



หัวข้อโครงการ..ระบบยืม – คืนหนังสือห้องสมุด

Borrow and return library books

โดย

1. นาย กิตติธัช เข้มนาถ รหัส 6806022510475  
(ผู้ทดสอบตรวจสอบโปรแกรม)
2. นาย กฤษฎา บุตรศรี รหัส 6806022510161  
(ผู้เขียนโปรแกรม)
3. นาย ศักดิ์ชัย วอนพรมราช รหัส 6806022510335  
(ผู้ประสานงาน)
4. นาย โสภณวิทย์ เจริญวงศ์ รหัส 6806022510378  
(ผู้เขียนเล่ม)

โครงการนี้เป็นส่วนหนึ่งของการศึกษาหลักสูตรวิศวกรรมศาสตรบัณฑิต  
สาขาวิศวกรรมสารสนเทศและเครือข่าย ภาควิชาเทคโนโลยีสารสนเทศ  
คณะเทคโนโลยีและการจัดการอุตสาหกรรม มหาวิทยาลัยเทคโนโลยีพระจอม  
เกล้าพระนครเหนือ

ปีการศึกษา 2568

ลิขสิทธิ์ของมหาวิทยาลัยเทคโนโลยีพระจอมเกล้าพระนครเหนือ

## คำนำ

การจัดทำโครงการ “ระบบยืม - คืนหนังสือห้องสมุด” นี้เป็นส่วนหนึ่งของวิชา Computer Programming ของหลักสูตรวิศวกรรมศาสตรบัณฑิต สาขาวิศวกรรมสารสนเทศและเครือข่าย ภาควิชาเทคโนโลยีสารสนเทศ คณะเทคโนโลยีและการจัดการอุตสาหกรรม มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าพระนครเหนือ เพื่อให้นักศึกษาได้นำความรู้ที่เรียนมาทั้งหมดมาประยุกต์ใช้ในการพัฒนาโปรแกรมที่สามารถทำงานได้จริง โดยเน้นการออกแบบและเขียนโปรแกรมด้วยภาษา Python ซึ่งเป็นภาษาที่เรียนมาในวิชา Computer Programming โดยโครงการนี้จะช่วยการคิดวิเคราะห์และการแก้ปัญหาทางเทคนิค เพื่อเตรียมความพร้อมในการประกอบอาชีพด้านวิศวกรรมสารสนเทศและเครือข่ายในอนาคต

คณะผู้จัดทำหวังว่า รายงานฉบับนี้จะเป็นประโยชน์กับผู้อ่าน หรือนักเรียน นักศึกษา ที่กำลังหาข้อมูลเรื่องนี้อยู่ หากมีข้อเสนอแนะหรือข้อผิดพลาดประการใด ผู้จัดทำขอน้อมรับไว้และขอ อภัยมา ณ ที่นี้ด้วย

ผู้จัดทำ/คณะผู้จัดทำ

กลุ่ม Fantastic 4

## สารบัญ

| เนื้อหา   | หน้า      |
|---|-----------|
| คำนำ  | ก         |
| สารบัญ  | ข         |
| สารบัญรูปภาพ  | ค         |
| สารบัญรูปภาพ(ต่อ)                                       | จ         |
| สารบัญรูปภาพ(ต่อ)                                       | ฉ         |
| สารบัญตาราง   | ช         |
| <b>บทที่ 1    บทนำ</b>                                  | <b>1</b>  |
| 1.1 วัตถุประสงค์ของโครงการ                              | 1         |
| 1.2 ขอบเขตของโครงการ                                    | 1         |
| 1.3 ประโยชน์ที่ได้รับ                                   | 2         |
| 1.4 เครื่องมือที่คิดว่าจะต้องใช้                        | 2         |
| <b>บทที่ 2    ระบบยืม - คืนห้องสมุด</b>                 | <b>3</b>  |
| 2.1 เพิ่มข้อมูลหนังสือ books.dat                        | 3         |
| 2.2 เพิ่มข้อมูลสมาชิก members.dat                       | 5         |
| 2.3 ข้อมูลการยืม - คืน loans.dat                        | 8         |
| 2.4.ไฟล์ report.txt                                     | 11        |
| <b>บทที่ 3    การใช้งานระบบยืม - คืนหนังสือห้องสมุด</b> | <b>13</b> |
| 3.1 การใช้งานโปรแกรมระบบยืม - คืนห้องสมุด               | 10        |
| 3.2 การใช้งานโปรแกรมเพิ่มข้อมูล                         | 14        |
| 3.3 การใช้งานโปรแกรมแสดงข้อมูล                          | 18        |
| 3.4 การใช้งานโปรแกรมแก้ไขข้อมูล                         | 20        |
| 3.5 การใช้งานโปรแกรมลบข้อมูล                            | 22        |
| 3.6 การใช้งานโปรแกรมยืม - คืนหนังสือ                    | 24        |

