

# 实训报告：图书管理系统

[REDACTED]

## 一、需求分析

本程序实现了**图书管理、用户管理、数据库管理、简单结算**的功能。试简单分列如下：

- **图书管理**
  - 图书添加/删除功能
  - 图书信息编辑功能
  - 图书一览功能
  - 图书上/下架功能
- **用户管理**
  - 用户添加/删除功能
  - 用户信息编辑功能
  - 用户一览功能
- **数据库管理**
  - 图书、用户数据库导出/导入功能
    - 图书数据库支持压缩功能
    - 数据库为便于维护的csv格式
    - 数据库添加BOM确保读取时不乱码
- **结算**
  - 书籍购买功能
- **杂项**
  - 通用编码：本程序使用UTF-8编码
    - 在Windows下自动切换编码，提高兼容性
  - 错误检查：检查用户输入操作是否合法
  - 越界检查：避免用户错误输入造成数组越界
  - 命令行参数：方便操作
    - `-h` `--help`
      - 显示命令帮助
    - `-u` `--user` `{path}`
      - 导入用户数据库
    - `-b` `--book` `{path}`
      - 导入书籍数据库

## 二、文件结构

以下是本实训的文件结构，通过 `tree` 生成。**声明** 以下所列文件之作者都是本人。

```
1  .
2  ├── LICENSE
3  ├── Makefile.am # autoconf脚本
4  ├── Makefile.in
5  ├── README.md
6  ├── aclocal.m4 # autoconf相关
7  ├── assets # md文档使用素材
8  │   ├── image-20230515000408869.png
9  │   ├── image-20230515000459499.png
10  │   ├── image-20230515002112455.png
11  │   ├── image-20230515002152321.png
12  │   ├── image-20230515004828789.png
13  │   ├── image-20230515005009981.png
14  │   └── uml.drawio.svg #UML图
15 ├── autom4te.cache # automake编译缓存
16 │   ├── output.0
17 │   ├── output.1
18 │   ├── output.2
19 │   ├── requests
20 │   ├── traces.0
21 │   ├── traces.1
22 │   └── traces.2
23 ├── aux
24 │   └── bookstore.cpp # 本实训由此重写而来，都是本人所作
25 ├── build_aux # 存放安装卸载等脚本
26 │   ├── depcomp
27 │   ├── install-sh
28 │   └── missing
29 ├── configure # configure脚本
30 ├── configure.ac # configure源
31 ├── main-darwin-aarch64 # macOS下编译得到的二进制文件， aarch64
32 ├── main-linux-amd64 # linux下编译得到的二进制文件， amd64
33 ├── main-windows-amd64.exe # windows下编译得到的二进制文件， amd64
34 ├── src # 源代码
35 │   ├── Makefile # make用自动编译脚本，Windows用，其他环境应该用autotool
36 │   ├── base.hpp # Base类定义和实现
37 │   ├── book.cpp
38 │   ├── book.hpp # Book类定义和实现
39 │   ├── books.csv # Book用测试样例数据库
40 │   ├── csv.cpp # CSV读写功能定义和实现
41 │   ├── csv.hpp
42 │   ├── data.h # Static成员定义
43 │   ├── database.hpp # Database容器模板库定义和实现
44 │   ├── database.hpp
45 │   ├── exec.cpp # 程序主要流程定义和实现
46 │   ├── exec.hpp
47 │   ├── interface.cpp # 程序界面依赖库定义和实现
48 │   ├── interface.hpp
49 │   ├── main.cpp # 主程序，程序入口代码
50 │   ├── user.cpp # User类定义和实现
51 │   ├── user.hpp
52 │   └── users.csv # User用测试样例数据库
53 └── report.md # 报告
54
55 6 directories, 48 files
```

macOS平台的程序是在本人电脑上编译完成的；Windows平台的程序是在机房电脑编译完成的；Linux平台的程序是在Arch下编译完成的，经过测试都可以正常运行。

