FSE598 前沿计算技术

模块 2 数据与数据处理
单元 5 文件操作与大数据处理
第 1 讲 文件操作和案例研究

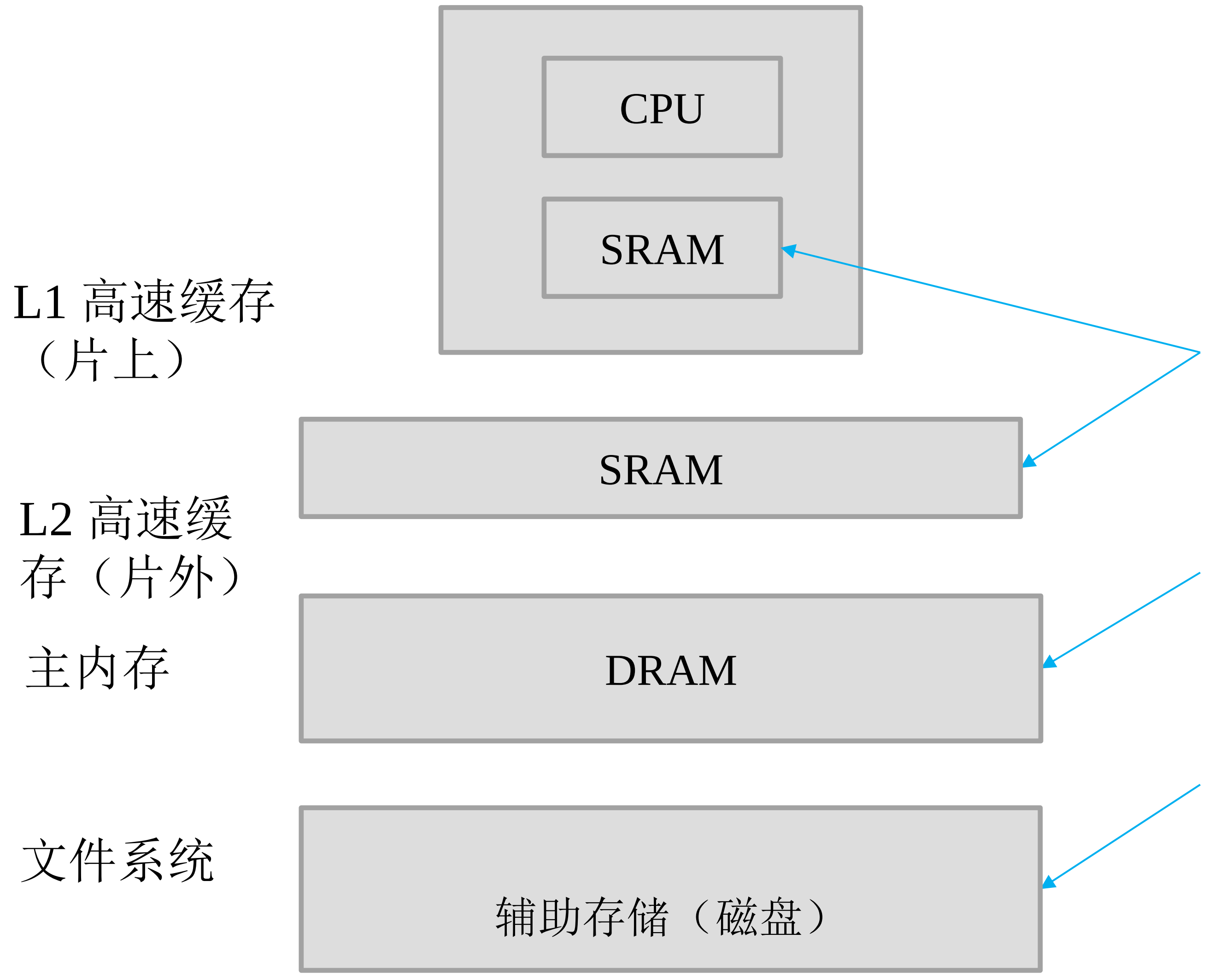
讲座的英文版内容基于本书：
 Y. Chen 《编程语言入门：C、C++、Scheme、Prolog、C# 和 Python 编程》(Introduction to Programming Languages: Programming in C, C++, Scheme, Prolog, C#, and Python)，第 6 版，Kendall Hunt Publishing Company，2019 年。
https://www.public.asu.edu/~ychen10/book/IntroPl.html

本讲大纲

学习

- ❑ 计算机存储和内存技术
- ❑ 文件操作
- ❑ 案例研究能够融合各方面的信息
  - 案例研究:文件操作
  - 每个类在一个单独的文件中
  - 构成链表的对象的容器
  - 继承、层次结构和多态性

存储和内存技术



创建一个具备以下特点的存储系统

- 容量大
- 速度快
- 价格便宜

❑ 静态 RAM (SRAM)

- 0.5ns – 2.5ns，每 GB $200 – $500

❑ 动态 RAM (DRAM)

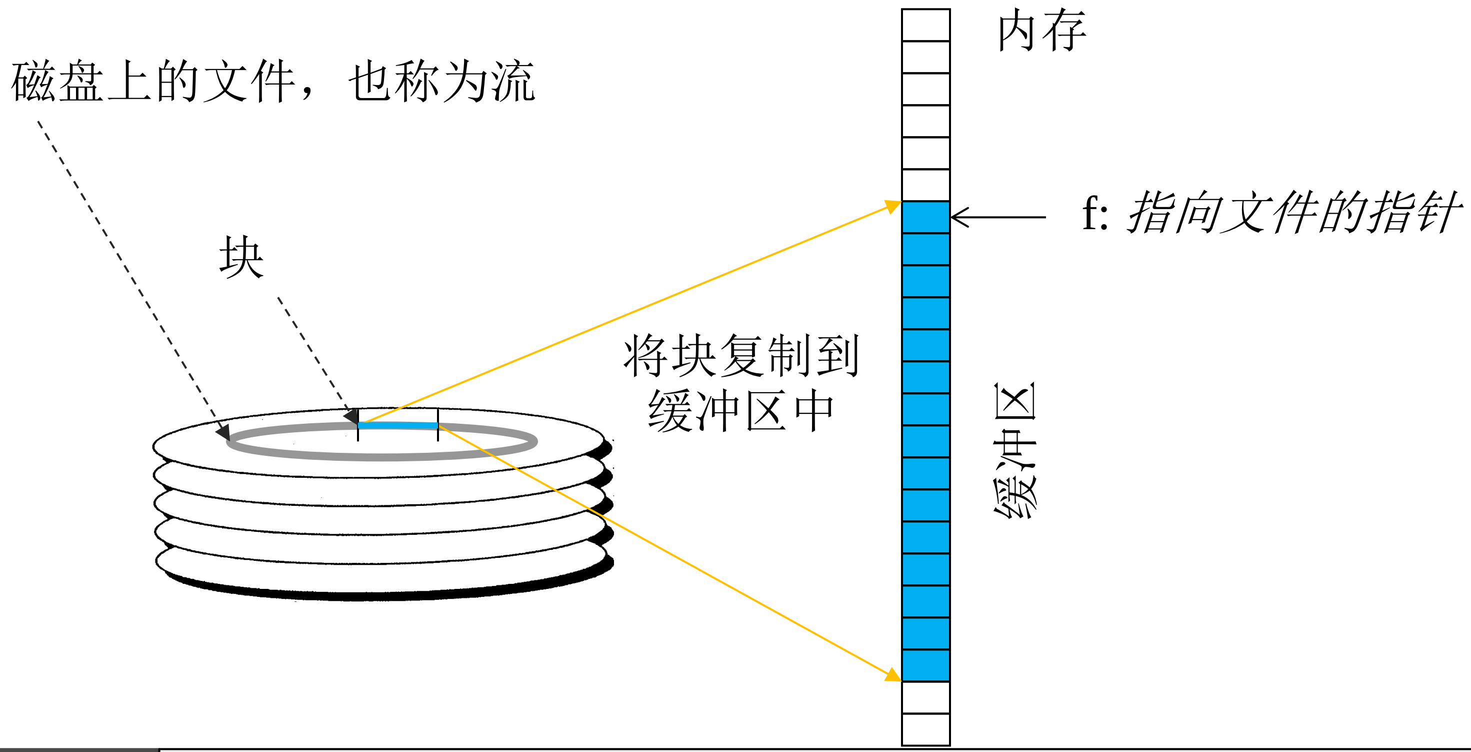
- 50ns – 70ns，每 GB $20 – $75

❑ 磁盘

- 5ms – 20ms，每 GB $0.20 – $2

文件操作概念：磁盘访问比内存访问慢百万倍

1. 说明一个指向 FILE 类型的指针 *f*；
2. **打开文件进行读取：**创建一个可以容纳一大个字节块（例如，1024 字节）的缓冲区；
3. 将文件的第一个块复制到缓冲区中；
4. 程序使用指针按顺序读取缓冲区中的数据；
5. 当指针向下移动到缓冲区的末尾时，下一个块被自动复制到缓冲区中，并且指针被重置到缓冲区的开头
6. 关闭文件