## สารบัญ (ต่อ)

| เนื้อหา        | หน้า  |
|----------------|---|
| <b>บทที่ 4</b> | <b>อธิบายการทำงานของ Code</b>               |
| 4.1            | ฟังก์ชันไบนารีพื้นฐานในระบบยืม - คืนหนังสือ |
| 4.2            | ฟังก์ชันเมนูระบบยืม - คืนหนังสือห้องสมุด    |
| 4.3            | เมนู generate_report                        |
| 4.4            | main_menu ระบบยืม - คืนหนังสือห้องสมุด      |
| 4.5            | เมนู Book                                   |
| 4.6            | เมนู Members                                |
| 4.7            | เมนู Loans                                  |
| <b>บทที่ 5</b> | <b>สรุปผลการดำเนินงานและข้อเสนอแนะ</b>      |
| 5.1            | สรุปผลการดำเนินงาน                          |
| 5.2            | ปัญหาและอุปสรรคในการดำเนินงาน               |
| 5.3            | ข้อเสนอแนะ                                  |
| 5.4            | สิ่งที่ผู้จัดทำได้รับในการพัฒนาโครงการ      |

## สารบัญรูปภาพ

| รูปภาพ         |  | หน้า |
|----------------|--|------|
| รูปภาพที่ 2-1  | ไฟล์ report                                | 9    |
| รูปภาพที่ 3-1  | การเลือกใช้งานฟังก์ชัน Book                | 3    |
| รูปภาพที่ 3-2  | เมนู Book                                  | 4    |
| รูปภาพที่ 3-3  | การเลือกใช้งานฟังก์ชันของ members          | 4    |
| รูปภาพที่ 3-4  | เมนู Members                               | 4    |
| รูปภาพที่ 3-5  | การเลือกใช้งานฟังก์ชันของ Loans            | 5    |
| รูปภาพที่ 3-6  | เมนูของ Loans                              | 5    |
| รูปภาพที่ 3-7  | การเลือกใช้งานฟังก์ชันของ Loans            | 6    |
| รูปภาพที่ 3-8  | การเลือกใช้งานฟังก์ชันของ Exit             | 6    |
| รูปภาพที่ 3-9  | การเลือกใช้งานฟังก์ชันของ Add Book         | 7    |
| รูปภาพที่ 3-10 | การเพิ่มหนังสือ                            | 7    |
| รูปภาพที่ 3-11 | การเลือกใช้งานฟังก์ชันของ Add Members      | 7    |
| รูปภาพที่ 3-12 | การเพิ่มผู้ใช้                             | 8    |
| รูปภาพที่ 3-13 | การเลือกใช้งานฟังก์ชัน View All Book       | 8    |
| รูปภาพที่ 3-14 | แสดงข้อมูล View All Book                   | 8    |
| รูปภาพที่ 3-15 | การเลือกใช้งานฟังก์ชันของ View All Members | 9    |
| รูปภาพที่ 3-16 | แสดงข้อมูล View All Members                | 9    |
| รูปภาพที่ 3-17 | การเลือกใช้งานฟังก์ชัน View All Loans      | 9    |
| รูปภาพที่ 3-18 | แสดงข้อมูล View All Loans                  | 10   |
| รูปภาพที่ 3-19 | การเลือกใช้งานฟังก์ชัน Edit All Book       | 10   |
| รูปภาพที่ 3-20 | การแก้ไขข้อมูล Edit All Book               | 11   |
| รูปภาพที่ 3-21 | การเลือกใช้งานฟังก์ชัน Edit All Members    | 11   |
| รูปภาพที่ 3-22 | การแก้ไขข้อมูล Edit All Members            | 12   |
| รูปภาพที่ 3-23 | การเลือกใช้งานฟังก์ชัน Delete Members      | 12   |

|                |                                       |    |
|----------------|---------------------------------------|----|
| รูปภาพที่ 3-24 | การลบข้อมูล Delete Book               | 13 |
| รูปภาพที่ 3-25 | การเลือกใช้งานฟังก์ชัน Delete Members | 13 |
| รูปภาพที่ 3.26 | การลบข้อมูล Delete Members            | 14 |
| รูปภาพที่ 3.27 | การเลือกใช้งานฟังก์ชัน Borrow Book    | 14 |