### 编译指南

代码遵守C++17标准，使用autotool构建

类UNIX

使用autotool自动编译

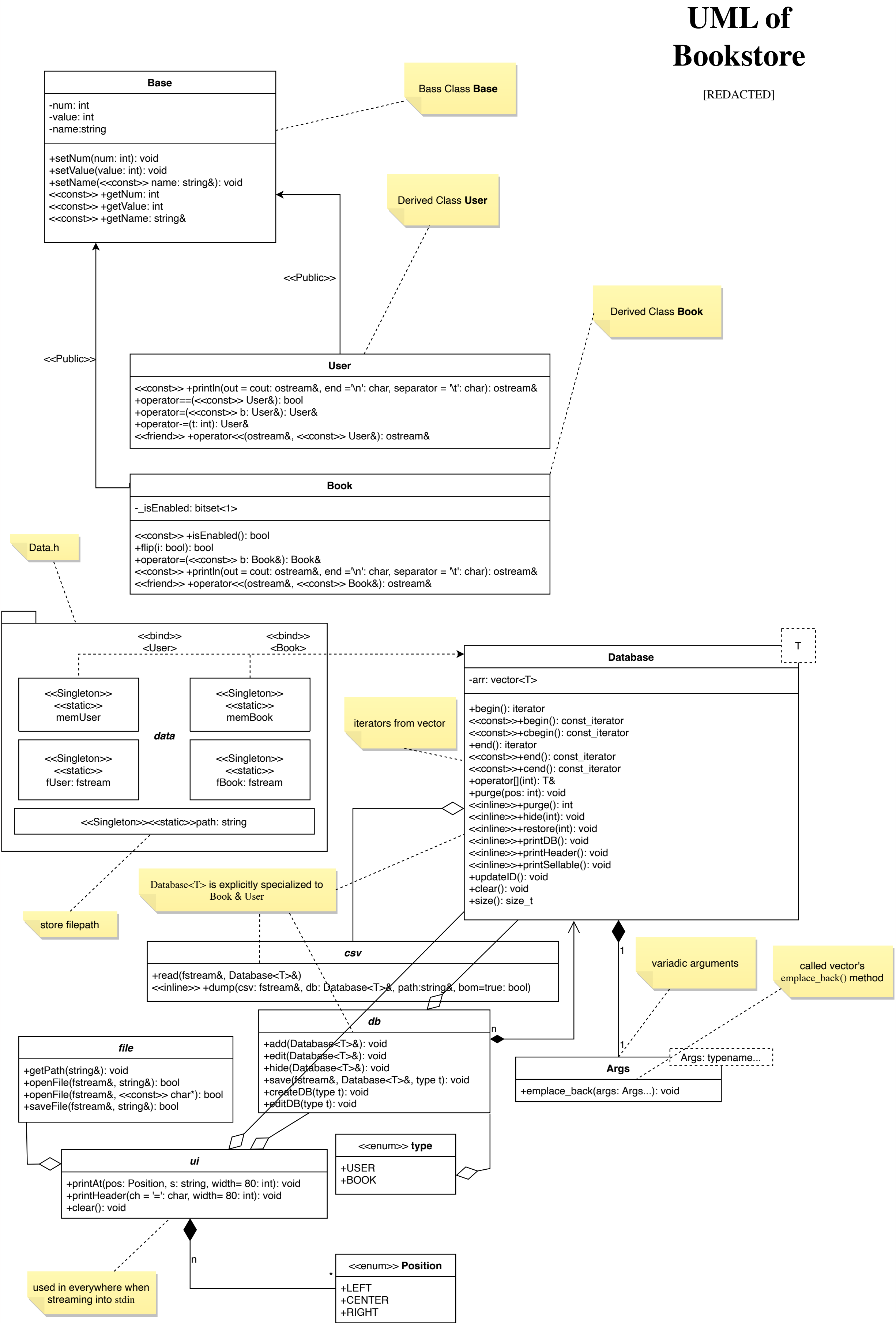
```
1  ./configure
2  make -j${nproc}
```

这将在根目录生成可执行文件 `bookstore`。

通过 src/Makefile 使用minGW自带的 minGW-make 来编译。其中 make clean 清除所有目标文件； make clear 清除所有编译文件。

