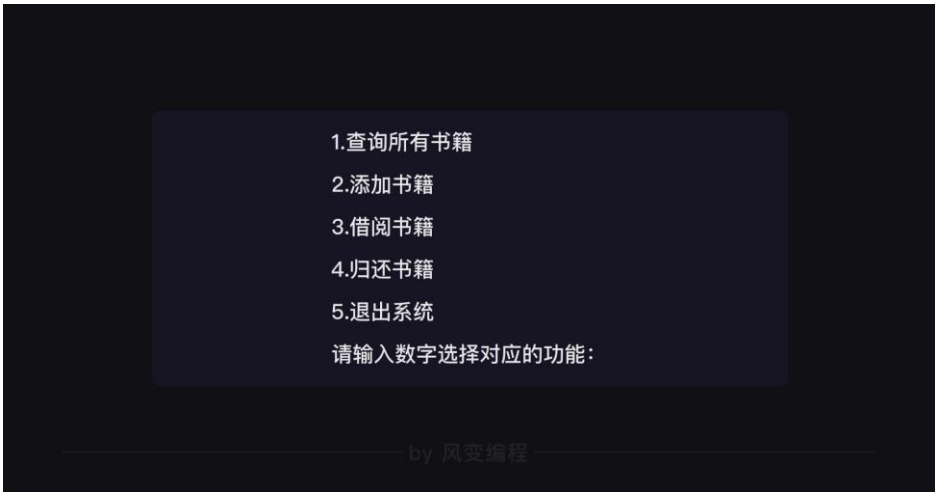
第 14 关　项目分析

根据题目需要，我们需要创建一个类 Book ，Book 类中含有四个基本的属性：书名、作家、推荐语和借阅状态。所以我们需要用到初始化方法__init__，让实例被创建时自动获得这些属性。



同时我们需要实现上图中的 4 个功能，那就得定义另一个类 BookManager，该类含有五个方法（含主菜单）：

①、主菜单：def menu()
②、显示书籍：show_all_book()
③、添加书籍：add_book()
④、借阅书籍：lend_book()
⑤、归还书籍：return_book()

而为什么要分为两个类呢，主要是我们 Book 类是为了让程序每次实例时初始化它的属性值，BookManager 类是为了实现我们的功能，二者区分开来，有利于代码的易读和整洁性

好了，我们再看回代码，Book 类不难理解，一个初始化化函数 __init__ 包含了四个实例化属性，而另一个函数 __str__ ，领过 13 关笔记的童鞋应该清楚，这个函数也是实例化后就执行，并返回 return 后面所带的值

```
class Book:


    def __init__(self, name, author, comment, state = 0):
        self.name = name
        self.author = author
        self.comment = comment
        self.state = state


    def __str__(self):
        status = '未借出'
        if self.state == 1:
            status = '已借出'
        return '名称：《%s》 作者：%s 推荐语：%s\n状态：%s ' % (self.name, self.author, self.comment, status)
```

接下来我们来分析一下 BookManager 类

```
class BookManager:

    books = []
    def __init__(self):
        book1 = Book('惶然录','费尔南多·佩索阿','一个迷失方向且濒于崩溃的灵魂的自我启示，一首对默默无闻、
        book2 = Book('以箭为翅','简娘','调和空灵文风与禅宗境界，刻画人间之缘起缘灭。像一条柔韧的绳子，情
        book3 = Book('心是孤独的猎手','卡森·麦卡勒斯','我们渴望倾诉，却从未倾听。女孩、黑人、哑巴、醉鬼、
        self.books.append(book1)
        self.books.append(book2)
        self.books.append(book3)
```

这里也定义了一个初始化函数，其中 book1、book2、book3 是实例化对象，实例化了类 Book，即这三本书是默认存在的。接下来的三个 append 语句的结果是 self.books=[book1、book2、book3]，即实例化对象的属性 books 含有这三个值

```
def menu(self):
    print('欢迎使用流浪图书管理系统，每本沉默的好书都是一座流浪的岛屿，希望你有缘发现并着陆，为精神家
    while True:
        print('1.查询所有书籍\n2.添加书籍\n3.借阅书籍\n4.归还书籍\n5.退出系统\n')
        choice = int(input('请输入数字选择对应的功能：'))
        if choice == 1:
            self.show_all_book()
        elif choice == 2:
            self.add_book()
        elif choice == 3:
            self.lend_book()
        elif choice == 4:
            self.return_book()
        elif choice == 5:
            print('感谢使用！愿你我成为爱书之人，在茫茫书海里相遇。')
            break
```

