self.\_pages = 0

def display\_info(self):

return f"{self.\_genre} book with {self.\_pages} pages, {super().display\_info()}"

def AppendData(self):

fieldnames = ["title", 'author', 'genre', 'pages']

data = [self.\_title, self.\_author, self.\_genre, self.\_pages]

file\_exists = os.path.isfile("books.csv")

with open("books.csv", 'a', newline='') as file:

writeData = csv.writer(file)

if not file\_exists:

writeData.writerow(fieldnames)

writeData.writerow(data)

@staticmethod

def readData():

data = []

with open("books.csv", 'r') as file:

csv\_reader = csv.reader(file)

for row in csv\_reader:

data.append(row)

return data

class Article(Document):

def \_\_init\_\_(self, title, author, genre, pages, journal=None, DOI=None):

super().\_\_init\_\_(title, author)

self.\_genre = genre

self.\_pages = pages

self.\_journal = journal if journal else "journal"

self.\_DOI = DOI if DOI else "DOI"

def display\_info(self):

return f"{self.\_genre} article with {self.\_pages} pages, published in {self.\_journal} with DOI {self.\_DOI}, {super().display\_info()}"

def AppendData(self):

fieldnames = ["title", 'author', 'genre', 'pages', 'journal', 'DOI']

data = [self.\_title, self.\_author, self.\_genre, self.\_pages, self.\_journal, self.\_DOI]

file\_exists = os.path.isfile("articles.csv")

with open("articles.csv", 'a', newline='') as file:

writeData = csv.writer(file)

if not file\_exists:

writeData.writerow(fieldnames)

writeData.writerow(data)

@staticmethod

def readData():

data = []

with open("articles.csv", 'r') as file:

csv\_reader = csv.reader(file)

for row in csv\_reader:

data.append(row)

return data

# Example Usage

book = Book("Turab", "Pasha", "Science", 150)

book.AppendData()

articleWithFull = Article("Pasha", "Turab", "Science", 10, "Science Journal", "123-456-789")

articleWithFull.AppendData()

articleWithOutFull = Article("Pasha", "Turab", "Science", 10)

articleWithOutFull.AppendData()

print(book.display\_info())

print(articleWithFull.display\_info())

print(articleWithOutFull.display\_info())

print("Books data:")

print(Book.readData())

print("Articles data:")

print(Article.readData())