三、系统结构略图

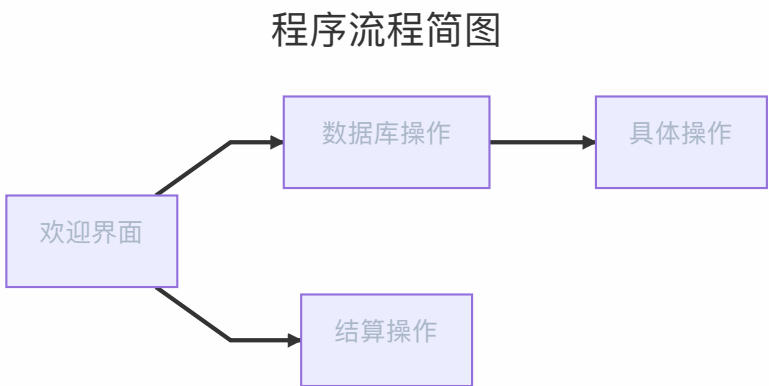
我们用UML图来描述代码结构，如下所示：（assets文件夹中有pdf版本）



四、具体功能介绍及知识点分析

具体功能介绍

程序运行的大致流程如下图所示：



菜单层级、具体操作如下：

- 修改USER数据库
  - 用户一览

- 显示用户的ID、姓名、余额： **注** 程序中所有整数类型为 unsigned long long 亦即 int64 ，满足大部分需求。考虑执行速度不采用 int128
  - 通过 println() 和 << 实现，前者可以调整输出流对象、是否换行、分隔符，较为方便。
2. 用户添加
- 输入用户名称和余额，将其添加在数据库最后，自动分配ID
  - 复杂度  $O(1)$ . 采用emplace\_back()方式，原地构造对象，不必调用复制或构造函数，较push\_back()更快
3. 用户修改
- 输入ID修改用户姓名和余额
4. 载入另一数据库
- 通过输入路径载入数据库，支持绝对/相对路径两种
  - 数据库载入内存，理论上数据库大小只受内存限制
  - 默认路径为 ./users.csv
5. 用户删除
- 复杂度  $O(kn)$ . 删除  $k$  名用户（输入ID，空格分隔），自动前移其之后所有用户的ID。
6. 保存数据库
- 添加BOM并导出，避免诸如excel等程序显示乱码
  - 默认路径为 ./users.csv
7. 返回上级菜单
8. 清空数据库
- 复杂度  $O(n)$ . 实质上相当于创建新USER数据库

2. 修改BOOK数据库

1. 上架书一览
- 显示书籍ID、书名、价格
  - 只列出所有上架（即对象属性的 \_isEnabled 为真的）书籍
  - 通过 println() 和 << 实现，前者可以调整输出流对象、是否换行、分隔符，较为方便。
2. 所有书一览
- 显示书籍ID、书名、价格、贩卖状态
  - 列出全部书籍，方便管理者得知各书的贩卖状态
3. 书籍添加
- 输入书籍和余额，将其添加在数据库最后，自动分配ID，默认设置为上架
  - 复杂度和具体实现同上
4. 上/下架书籍
- 输入书籍ID（可多个，空格分隔）设置其上下架状态，更为灵活，贴近日常需求
  - 状态的改变不影响ID
  - 对象属性的 \_isEnabled 存储这一信息。
5. 书籍删除
6. 所有下架书籍删除
- 复杂度  $O(2n)$ . 将所有下架书籍全数删除，缩小数据库体积
  - 搭配4操作可以实现数据库压缩
7. 载入另一数据库
- 默认路径为 ./books.csv
8. 保存数据库
- 默认路径为 ./books.csv
9. 返回上级菜单
10. 清空数据库
3. 结帐
- 在上述两种数据库加载的情况下，通过ID选择对应用户以及书籍和购买数量来结算
4. 退出程序

此外，程序中多次调用了系统的清屏操作：Windows下(cmd,powershell)是 cls ，Linux和macOS下(ncurses)是 clearn ，针对不同平台做了适配。

通过规定输出宽度，可以通过填充空格的方式达到文本置中/置右，且填充数

```
1 | n(\textup{Space})=\begin{cases}w - l \hspace{7.5mm}\textup{where p = RIGHT}\\2 | \frac{w - l}{2}\hspace{10mm}\textup{where p = CENTER}\end{cases}
```