Python 文件操作

- Python 为程序员提供了许多文件操作以打开、读取、写入、管理和关闭文件。

运算	说明	示例
Open (……)	打开一个文件并将文件位置链接到文件名，并创建一个缓冲区用于存储数据块以加快文件操作。允许多个选项，r：读取；w：写入；a：附加；+：更新（读取并写入）；　x：创建新文件；t：以文本模式打开；b：以二进制模式打开。	fileBuf = open("myFile. txt",'wt') fileBuf = open("myFile",'rb')
Close ()	关闭打开的文件并释放文件缓冲区。如果文件已经关闭，则无效。	fileBuf.close ()
Read () Read (n)	读取到文件末尾。 读取文件中最多 n 个字符。如果 n 为负数，则读取到文件末尾 (EOF)。	1 = fileBuf.read () 1 = fileBuf.read (8)

Python 文件操作（续）

readable()	如果文件能被读取，则返回 True。	r = fileBuf.readable()
readline () readline (n)	从文件中读取一行。 如果指定 n，则最多读取 n 个字节。	1 = fileBuf.readline() 1 = fileBuf.readline(8)

readlines ()
 	
readlines (n)
	从文件中读取行列表直到文件末尾。
	
	如果指定 n，则最多读取 n 个行。
		ls = fileBuf.readline()
		ls = fileBuf.readline(8)

seek(ref,
offset)
	通过添加参考从 (start - 0, current - 1, or end - 2) 位置的偏移字节来更改文件光标的位置。	pos = fileBuf.seek(0,8)
		pos = fileBuf.seek(1,8)

tell ()	返回当前光标位置。	Pos = fileBuf.tell()
write(s)	将字符串 s 写入文件并返回写入的字符数。	File Name. write("Hello")
writable()	如果文件可以写入，则返回 True。	w = fileBuf.writable()

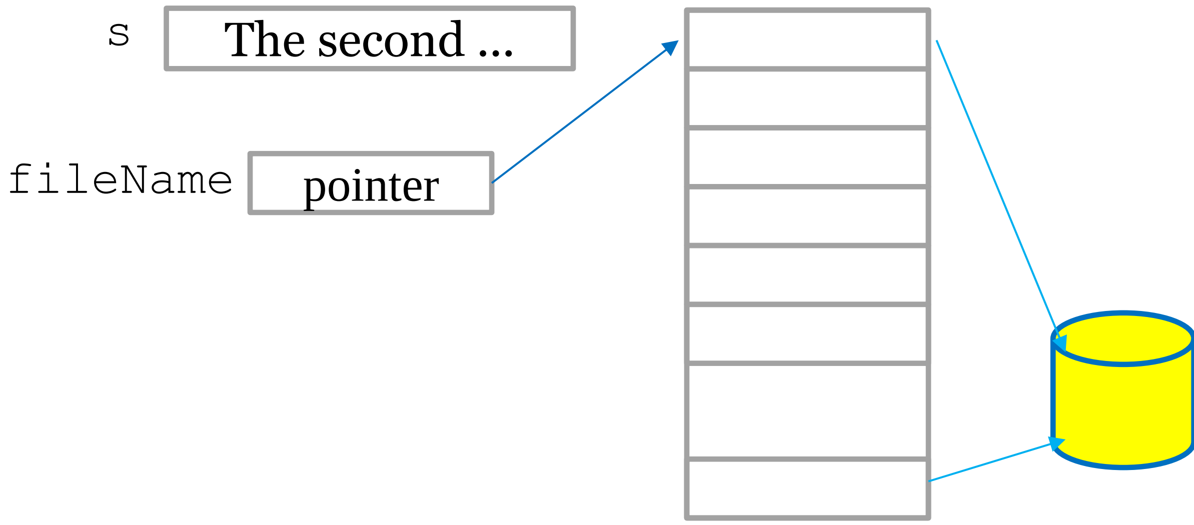
Python 文件操作（续）

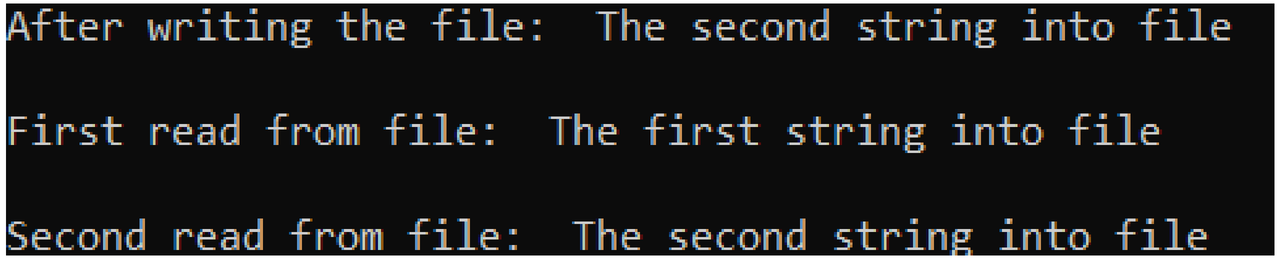
Writelines()	将行列表写入文件。	1st = ["Hello\n", "World"] fileBuf.writelines(1st)
Flush()	清空文件缓冲区以删除先前操作留下的字符。	fileBuf.flush()

Python 文件操作示例

下文的示例将几项文件操作结合在一起

```
fileName = "MyFile.txt"
fileBuf = None
fileBuf = open(fileName, 'w’)   / or ‘a’
s = "The first string into file\n"
fileBuf.write(s)
s = "The second string into file\n"
fileBuf.write(s)
print("After writing the file: ", s)
fileBuf.close()
fileBuf = open(fileName, 'r')
r = fileBuf.readline()
print("First read from file: ", r)
r = fileBuf.readline()
print("Second read from file: ", r)
fileBuf.close()
```





```
After writing the file:  The second string into file

First read from file:  The first string into file

Second read from file:  The second string into file
```

将异常用于文件操作

```
fileName = "MyFile.txt"
fileBuf = None
try:
    fileBuf = open(fileName, 'w')
    s = "The first string into file\n"
    fileBuf.write(s)
    s = "The second string into file\n"
    fileBuf.write(s)
    print("After writing the file: ", s)
    fileBuf.close()
    fileBuf = open(fileName, 'r')
    r = fileBuf.readline()
    print("First read from file: ", r)
    r = fileBuf.readline()
    print("Second read from file: ", r)
    fileBuf.close()
```

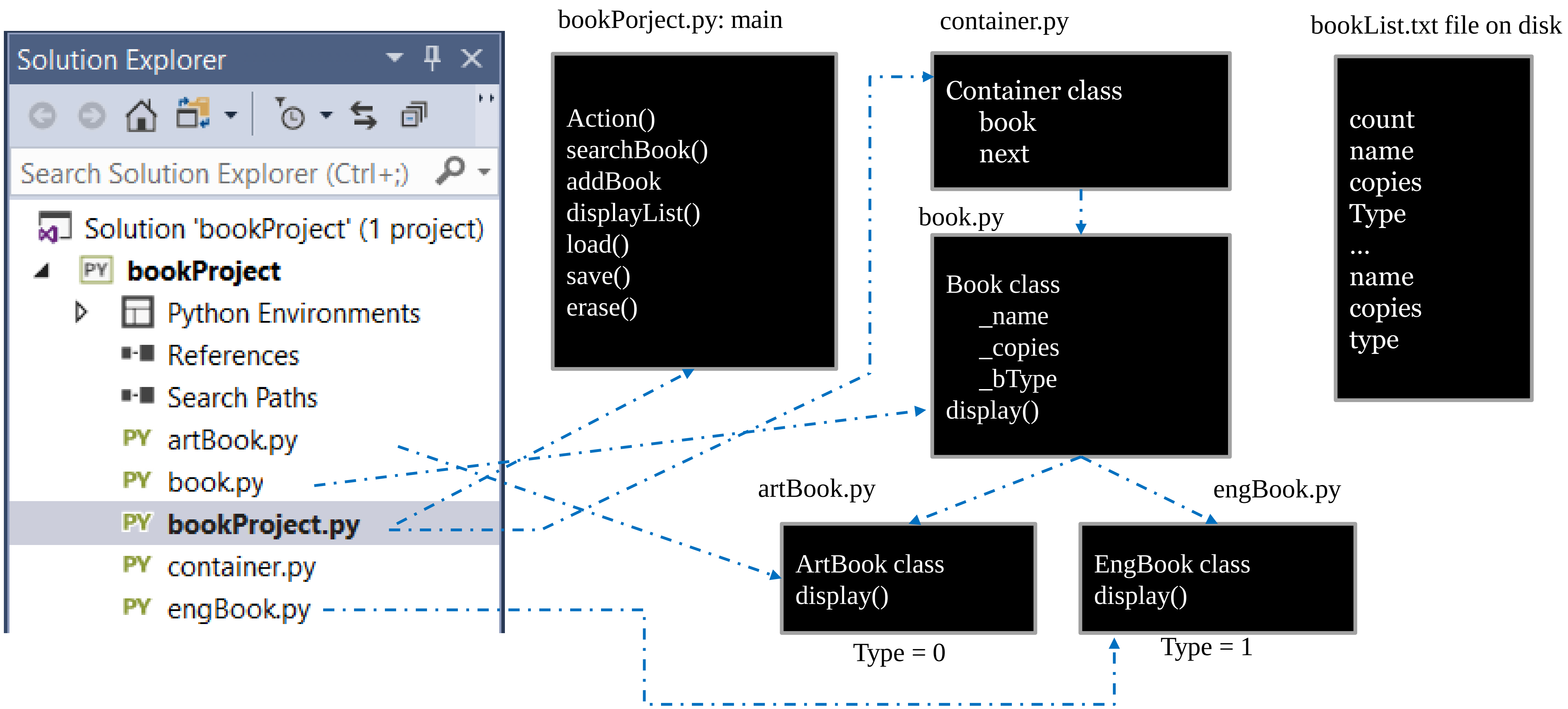
```
except IOError:
    msg = ("Unable to create file on disk.")
    print("In except: Unable to create file
on disk")
    fileBuf.close()
    exit
finally:
    print("In finally: Make sure the file is
closed!")
    fileBuf.close()
```