## สารบัญรูปภาพ (ต่อ)

| รูปภาพ        |                                      | หน้า |
|---------------|--------------------------------------|------|
| รูปที่ 3 - 28 | การยืมหนังสือ                        | 15   |
| รูปที่ 3 - 29 | การเลือกใช้ฟังก์ชัน Return Book      | 15   |
| รูปที่ 3 - 30 | การคืนหนังสือ                        | 16   |
| รูปที่ 3 - 31 | การเลือกใช้งานฟังก์ชัน Current Loans | 16   |
| รูปที่ 3 - 32 | แสดงการยืมหนังสือ                    | 18   |
| รูปที่ 4 - 1  | Code Module pickle                   | 18   |
| รูปที่ 4 - 2  | Code Module time                     | 18   |
| รูปที่ 4- 3   | โครงสร้างข้อมูล book struck          | 19   |
| รูปที่ 4 - 4  | ฟังก์ชัน add_book                    | 19   |
| รูปที่ 4 - 5  | โครงสร้างข้อมูล members struck       | 20   |
| รูปที่ 4 - 6  | ฟังก์ชัน add_members                 | 20   |
| รูปที่ 4 - 7  | โครงสร้างข้อมูล loans struck         | 21   |
| รูปที่ 4 - 8  | ฟังก์ชัน add_loans                   | 22   |
| รูปที่ 4 - 9  | ฟังก์ชัน read_all_books              | 23   |
| รูปที่ 4 - 10 | ฟังก์ชัน read_all_members            | 22   |
| รูปที่ 4 - 11 | ฟังก์ชัน read_all_loans              | 26   |
| รูปที่ 4 - 13 | ฟังก์ชัน menu_add_book               | 25   |
| รูปที่ 4 - 14 | menu_delete_book                     | 26   |
| รูปที่ 4 - 15 | ฟังก์ชัน menu_view_books             | 27   |
| รูปที่ 4 - 16 | ฟังก์ชัน menu_edit_book              | 28   |
| รูปที่ 4 - 17 | ฟังก์ชัน menu_add_member             | 29   |
| รูปที่ 4 - 18 | ฟังก์ชัน menu_view_members           | 29   |
| รูปที่ 4 - 19 | ฟังก์ชัน menu_edit_member            | 30   |
| รูปที่ 4 - 20 | ฟังก์ชัน menu_delete_member          | 31   |
| รูปที่ 4 - 21 | ฟังก์ชัน get_current_loans           | 32   |

|               |                                  |    |
|---------------|----------------------------------|----|
| รูปที่ 4 – 22 | ฟังก์ชัน menu_borrow_book        | 33 |
| รูปที่ 4 – 23 | ฟังก์ชัน menu_return_book        | 34 |
| รูปที่ 4 – 24 | ฟังก์ชัน menu_view_all_loans     | 35 |
| รูปที่ 4 – 25 | ฟังก์ชัน menu_view_current_loans | 35 |



## สารบัญรูปภาพ(ต่อ)

| รูปภาพ        |  | หน้า |
|---------------|--|------|
| รูปที่ 4 - 26 | ฟังก์ชัน generate_report .ใช้ข้อมูลสร้างรายงานสรุป | 36   |
| รูปที่ 4 - 27 | ฟังก์ชัน generate_report .ใช้สร้างไฟล์             | 37   |
| รูปที่ 4 - 28 | Main_menu  | 38   |
| รูปที่ 4 - 29 | เมนู Book  | 39   |
| รูปที่ 4 - 30 | เมนู Members                                       | 40   |
| รูปที่ 4 - 31 | เมนู Loans   | 40   |

## สารบัญตาราง

| ตาราง                   | หน้า |
|-------------------------|------|
| ตารางที่ 2.1            |      |
| เพิ่มข้อมูลหนังสือ      | 3    |
| ตารางที่ 2.2            |      |
| เพิ่มข้อมูลสมาชิก       | 5    |
| ตารางที่ 2.3            |      |
| เพิ่มข้อมูลการยืม - คืน | 8    |