其中  $w$  是输出宽度， $l$  是字符串长度， $p$  是位置。代码中的 ui::printAt() 用到了 (1)。

知识点运用

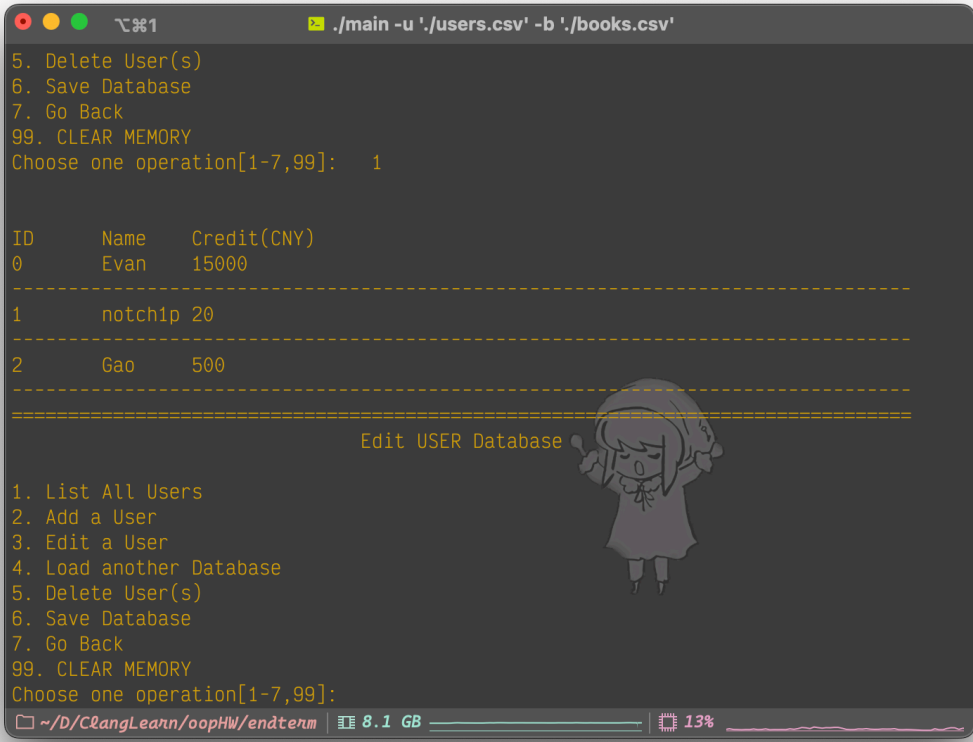
以下按类列出相对不基础的知识点：

- 函数
  - 内联函数
  - 带默认形参值的函数
  - 函数重载
- 类与对象
  - 类成员的访问控制
  - 移动构造函数
  - 枚举类型
- 数据的共享与保护
  - 命名空间
  - 多文件结构与编译预处理命令
  - 静态成员
  - 友元
  - 数据保护（ const ）
- 指针、数组相关
  - 对象数组
  - 指针传参、引用传参
  - 指针运算
  - 动态内存分配
- 类的继承
- 多态、泛型、metaprogramming
  - 运算符重载
  - 模板
    - 实例化（Instantiate）机制
    - Template Specialization
  - 迭代器
  - Lambda Expression
  - Trailing Return Type
  - 函数对象的参数绑定
- STL、流类库
  - vector string bitset
  - 文件流、字符串流、输入输出流
  - 读写加速（将cin cout的速度提高到scanf printf的水准）