案例研究：全部结合起来

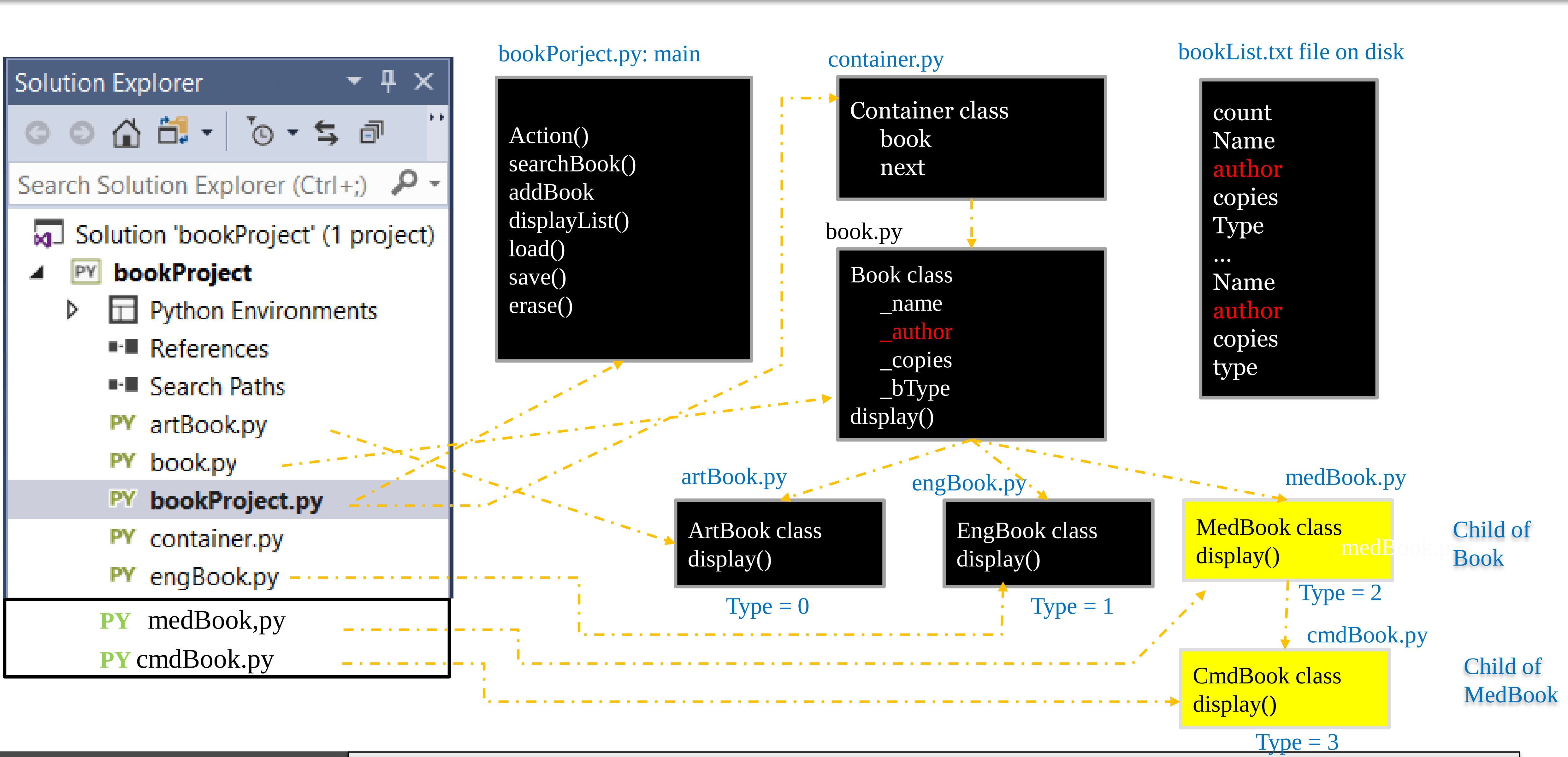
使用文件操作和继承层次结构

类和代码文件组织

每个类置于不同的代码文件中

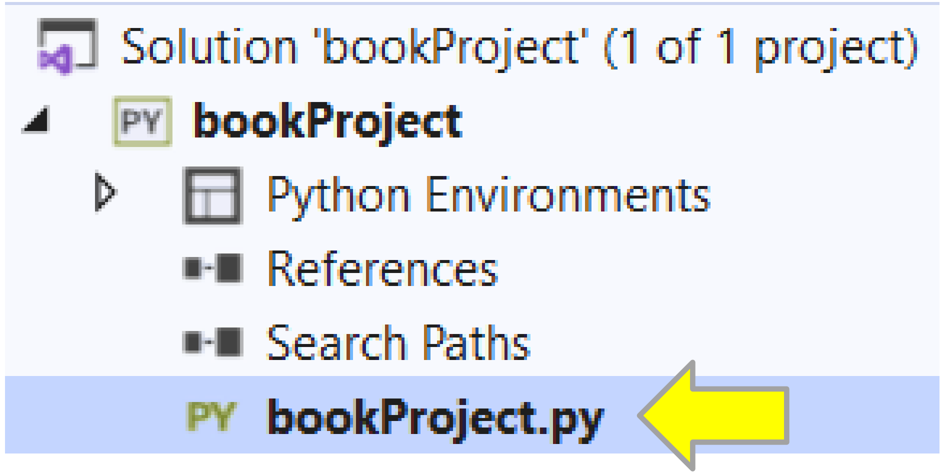


练习：可以添加更多的类


Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution 'bookProject' (1 project)
bookProject
Python Environments
References
Search Paths
artBook.py
book.py
bookProject.py
container.py
engBook.py
medBook,py
cmdBook.py
bookPorject.py: main
Action()
searchBook()
addBook
displayList()
load()
save()
erase()
container.py
Container class
book
next
book.py
Book class
_name
_author
_copies
_bType
display()
bookList.txt file on disk
count
Name
author
copies
Type
...
Name
author
copies
type
artBook.py
ArtBook class
display()
Type = 0
engBook.py
EngBook class
display()
Type = 1
medBook.py
MedBook class
display()
Type = 2
Child of Book
cmdBook.py
CmdBook class
display()
Type = 3
Child of MedBook

定义 bookProject.py 中全局函数的主程序 – 第 1 页

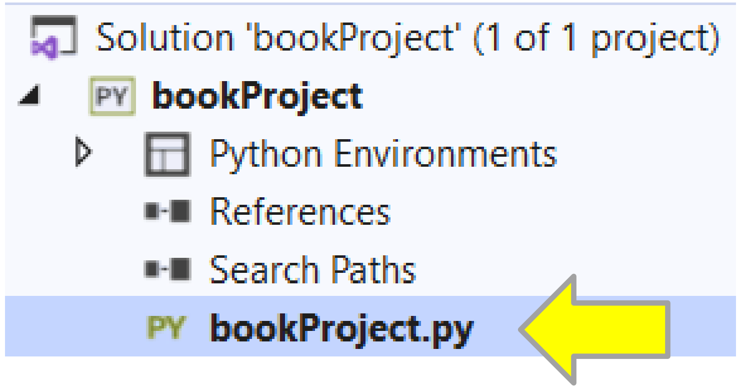
```
# Book Project consisting of multiple files:
# bookProject.py, Book.py, ArtBook.py, EngBook.py, and container.py
from book import *
from container import *
from artBook import *
from engBook import *
#Initializes bookList
bookList = None
#Searches for the given book name
def searchBook(name):
    #Grants access to global variable
    global bookList
    #Creates a bookList copy
    temp = bookList;
```



```
# Loops until all books are checked.
# returns book name or None
while (temp != None):
       if (temp.Book.getName() == name):
           return temp.Book;
       temp = temp.next;
    return None;
```