## บทที่ 1

### บทนำ

#### 1.1 วัตถุประสงค์

- 1.1.1 เพื่อพัฒนาระบบยืม - คืนหนังสือห้องสมุดได้อย่างมีประสิทธิภาพ
- 1.1.2 เพื่อฝึกฝนทักษะการเขียนโปรแกรมด้วยภาษา Python
- 1.1.3 เพื่อเรียนรู้วิธีการจัดการไฟล์และข้อมูล
- 1.1.4 เพื่อให้ฝึกการเรียนรู้การทำงานเป็นทีม

#### 1.2 ขอบเขตของการดำเนินงาน

- 1.2.1 ระบบยืม - คืนหนังสือห้องสมุดมีฟังก์ชันพื้นฐานทั้งหมด 15 ฟังก์ชัน เช่น
  - 1. เพิ่มหนังสือ
  - 2. แก้ไขหนังสือ
  - 3. ดูข้อมูลหนังสือ
  - 4. ลบหนังสือ
  - 5. กลับไปที่เมนู
  - 6. เพิ่มสมาชิก
  - 7. ลบสมาชิก
  - 8. แก้ไขสมาชิก
  - 9. แสดงสมาชิกทั้งหมด
  - 10. แสดงข้อมูลการยืม
  - 11. แสดงข้อมูลการยืมปัจจุบัน
  - 12. กลับไปที่เมนู
  - 13. แสดงข้อมูลการคืน
  - 14. เมนูกลางระบบยืม - คืนหนังสือห้องสมุด
  - 15. เมนูออกจากหน้าปัจจุบัน

1.2.2 ระบบยืม - คินหนังสือห้องสมุดประกอบด้วย 4 ไฟล์ ได้แก่

1. เพิ่มข้อมูลหนังสือ books.dat
2. เพิ่มข้อมูลสมาชิก member.dat
3. เพิ่มข้อมูลการยืม-คืน loans.dat
4. ไฟล์ report.txt

1.2.3 ระบบยืม - คินหนังสือห้องสมุดมีการจัดเก็บข้อมูลหนังสือไว้ใน Text file ชื่อ report ซึ่งมี รหัสหนังสือ ชื่อหนังสือ ชื่อผู้เขียน ปีที่เขียน ชื่อผู้ยืม จำนวนหนังสือทั้งหมด รายการผู้ยืม สถานการณ์ยืม จำนวนหนังสือถูกยืม จำนวนหนังสือที่เหลือให้ยืม ยอดสรุปทั้งหมด

1.2.4 ระบบยืม - คินหนังสือห้องสมุดจะมีเมนูให้ผู้ใช้เลือก ดำเนินการได้

### 1.3 ประโยชน์ที่ได้รับ

- 1.3.1 พัฒนาระบบที่สามารถทำการยืม - คินหนังสือได้อย่างมีประสิทธิภาพ
- 1.3.2 พัฒนาทักษะการเขียนโปรแกรม
- 1.3.3 เรียนรู้การจัดการข้อมูลและไฟล์
- 1.3.4 เรียนรู้การทำงานรวมกันเป็นทีม

### 1.4 เครื่องมือที่คิดว่าจะได้ใช้

- 1.4.1 โปรแกรม Visual Studio Code
- 1.4.2 Microsoft Office

## บทที่ 2

### ระบบยืม - คืนหนังสือห้องสมุด

#### 2.1 เพิ่มข้อมูลหนังสือ books.dat

เพิ่มข้อมูลหนังสือประกอบด้วย 8 ฟิลด์หลัก ซึ่งแต่ละฟิลด์มีรายละเอียดและความสำคัญดังนี้

| # | ฟิลด์            | ชนิด   | ขนาด | ตัวอย่าง             |
|---|------------------|--------|------|----------------------|
| 1 | book_id          | int    | 4    | รหัสหนังสือ          |
| 2 | Title            | string | 20   | ชื่อหนังสือ          |
| 3 | Author           | string | 20   | ชื่อผู้แต่ง          |
| 4 | Year             | int    | 4    | ปีที่พิมพ์           |
| 5 | Copies           | int    | 4    | จำนวนเล่มทั้งหมด     |
| 6 | Available_copies | int    | 4    | จำนวนเล่มที่พร้อมยืม |