五、运行情况与截图

以下是在macOS下通过 make 经由 clang++ 编译得到的程序：

```
1 notch1p in oopHW/EndTerm on 7 master [↑!?] took 20.7s
2 [I] → ./main -u './users.csv' -b './books.csv'
3 Loaded USER database
4 Loaded BOOK database
5
6 =====
7
8             Do Something...
9
10            1. Edit USER
11            2. Edit BOOK
12            3. Checkout
13            4. Exit
14
15 Input:  1
16
17 =====
18
19             Edit USER Database
20
21 1. List All Users
22 2. Add a User
23 3. Edit a User
24 4. Load another Database
25 5. Delete User(s)
26 6. Save Database
27 7. Go Back
28 99. CLEAR MEMORY
29 Choose one operation[1-7,99]: 1
30
31 ID  Name  Credit(CNY)
32 0 Evan  15000
33 -----
34 1 notch1p 20
35 -----
36
37 =====
38
39             Edit USER Database
40
41 1. List All Users
42 2. Add a User
43 3. Edit a User
44 4. Load another Database
45 5. Delete User(s)
46 6. Save Database
47 7. Go Back
48 99. CLEAR MEMORY
49 Choose one operation[1-7,99]:2
50
51 =====
52
53             Add a user
54
55 Enter Username:
56 Gao
57 Enter Credit:
58 500
59
60 =====
61
62             Edit USER Database
63
64 1. List All Users
65 2. Add a User
66 3. Edit a User
67 4. Load another Database
68 5. Delete User(s)
69 6. Save Database
70 7. Go Back
71 99. CLEAR MEMORY
72 Choose one operation[1-7,99]: 1
73
74 ID  Name  Credit(CNY)
75 0 Evan  15000
76 -----
77 1 notch1p 20
78 -----
79 2 Gao  500
80 -----
81
82 =====
83
84             Edit USER Database
85
86 1. List All Users
87 2. Add a User
88 3. Edit a User
89 4. Load another Database
90 5. Delete User(s)
91 6. Save Database
92 7. Go Back
93 99. CLEAR MEMORY
94 Choose one operation[1-7,99]:
```



从文件加载数据库&用户添加

```
1 =====
2
3             (Un)Hide Books
4 Actions: 1.Hide Books; 2. Restore Books ≥3. Go Back  1
5 Input ID (Split by SPACE):
6 2
7
8 =====
9
10            Edit BOOK Database
11
12 1. List All Sellable Books
```



```
10 2. List All Books
11 3. Add a Book
12 4. (Un)Hide Book(s)
13 5. Delete Book(s)
14 6. Delete All Hidden Books
15 7. Load another Database
16 8. Save Database
17 9. Go Back
18 99. CLEAR MEMORY
19 Choose one operation[1-9,99]: 2
20
21
22 ID  Name  Price Selling?
23 0 Book1 1000  True
24 -----
25 1 Book2 500  True
26 -----
27 2 Book3 200  False
28 -----
29 =====
30                               Edit BOOK Database
31
32 1. List All Sellable Books
33 2. List All Books
34 3. Add a Book
35 4. (Un)Hide Book(s)
36 5. Delete Book(s)
37 6. Delete All Hidden Books
38 7. Load another Database
39 8. Save Database
40 9. Go Back
41 99. CLEAR MEMORY
42 Choose one operation[1-9,99]:6
43 Purged 1 Books
44 =====
45                               Edit BOOK Database
46
47 1. List All Sellable Books
48 2. List All Books
49 3. Add a Book
50 4. (Un)Hide Book(s)
51 5. Delete Book(s)
52 6. Delete All Hidden Books
53 7. Load another Database
54 8. Save Database
55 9. Go Back
56 99. CLEAR MEMORY
57 Choose one operation[1-9,99]: 2
58
59
60 ID  Name  Price Selling?
61 0 Book1 1000  True
62 -----
63 1 Book2 500  True
64 -----
65 =====
66                               Edit BOOK Database
67
68 1. List All Sellable Books
69 2. List All Books
70 3. Add a Book
71 4. (Un)Hide Book(s)
72 5. Delete Book(s)
73 6. Delete All Hidden Books
74 7. Load another Database
75 8. Save Database
76 9. Go Back
77 99. CLEAR MEMORY
78 Choose one operation[1-9,99]:
```



书籍下架&下架书籍删除

```
1 =====
2                               Saving Database as...
3 input path:(enter -1 for default path)
4 !!use '/' instead of '\' on Windows!!
5 ./books.csv
6                               Save Succeed
7 =====
8                               Edit BOOK Database
9
10 1. List All Sellable Books
11 2. List All Books
12 3. Add a Book
13 4. (Un)Hide Book(s)
14 5. Delete Book(s)
15 6. Delete All Hidden Books
16 7. Load another Database
17 8. Save Database
18 9. Go Back
19 99. CLEAR MEMORY
20 Choose one operation[1-9,99]:
```