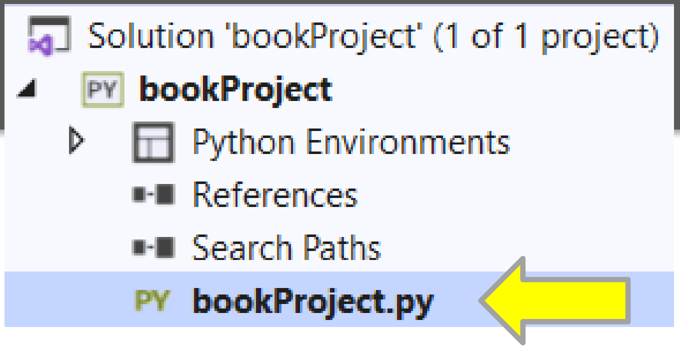
bookProject.py – 第 2 页

```
#Changes the number of copies of a given book
def changeNumberOfCopies(b, count):
    b.changeNoOfCopies(count)
#Adds a book to the bookList
def addBook(name_input ,copies_input ,type):
    #Gives access to global bookList
    global bookList
    #Creates a copy of bookList
    temp = bookList
    #Changes type to an actual Category
    if (type == Category.artBook):
        b = ArtBook(name_input, copies_input, type)
    else:
        b = EngBook(name_input, copies_input, type)
```



```
# Checks if there are any books in
# the bookList and if not makes a
# new Container and sets bookList
# to that container
if (bookList == None):
    bookList = Container()
    bookList.Book = b
    bookList.next = None
    return;
```

bookProject.py – 第 3 页


Solution 'bookProject' (1 of 1 project)
bookProject
Python Environments
References
Search Paths
bookProject.py

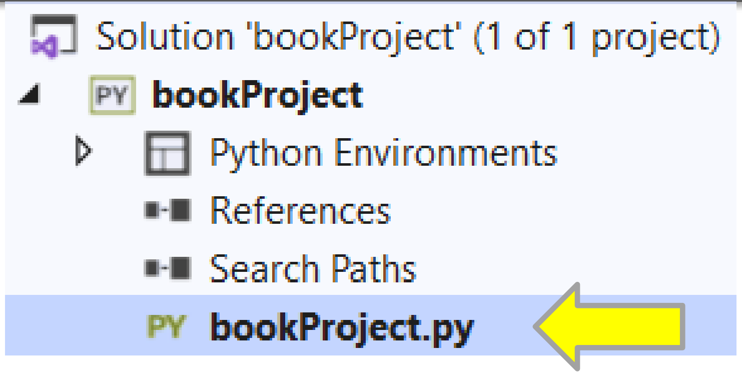
The code Creates a new Container object and adds to linked list end

```
#Creates a new Container
con = Container()
con.Book = b
con.next = None
#Finds last container
while(temp.next != None):
    temp = temp.next
#Sets the next Container to the new Container
temp.next = con
```

在链表结尾
添加新节点

```
#Displays the bookList
def displayList():
    #Gives access to bookList
    global bookList
    #Creates copy of bookList
    temp = bookList
    #Displays each book's information
    while(temp!=None):
        temp.Book.displayBook()
        temp = temp.next
```

bookProject.py – 第 4 页: 将数据保存到磁盘

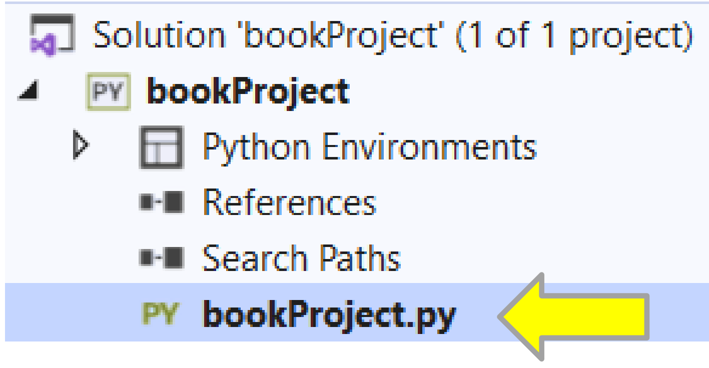


```
#Saves the bookList
def save(fileName):
    global bookList #Gives access to global bookList
    temp = bookList #Creates a temp variable to bookList
    count = 0
    #Counts the number of books
    while(temp != None):
        count +=1
        temp = temp.next
    #Opens the file for writing
    f = open(fileName,"w")
    temp = bookList #Resets temp to be a copy of bookList
    f.write(str(count)+"\n"); # Save count first
```

```
 #Loops through and adds the elements
 #of each book
 #Save (1) name, (2) Copies, (3)Type
 while (temp != None):
       f.write(temp.Book.getName()+"\n");
       f.write(str(temp.Book.getCopies())+"\n");
       f.write(str(temp.Book.getBookType())
+"\n");
       temp = temp.next
 f.close() #Closes file
```

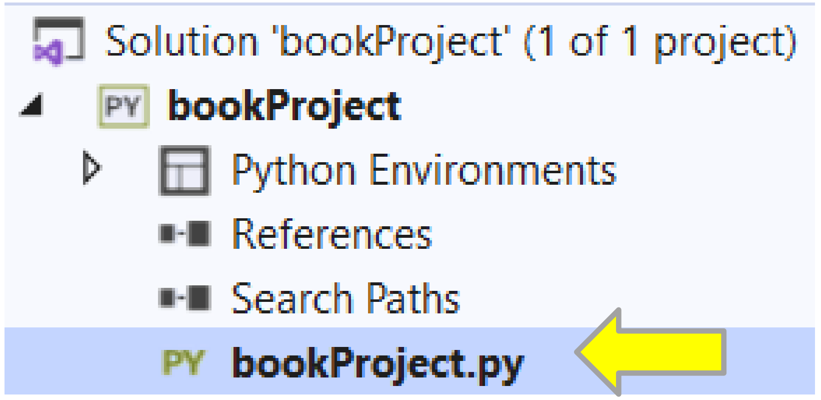
bookProject.py – 第 5 页: 从磁盘中加载数据

```
#Loads existing books
def load(filename):
    global bookList #Gives access to global bookList
    temp = bookList #Creates a temp variable to bookList
    count = 0
    #Tries to open the file, if there is no file it does nothing
    try:
        #Reads file
        f = open(filename,"r") # Opens for read
        #Gets the number of books
        count = f.readline()
        count = int(count)
        #Sets index to 0
        index = 0
        #Adds each book
```



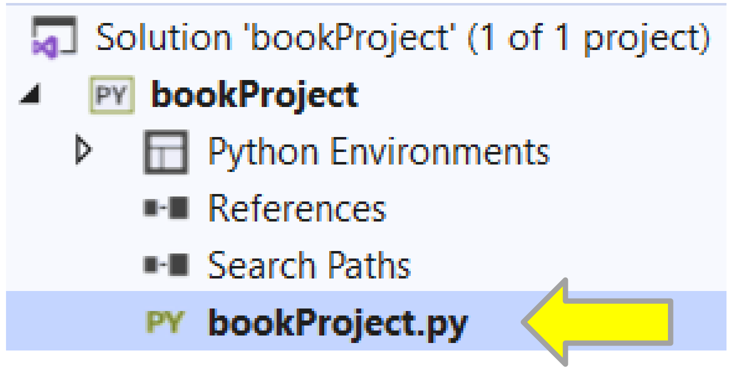
bookProject.py – 第 6 页

```
        #Adds each book
while (index<count):
    #Creates a new container
    con = Container()
    #Resets temp to be a copy of bookList
    temp = bookList
    #Gets name, copies, and type
    name = f.readline()
    name = name[0:(len(name)-1)]
    copies = int(f.readline())
    t = f.readline()
    t = t[12:(len(t)-1)]
    #Sets the string version of type to a bookType type
```


Solution 'bookProject' (1 of 1 project)
bookProject
Python Environments
References
Search Paths
bookProject.py

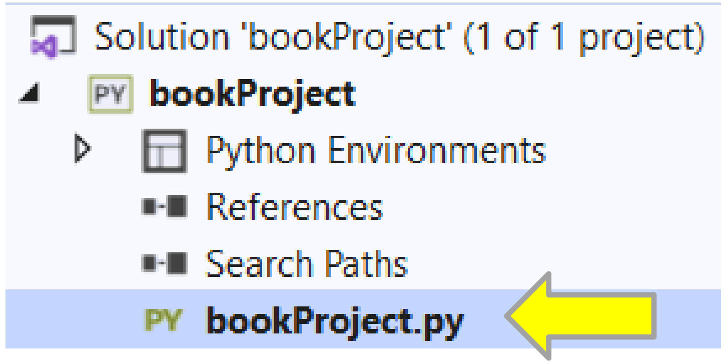
bookProject.py – 第 7 页

```
#Sets the string version of type to a bookType type
     if(t=="engBook"):
         type = Category.engBook
     else:
         type = Category.artBook
     #Creates a book based on its bookType type
     #Sets it to the Container's book
     if(type == Category.artBook):
         con.Book = ArtBook(name,copies, type)
     else:
         con.Book = engBook(name,copies,type)
     #Sets Container's next to nothing
```