ตารางที่ 2.1 เพิ่มข้อมูลหนังสือ

##### 2.1.1 book\_id รหัสหนังสือ

เป็นรหัสหนังสือที่ใช้ในการระบุหนังสือแต่ละเล่มอย่างชัดเจนและไม่ซ้ำกันฟิลด์นี้ถูกสร้างขึ้นโดยระบบในรูปแบบของตัวเลข (integer) เช่น 1001, 1002, 1003 เป็นต้น การมี รหัสหนังสือที่เป็นเอกลักษณ์นี้เป็นสิ่งจำเป็นเพื่อหลีกเลี่ยงความสับสนระหว่างหนังสือหลายเล่ม และช่วยให้สามารถค้นหาและเรียกดูข้อมูลของหนังสือได้อย่างแม่นยำและรวดเร็ว โดยเฉพาะในกรณีที่มีหนังสือจำนวนมาก

### 2.1.2 title ชื่อหนังสือ

title คือ ชื่อเต็มของหนังสือแต่ละเล่ม ซึ่งฟิลด์นี้จะแสดงข้อมูลชื่อหนังสือแต่ละเล่มของห้องสมุด ฟิลด์นี้เป็นประเภทข้อมูลข้อความ (string) ตัวอย่างเช่น " PATRIOT " หรือ " COMPUTER Python " การมีชื่อหนังสือในระบบมีความสำคัญอย่างยิ่ง เพราะใช้ในการเรียกดูข้อมูล ตรวจสอบการยืม - คืน และทำการแก้ไขข้อมูลต่างๆ หนังสือแต่ละเล่มจะมีชื่อตามที่ระบุในการลงทะเบียน และระบบจะใช้ชื่อดังกล่าวในการค้นหาและแสดงผลข้อมูลที่เกี่ยวข้องกับหนังสือเล่มนั้น

### 2.1.3 Author ชื่อผู้แต่ง

Author คือ ชื่อของผู้เขียนหนังสือเล่มนั้นๆ ซึ่งฟิลด์นี้จะแสดงข้อมูลชื่อผู้เขียนหนังสือแต่ละเล่ม ฟิลด์นี้เป็นประเภทข้อมูลข้อความ ( string) ตัวอย่างเช่น " ALEX LAYER " หรือ " ANDEL P. SOLOS " การมีชื่อผู้เขียนหนังสือในระบบมีความสำคัญอย่างยิ่ง เพราะใช้ในการเรียกดูข้อมูล ตรวจสอบหนังสือ การยืม - คืน และทำการแก้ไขข้อมูลต่างๆ หนังสือแต่ละเล่มจะมีชื่อตามที่ระบุในการลงทะเบียน และ ระบบจะใช้ชื่อดังกล่าวในการค้นหาและแสดงผลข้อมูลที่เกี่ยวข้องกับหนังสือเล่มนั้น author คือ ชื่อของผู้เขียนหนังสือเล่มนั้นๆ ซึ่งฟิลด์นี้จะแสดงข้อมูลชื่อผู้เขียนหนังสือแต่ละเล่ม ฟิลด์นี้เป็นประเภทข้อมูลข้อความ ( string) ตัวอย่างเช่น " ALEXEI NAVALNY " หรือ "RANDAL E. BRYANT " การมีชื่อผู้เขียนหนังสือในระบบมีความสำคัญอย่างยิ่ง เพราะใช้ในการเรียกดูข้อมูล ตรวจสอบหนังสือ การยืม - คืน และทำการแก้ไขข้อมูลต่างๆ หนังสือแต่ละเล่มจะมีชื่อตามที่ระบุในการลงทะเบียน และ ระบบจะใช้ชื่อดังกล่าวในการค้นหาและแสดงผลข้อมูลที่เกี่ยวข้องกับหนังสือเล่มนั้น

### 2.1.4 Year ปีที่พิมพ์

Year คือ ปีที่เขียนหนังสือที่ใช้ในการระบุหนังสือแต่ละเล่มอย่างชัดเจน ฟิลด์นี้ถูกสร้างขึ้นโดยระบบในรูปแบบของตัวเลข (integer) เช่น 2025, 2019, 2016 เป็นต้น การมีปีที่เขียนหนังสือที่เป็นเอกลักษณ์นี้เป็นสิ่งจำเป็นเพื่อแสดงปีที่เขียนหนังสือเล่มนั้นๆ และช่วยให้สามารถทราบปีที่เขียนได้และทราบระยะเวลาของหนังสือเพื่อเพิ่มความน่าเชื่อถือได้มากยิ่งขึ้น