接下来的 menu 方法，主要是提供了我们对于功能的选择，根据我们的选择跳转执行相对应的方法

```
def show_all_book(self):
    print('书籍信息如下：')
    for book in self.books:
        print(book)
        print('')
```

show_all_book 方法中，for book in self.books，根据我们之前学的知识，这一步是为了遍历上方说的 self.books 列表，而因为实例化 Book 类后会执行 __str__ 函数，即会有返回值，所以可以显示目前所有的书籍信息

```
def add_book(self):
    new_name = input('请输入书籍名称：')
    new_author =  input('请输入作者名称：')
    new_comment = input('请输入书籍推荐语：')
    new_book = Book(new_name, new_author, new_comment)
    self.books.append(new_book)
    print('书籍录入成功！\n')
```

add_book 方法中，代码比较简单，是让我们手动输入后，把这些值传入后实例化类 Book ，并在 self.books 中添加

```
def check_book(self,name):
    for book in self.books:
        if book.name == name:
            return book
    else:
        return None
```

check_book 方法，我们依旧是遍历我们的 self.books 列表，相信同学们没有忘记 self.books=[book1、book2、book3]，而 book1、book2、book3 是我们的实例化对象，所以 book.name 实际上就是我们的实例化属性 name，当实例化属性 name 等于我们外部传入的值 name 时，我们就返回当前匹配的实例化对象，否则返回空

```
def lend_book(self):
    name = input('请输入书籍的名称：')
    res = self.check_book(name)

    if res != None:
        if res.state == 1:
            print('你来晚了一步，这本书已经被借走了噢')
        else:
            print('借阅成功，借了不看会变胖噢~')
            res.state = 1
    else:
        print('这本书暂时没有收录在系统里呢')
```

lend_book 方法，手动输入后，调用我们上面说到的 check_book 方法，如果返回值不为空，则继续下一步，再判断我们的 state 属性是否为 1，是则无法借阅，否则可以借阅书籍，并把当前书籍的 state 值改为 1（当该书籍未归还则无法借阅）

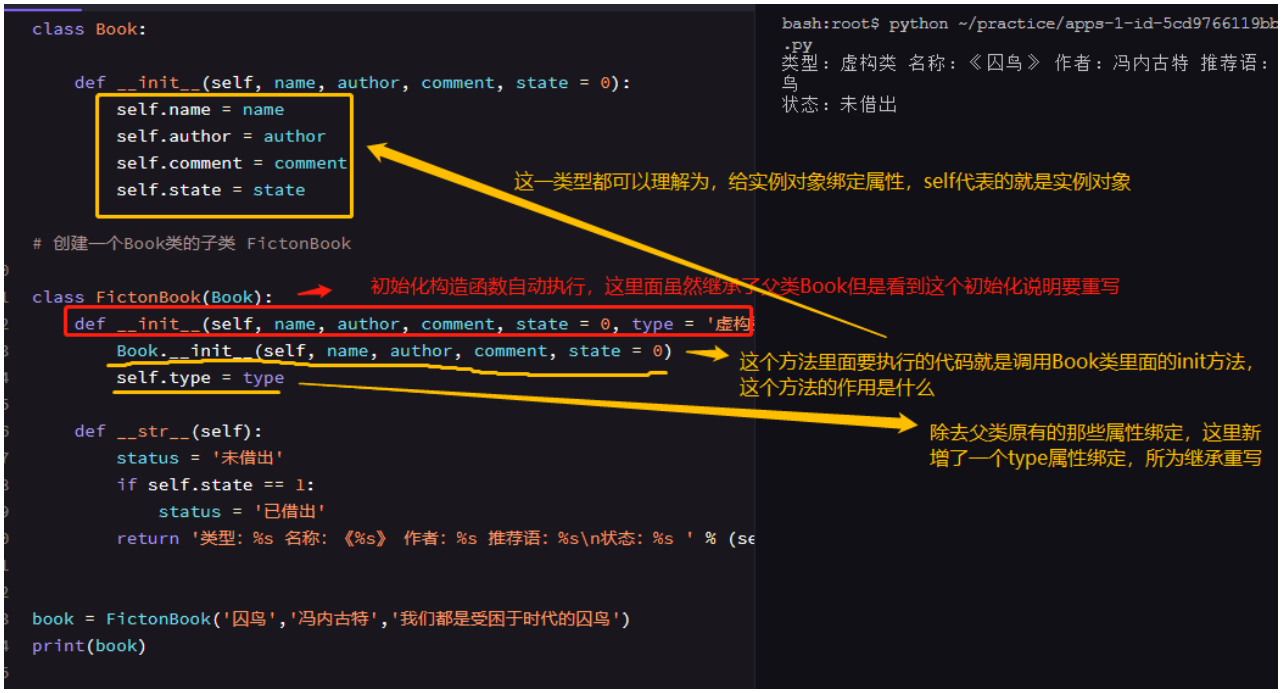
```
def return_book(self):
    name = input('请输入归还书籍的名称：')
    res = self.check_book(name)
    # 调用check_book方法，将返回值赋值给变量res
    if res == None:
    # 如果返回的是空值，即这本书的书名不在系统里
        print('没有这本书噢，你恐怕输错了书名~')
    else:
    # 如果返回的是实例对象
        if res.state == 0:
         # 如果实例属性state等于0，即这本书的借阅状态为'未借出'
            print('这本书没有被借走，在等待有缘人的垂青呢！')
        else:
         # 如果实例属性state等于1，即状态为'已借出'
            print('归还成功！')
            res.state = 0
            # 归还后书籍借阅状态为0，重置为'未借出'
```

最后一个方法 return_book，依旧是传入 check_book 方法进行判断，根据返回值判断该书籍是否之前有存在在这个系统中，有则继续继续判断 state 值，不为 0 则归还成功

最后我们实例化类 BookManager，再调用它的 menu 方法执行开始程序的运行

14 关课后练习注解:

基础练习



进阶练习