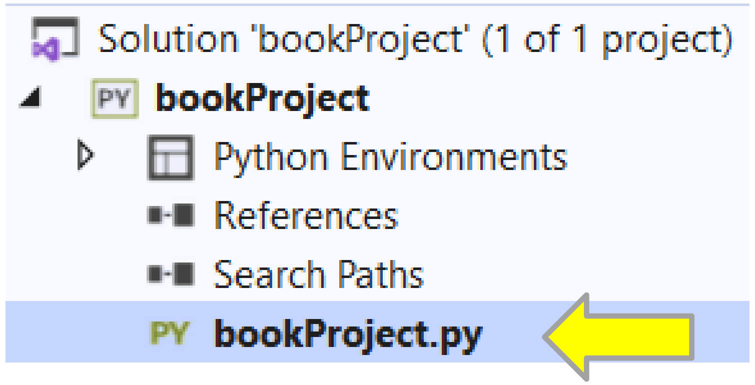
bookProject.py – 第 8 页

```
    #Sets Container's next to nothing
            con.next = None
            #Sets bookList to the new container if it is empty
            if(bookList == None):
                bookList = con
            #Adds the Container to the end of the bookList
            else:
                while(temp.next != None):
                    temp = temp.next
                temp.next = con
            #Increases number of books added
            index+=1
    f.close() #Closes file
except: #Does nothing if there is no existing file
    done = 0
```



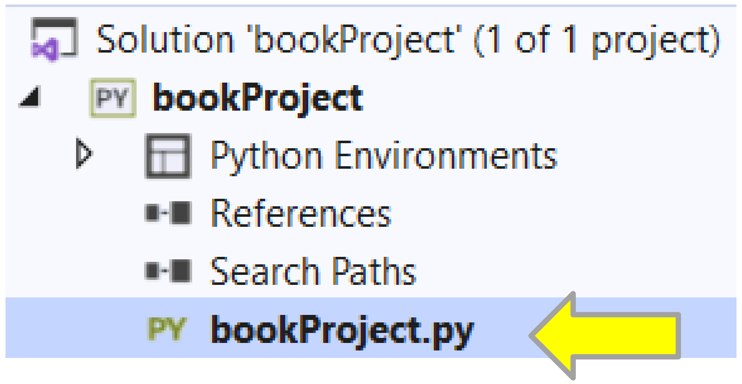
bookProject.py – 第 9 页

```
#Erase the content of the bookList
def erase(filename):
    global bookList #Gives access to global bookList
    f = open(filename, 'r+')
    f.truncate(0) # Erases contents of the file
    bookList = None # Erases contents of the bookList
    load("bookList.txt") # Make sure the file does not have contents
    print('All books have been deleted from bookList')
    f.close() # Closes file
```



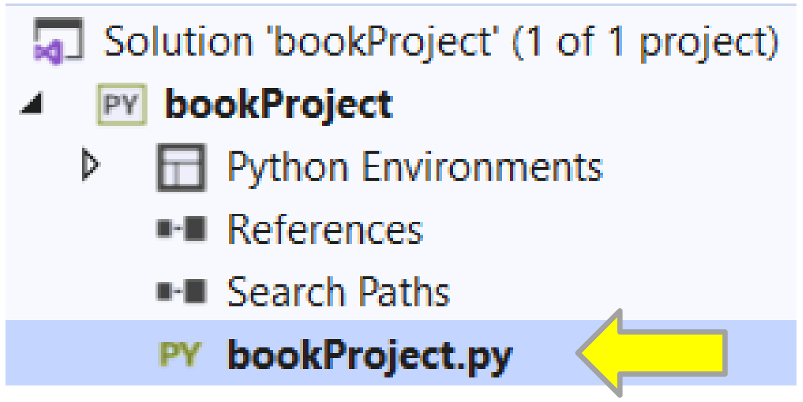
bookProject.py – 第 10 页

```
#Perform a different function based on user choice
def Action(c):
    if (c == 'a'): #Adds a book
        type = 1 #Default type (artBook)
        #Makes user enter name and copies
        name = input("Enter book name: ")
        copies = input("Enter number of copies: ")
        copies = int(copies)
        #Makes user enter 0 or 1 for type only
        while(type != 0 and type !=1):
            print("Enter book type: 0 or 1 only ")
            print("0. artBook ")
            print("1. engBook")
            type = input()
            type = int(type)
```



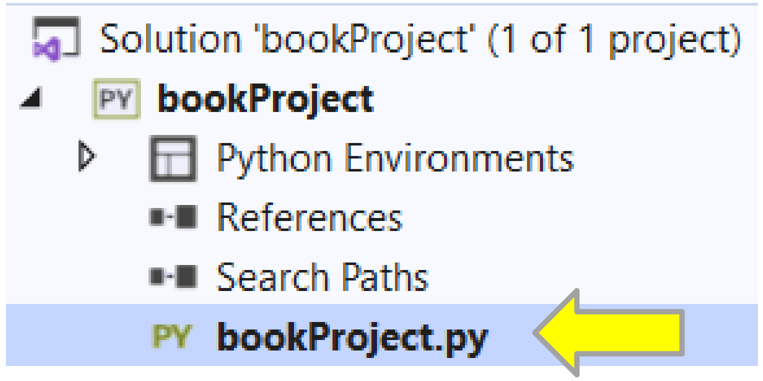
bookProject.py – 第 11 页

```
#Makes type an actual category
 if(type == 0):
     type = Category.artBook
 else:
     type = Category.engBook
 #Checks if the book is in the bookList
 bookResult = searchBook(name)
 #Reports if the book is added correctly or
 #if the book is already in the bookList
 if(bookResult == None):
     addBook(name, copies, type)
     print("Book added to bookList!")
 else:
     print("Book already present in the bookList!")
```



bookProject.py – 第 12 页

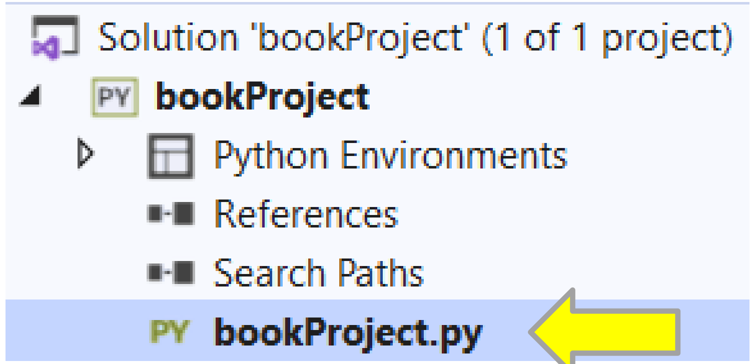
```
elif(c == 'd'): #Displays the books
    displayList()
elif(c == 'c'): #Changes the number of copies of a book
    name = input("Enter book name: ")
    #Checks if the book is in the bookList
    bookResult = searchBook(name)
    if (bookResult == None):
        print("Book not in bookList!")
    else:
        copies = input("Enter new number of copies: ")
        changeNumberOfCopies(bookResult, copies)
        print("Number of copies changed!")
elif(c == 's'): #Save the entered books into disk file
    save("bookList.txt")
elif(c == 'e'): #Erase the contents of the bookList
```



bookProject.py – 第 13 页

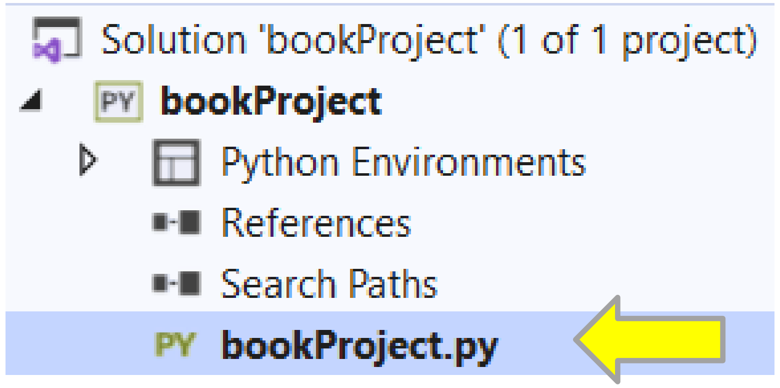
```
erase("bookList.txt")
    elif(c =='q'): #Exits the program
        exit
    #Tells the user they entered an incorrect input
    else:
        print (c+" is invalid input!\n")

# Main program starts here
choice = 'a' #Sets default choice
load("bookList.txt") #Load existing books from file to bookList
#Asks for user choice and then performs an action based on the
choice
while (choice != 'q'):
    print("\nBook List Manipulation")
    print("Please enter your selection:")
    print("\t a: add a new book")
```