### 2.1.5 Copies จำนวนเล่ม

Copies ใช้เก็บจำนวนเล่มของหนังสือที่มีอยู่ในห้องสมุดหรือคลังหนังสือ ฟิลด์นี้เก็บในรูปแบบตัวเลขจำนวนเต็ม (integer) เช่น 3, 10, 25 การเก็บจำนวนเล่มจะช่วยในการบริหารจัดการทรัพยากร เช่น การยืม-คืน เป็นต้น

### 2.1.7 Available\_copies

Available\_copies ใช้ระบุว่าจำนวนเล่มหนังสือที่พร้อมยืมจำนวนเท่าใดเพื่อให้ผู้ยืมสามารถรู้ได้ ฟิลด์นี้เก็บข้อมูลเป็น (integer) เช่น 8, 9, เป็นต้น การบอกจำนวนเล่มที่พร้อมยืมสามารถทำให้เราทราบได้ง่ายอำนวยความสะดวก เป็นการบอกไปในตัวว่าหนังสือมีครบพร้อมสำหรับการยืมไหม เป็นต้น

## 2.2 เพิ่มข้อมูลสมาชิก members.dat

เพิ่มข้อมูลสมาชิกประกอบด้วย 7 ฟิลด์หลัก ซึ่งแต่ละฟิลด์มีรายละเอียดและความสำคัญดังนี้

| # | ฟิลด์         | ชนิด   | ขนาด | ตัวอย่าง   |
|---|---------------|--------|------|--|
| 1 | member_id     | int    | 4    | 4001   |
| 2 | Name          | string | 20   | "Kanye west"   |
| 3 | Email address | string | 20   | <a href="mailto:Kanye@example.com">"Kanye@example.com"</a> |
| 4 | phone         | string | 10   | "0877494911"   |
| 5 | Major         | string | 30   | Chemical   |
| 6 | Join_Year     | int    | 4    | 2020   |
| 7 | Active        |        |      | Yes/no   |

ตารางที่ 2.1 เพิ่มข้อมูลสมาชิก

### 2.2.1 Member\_id รหัสสมาชิก

Member\_id เป็นรหัสที่ใช้ระบุสมาชิกแต่ละคนในระบบ ฟิลด์นี้เป็นตัวเลขจำนวนเต็ม (integer) มีความเป็นเอกลักษณ์ไม่ซ้ำกัน เช่น 10001, 4001 การมีรหัสสมาชิกช่วยให้ระบบสามารถจัดการข้อมูลสมาชิกจำนวนมากได้อย่างถูกต้องและรวดเร็ว

### 2.2.2 Name ชื่อเต็ม

Name ใช้เก็บชื่อที่ใช้งานของสมาชิก ในรูปแบบข้อมูลตัวอักษรข้อความ (string) เช่น Kanye ฟิลด์นี้ช่วยให้ระบบสามารถจัดการแยกแยะสมาชิกได้และเป็นข้อมูลเบื้องต้นของผู้ใช้ ทำให้ทราบชื่อได้ เพื่อระบุตัวตนของสมาชิกด้วย

### 2.2.5 email อีเมล

email เป็นตัวอักษร ข้อความ(string) ฟิลด์นี้ใช้เก็บ ช่องทางติดต่ออื่นๆหรือเพื่อ ส่งข่าวสาร ส่งข้อมูลให้สมาชิกรับทราบต่างๆ

### 2.2.4 phone เบอร์ติดต่อ

phone เป็นตัวเลขจำนวนเต็ม (integer) ใช้เก็บเบอร์โทรของสมาชิก เช่น 0877494911 ข้อมูลนี้อาจถูกนำไปใช้ในติดต่อกับสมาชิก

### 2.2.5 Major สาขาวิชาเอก

Major เป็นตัวอักษร ข้อความ(string) ฟิลด์นี้ใช้สำหรับเก็บข้อมูล/แสดงข้อมูลที่บ่งบอกว่า สาขาวิชาเอกอะไร