保存数据库

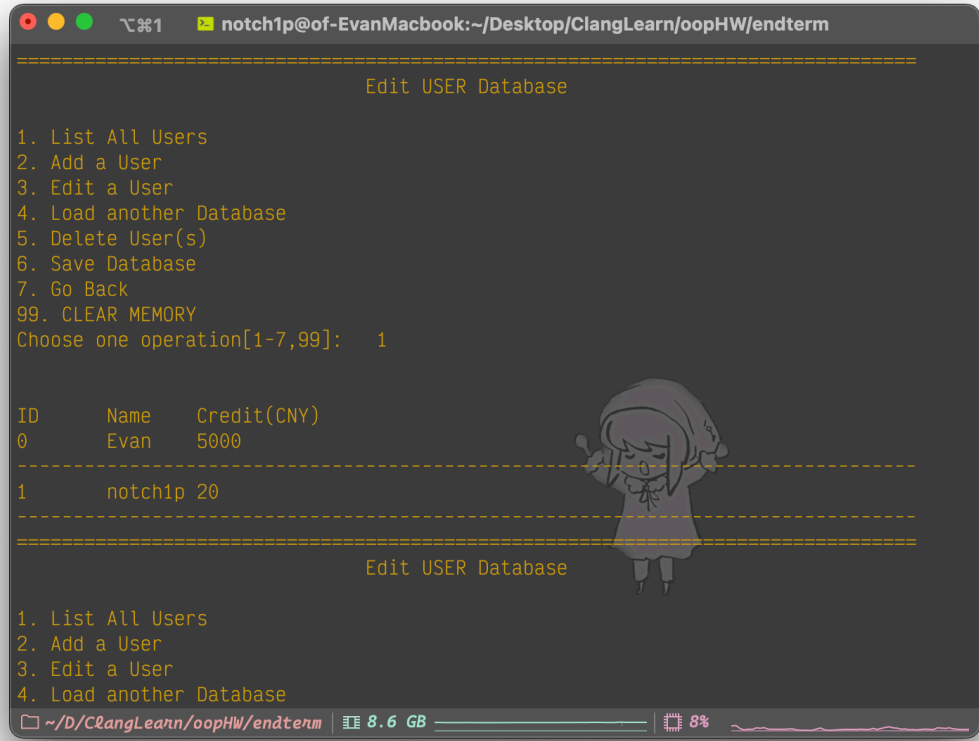
原来的数据库

ID	Name	Price	Selling?
0	Book1	1000	True
1	Book2	500	True
2	Book3	200	False

保存后的数据库

ID	Name	Price	Selling?
0	Book1	1000	True
1	Book2	500	True

```
1 =====
2                               Checkout
3
4
5 ID  Name  Price
6 0 Book1 1000
7 -----
8 1 Book2 500
9 -----
10 Enter USERID BOOKID and Quantity(Split by SPACE):
11 0 0 10
12 Total: 10000(CNY) for 10x Book1
13 Pay?(Yes/anything else) Yes
14 =====
15                               Do Something...
16
17                               1. Edit USER
18                               2. Edit BOOK
19                               3. Checkout
20                               4. Exit
21 Input:  1
22 =====
23                               Edit USER Database
24
25 1. List All Users
26 2. Add a User
27 3. Edit a User
28 4. Load another Database
29 5. Delete User(s)
30 6. Save Database
31 7. Go Back
32 99. CLEAR MEMORY
33 Choose one operation[1-7,99]: 1
34
35
36 ID  Name  Credit(CNY)
37 0 Evan  5000
38 -----
39 1 notchlp 20
40 -----
41 =====
42                               Edit USER Database
43
44 1. List All Users
45 2. Add a User
46 3. Edit a User
47 4. Load another Database
48 5. Delete User(s)
49 6. Save Database
50 7. Go Back
51 99. CLEAR MEMORY
52 Choose one operation[1-7,99]:
```



结算功能:Evan的余额由15000→5000