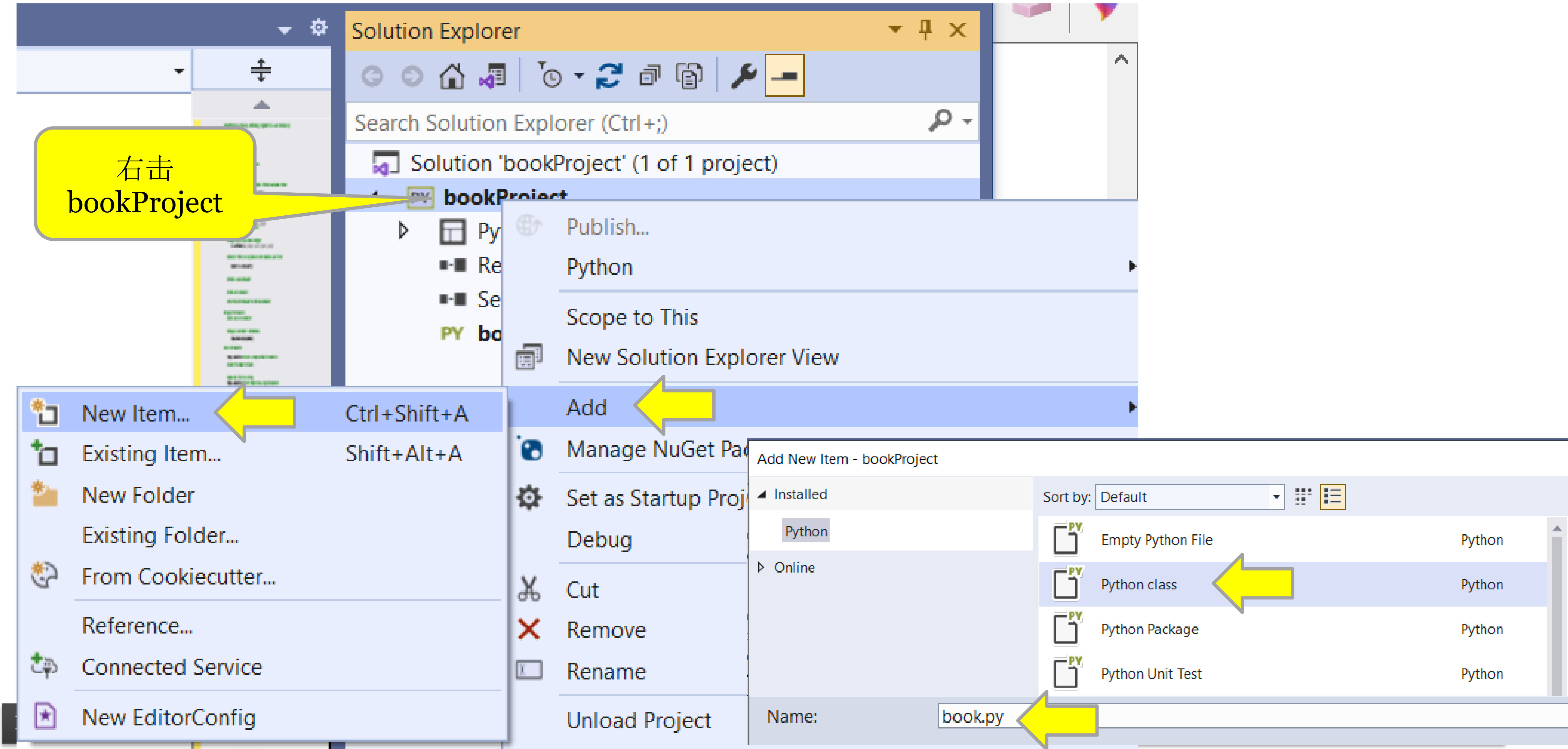


bookProject.py – 第 14 页

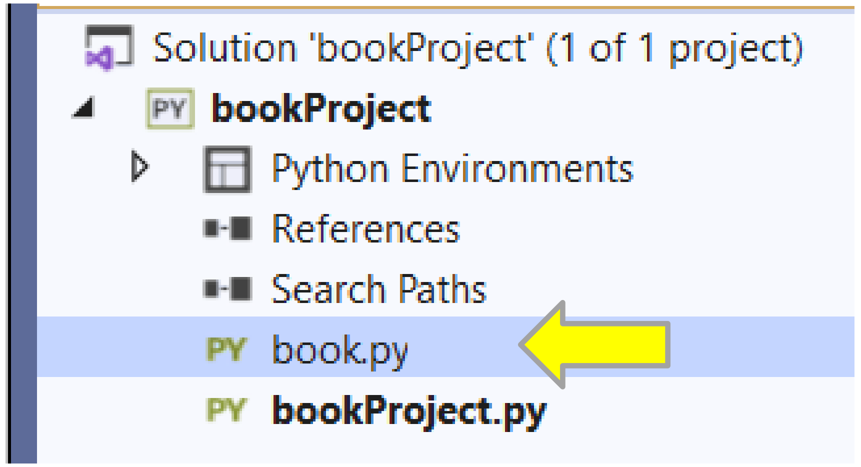
```
    print("\t d: display bookList")
    print("\t c: change copies of a book")
    print("\t s: save books into disk file")
    print("\t e: erase contents of the bookList")
    print("\t q: quit")
    choice = input()
    choice = choice[0:1]
    Action(choice)
save("bookList.txt") #Saves the bookList before exiting the program
```



添加新类：book.py


右击
bookProject
Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution 'bookProject' (1 of 1 project)
bookProject
Publish...
Python
Scope to This
New Solution Explorer View
Add
Manage NuGet Pac
Set as Startup Proj
Debug
Cut
Remove
Rename
Unload Project
New Item... Ctrl+Shift+A
Existing Item... Shift+Alt+A
New Folder
Existing Folder...
From Cookiecutter...
Reference...
Connected Service
New EditorConfig
Add New Item - bookProject
Installed
Python
Online
Sort by: Default
Empty Python File Python
Python class Python
Python Package Python
Python Unit Test Python
Name: book.py

Book.py 类

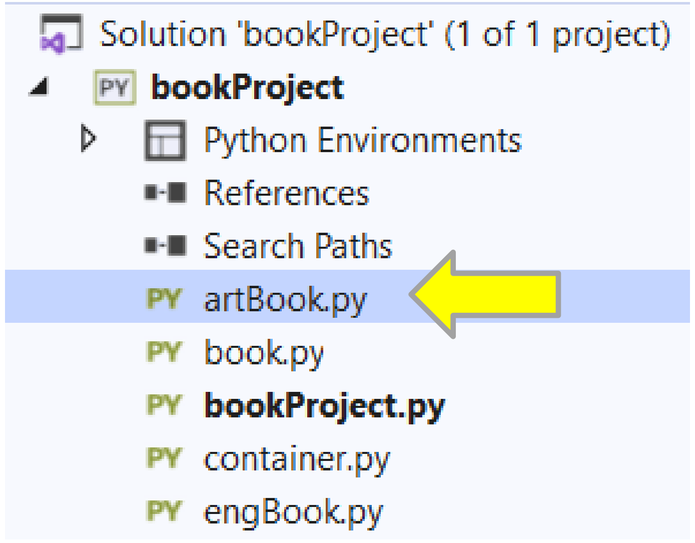

Solution 'bookProject' (1 of 1 project)
bookProject
Python Environments
References
Search Paths
book.py
bookProject.py

```
from enum import Enum
class Category(Enum):
     artBook = 0
     engBook = 1
class Book():
     #Book Constructor
     def __init__(self, bookName, noOfCopies, bType):
         self._name = bookName
         self._copies = noOfCopies
         self._bType = bType
      def getName(self):     #Returns name
         return self._name;
      def getCopies(self):   #Returns copies
         return self._copies;
      def getBookType(self): #Returns bookType type
         return self._bType;
      def changeNoOfCopies(self,num): #Changes number of copies
         self._copies=num
         return;
       def displayBook(self):# function will be overridden
         pass
```

ArtBook.py 类继承 Book 类

```
from book import *
class ArtBook(Book):
    #ArtBook Constuctor that implements Book constructor
    def __init__(self,name,copies,bookType):
        super().__init__(name,copies,bookType)

    #Displays Book's information. It overrides the based class's function
    def displayBook(self):
        print("Book name: "+self.getName())
        print("Copies: "+(str)(self.getCopies()))
        print("bookType: artBook")
```

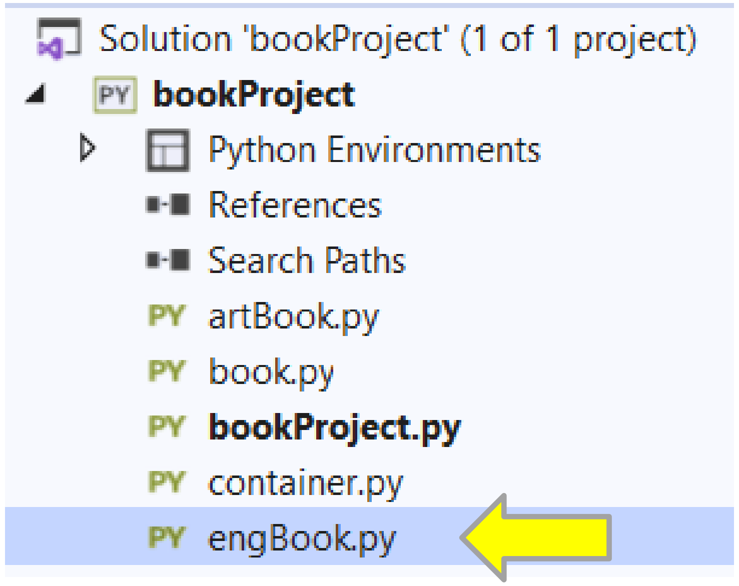


EngBook.py 类继承 Book 类

```
from book import *
class EngBook(Book):

    #EngBook Constuctor that implements Book constructor
    def __init__(self,name,copies,bookType):
        super().__init__(name,copies,bookType)

    #Displays Book's information
    def displayBook(self):
        print("Book name: "+self.getName())
        print("Copies: "+(str)(self.getCopies()))
      print("bookType: engBook")
```


Solution 'bookProject' (1 of 1 project)
bookProject
Python Environments
References
Search Paths
artBook.py
book.py
bookProject.py
container.py
engBook.py

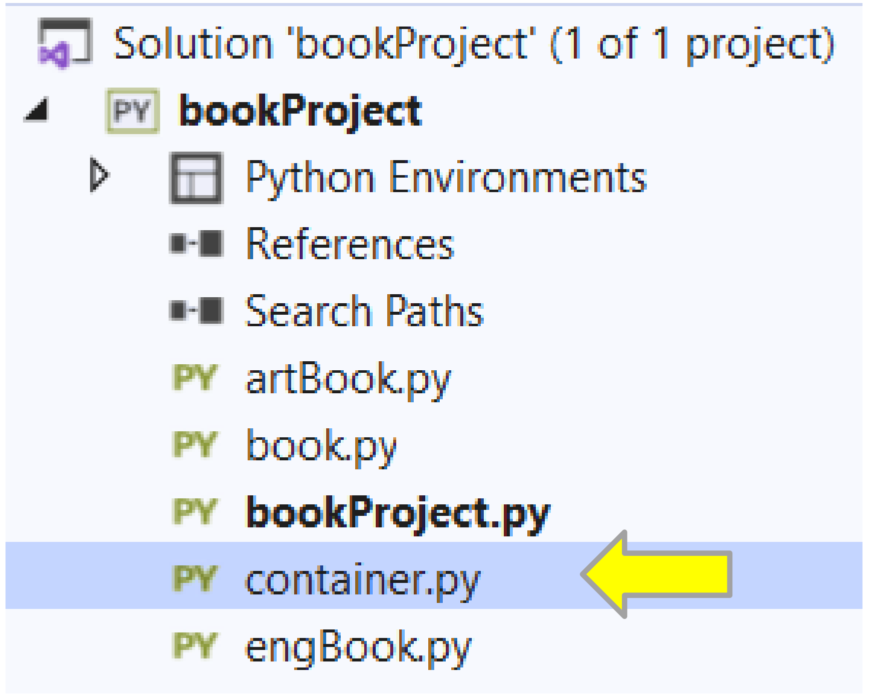
Container.py 类：包容 vs. 继承

我们可以使用继承关系或包容关系将两个类关联起来。

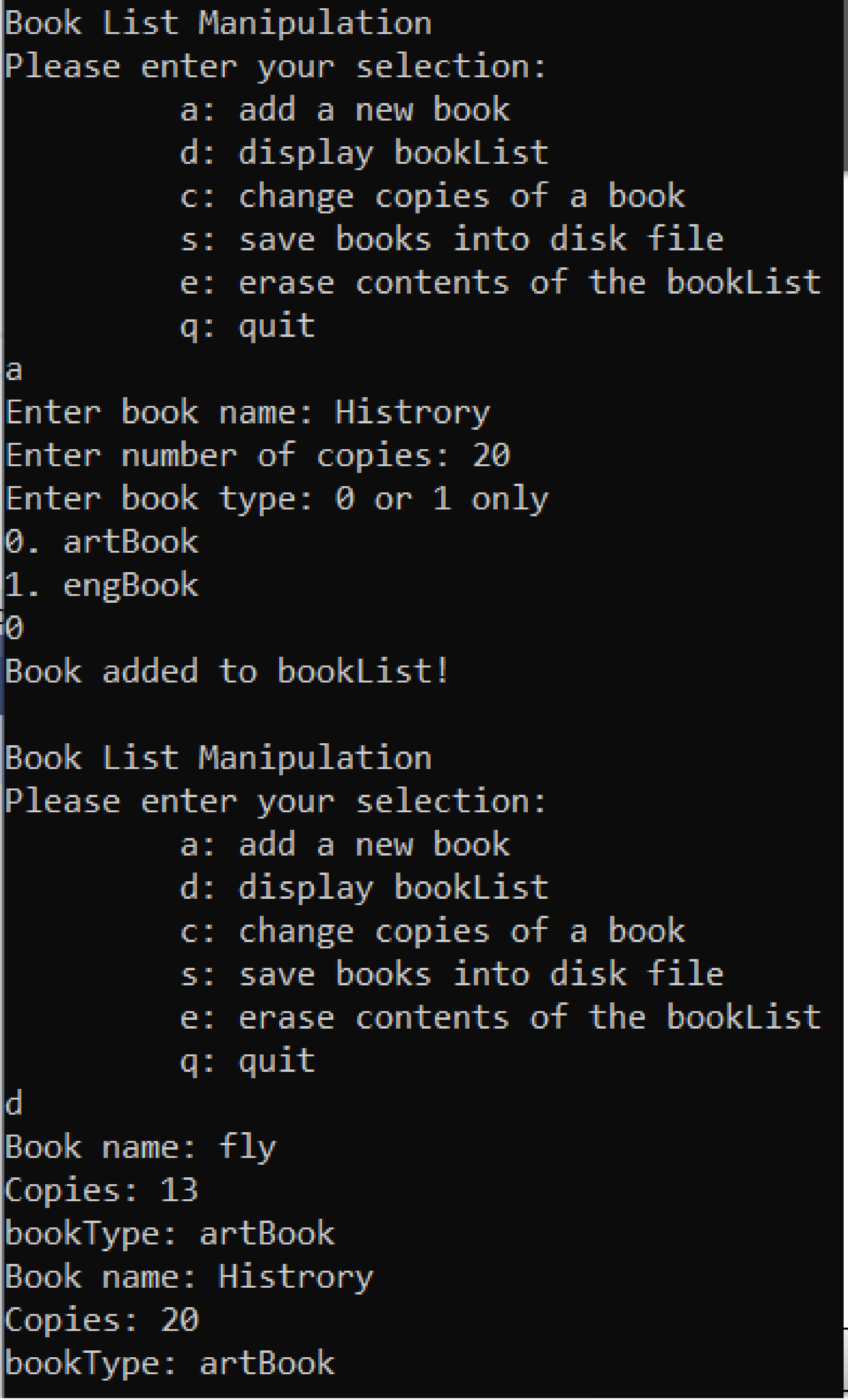
- ❑ 继承：称为 is-a 关系。它从父类复制成员
- ❑ 包容关系称为 has-a 关系。
- ❑ 在本例中，其中包含一个
  - ▪ book 对象
  - ▪ 链接到下一个节点，形成一个链表

```
class Container():
    #Container Constructor
    def __init__(self):
        self.book = None
        self.next = None
```

Container 包含一个使用包含关系的 Book 类

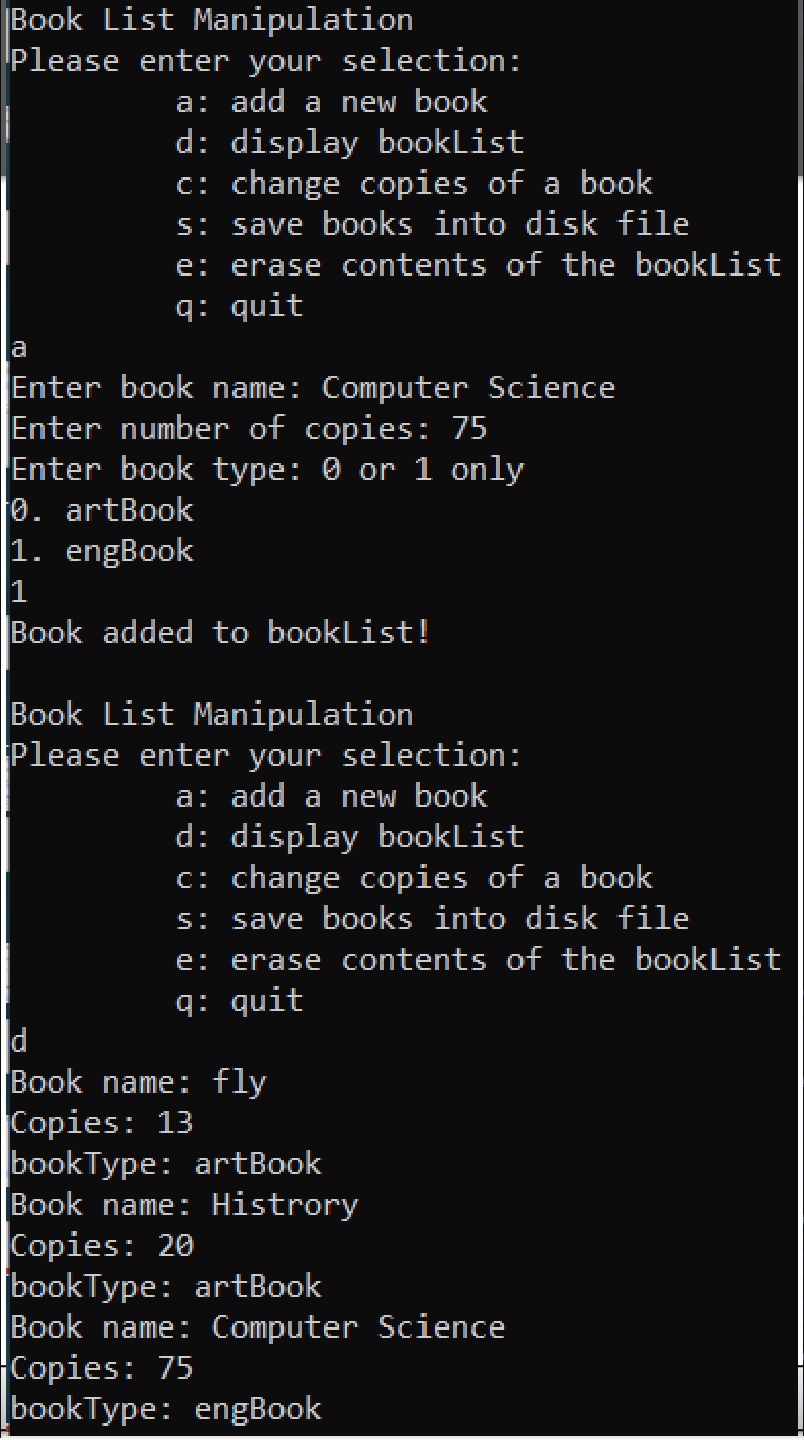


Solution 'bookProject' (1 of 1 project)
- bookProject
  - Python Environments
  - References
  - Search Paths
  - artBook.py
  - book.py
  - bookProject.py
  - container.py
  - engBook.py



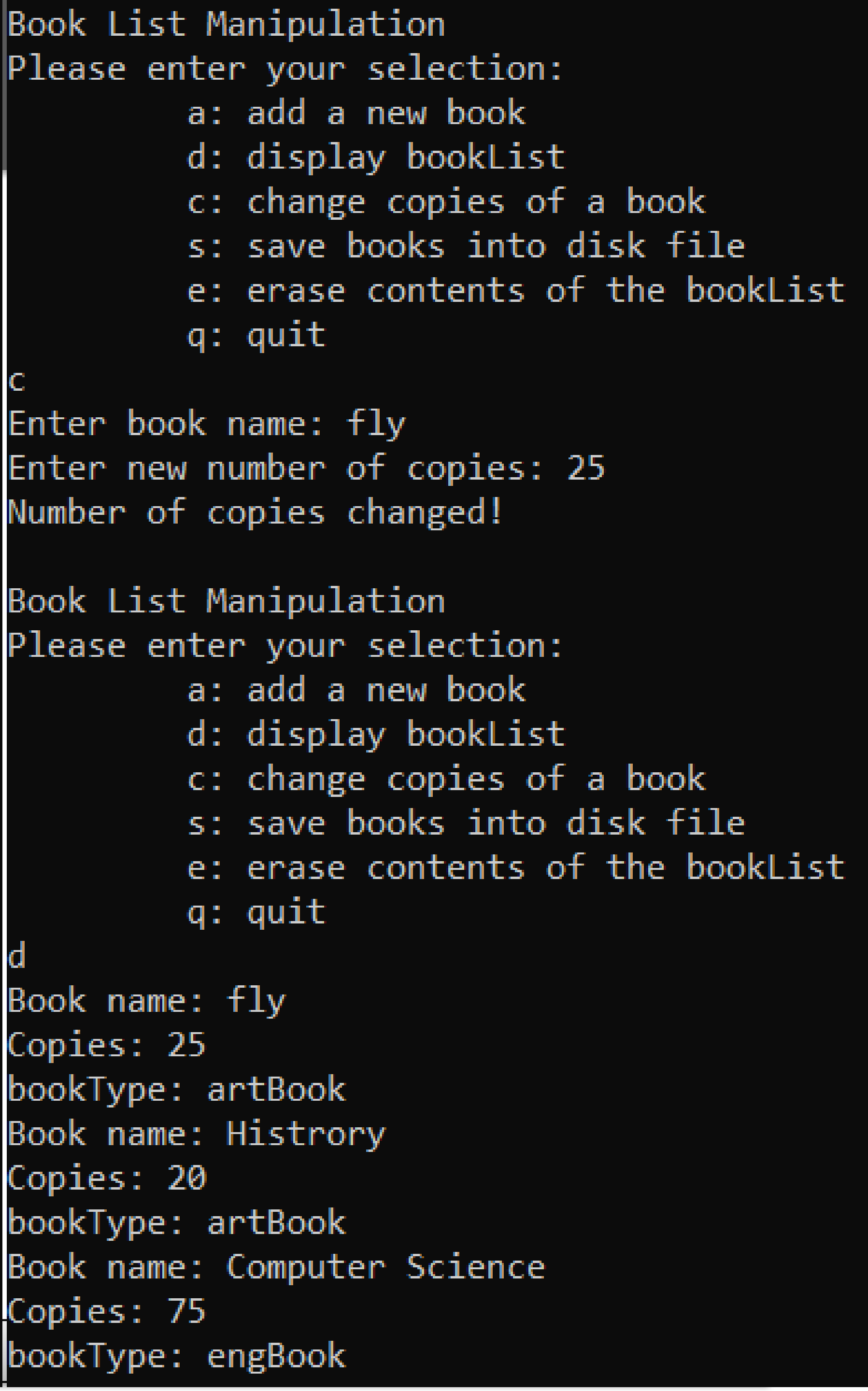
```
Book List Manipulation
Please enter your selection:
        a: add a new book
        d: display bookList
        c: change copies of a book
        s: save books into disk file
        e: erase contents of the bookList
        q: quit
a
Enter book name: Histrory
Enter number of copies: 20
Enter book type: 0 or 1 only
0. artBook
1. engBook
0
Book added to bookList!

Book List Manipulation
Please enter your selection:
        a: add a new book
        d: display bookList
        c: change copies of a book
        s: save books into disk file
        e: erase contents of the bookList
        q: quit
d
Book name: fly
Copies: 13
bookType: artBook
Book name: Histrory
Copies: 20
bookType: artBook
```



```
Book List Manipulation
Please enter your selection:
        a: add a new book
        d: display bookList
        c: change copies of a book
        s: save books into disk file
        e: erase contents of the bookList
        q: quit
a
Enter book name: Computer Science
Enter number of copies: 75
Enter book type: 0 or 1 only
0. artBook
1. engBook
1
Book added to bookList!

Book List Manipulation
Please enter your selection:
        a: add a new book
        d: display bookList
        c: change copies of a book
        s: save books into disk file
        e: erase contents of the bookList
        q: quit
d
Book name: fly
Copies: 13
bookType: artBook
Book name: Histrory
Copies: 20
bookType: artBook
Book name: Computer Science
Copies: 75
bookType: engBook
```



```
Book List Manipulation
Please enter your selection:
        a: add a new book
        d: display bookList
        c: change copies of a book
        s: save books into disk file
        e: erase contents of the bookList
        q: quit
c
Enter book name: fly
Enter new number of copies: 25
Number of copies changed!

Book List Manipulation
Please enter your selection:
        a: add a new book
        d: display bookList
        c: change copies of a book
        s: save books into disk file
        e: erase contents of the bookList
        q: quit
d
Book name: fly
Copies: 25
bookType: artBook
Book name: Histrory
Copies: 20
bookType: artBook
Book name: Computer Science
Copies: 75
bookType: engBook
```