### 2.2.6 Year ปีที่เข้าร่วมเป็นสมาชิก

Year ใช้เก็บข้อมูลปีที่เข้าร่วมของสมาชิก โดยฟิลด์นี้เก็บข้อมูลเป็น (string) เพื่อให้ทราบถึงการเข้าร่วมเป็นสมาชิกของแต่ละ members



### 2.2.5 Active การใช้งาน

Major เป็นตัวอักษร ข้อความ(string) ฟิลด์นี้ใช้สำหรับแสดงข้อมูลที่บ่งบอกว่าได้ใช้งานอยู่ ใช่/ไม่ เพื่อให้ทราบว่าผู้ใช้ยังใช้งานอยู่

## 2.3 เพิ่มข้อมูลการยืม-คืน borrow.dat

เพิ่มข้อมูลนี้ประกอบไปด้วย 8 ฟิลด์หลัก ซึ่งแต่ละฟิลด์มีรายละเอียดดังนี้

| # | ฟิลด์       | ชนิด      | ขนาด | ตัวอย่าง |
|---|-------------|-----------|------|----------|
| 1 | borrow_id   | int       | 4    | 80001    |
| 2 | member_id   | int       | 4    | 4001     |
| 3 | book_id     | int       | 4    | 1001     |
| 4 | borrow_date | int       | 4    | 20250901 |
| 5 | due_date    | int       | 4    | 20250910 |
| 6 | return_date | int       | 4    | 0        |
| 7 | is_returned | int (1/0) | 4    | 0        |
| 8 | ts          | int       | 4    | 0        |

ตารางที่ 2.3 เพิ่มข้อมูลการยืม-คืน

### 2.3.1 borrow\_id

Borrow\_id ใช้เก็บข้อมูล แสดงรหัสการยืมของสมาชิก มีฟิลด์เป็น (integer) ตัวอย่าง เช่น “80001, 90001” เป็นต้น เป็นฟิลด์จำนวนเต็ม สามารถดูได้ว่า ไอเดียการยืมนี้ คือ สมาชิกคนใด ยืมอะไรไป กำหนดการคืนวันไหน คืนหรือยัง

### 2.3.2 member\_id รหัสสมาชิก (อ้างอิงไปยัง members.dat)

เป็นฟิลด์จำนวนเต็ม (integer 4 bytes) ใช้เก็บรหัสสมาชิกที่เกี่ยวข้องกับเหตุการณ์ โดยอ้างอิง ไปยังข้อมูลในไฟล์ members.dat ช่วยให้ทราบว่าใครเป็นผู้ทำการยืม-คืน

### 2.3.3 book\_id รหัสหนังสือ (อ้างอิงไปยัง books.dat)

book id เป็นรหัสหนังสือที่ใช้ในการระบุหนังสือแต่ละเล่มอย่างชัดเจนและไม่ซ้ำกันฟิลด์นี้ถูกสร้างขึ้นโดยระบบในรูปแบบของตัวเลข (integer) เช่น 1001, 1002, 1003 เป็นต้นการมีรหัสหนังสือที่เป็นเอกลักษณ์นี้เป็นสิ่งจำเป็นเพื่อหลีกเลี่ยงความสับสนระหว่างหนังสือหลายเล่มและช่วยให้สามารถค้นหาและเรียกดูข้อมูลของหนังสือได้อย่างแม่นยำและรวดเร็วโดยเฉพาะในกรณีที่มีหนังสือจำนวนมาก

### 2.3.4 borrow\_date วันที่ยืม

เป็นฟิลด์ข้อความ (string ขนาด 10 ตัวอักษร) ใช้เก็บวันที่ยืมหนังสือในรูปแบบ YYYY-MM-DD เช่น 2025-09-30 ข้อมูลนี้ใช้ตรวจสอบว่าหนังสือถูกยืมไปตั้งแต่เมื่อใด

### 2.3.5 return\_date วันที่คืน

เป็นฟิลด์ข้อความ (string ขนาด 10 ตัวอักษร) ใช้เก็บวันที่คืนหนังสือในรูปแบบ YYYY-MM-DD เช่น 2025-10-05 หากยังไม่ได้คืนค่าของฟิลด์นี้อาจเป็นค่าว่าง

### 2.3.6 is\_returned วันที่คืนจริง

เป็นฟิลด์จำนวนเต็ม(integer 4 bytes) ใช้เก็บสถานะของรายการหลังจากเหตุการณ์เกิดขึ้น เช่น 1 = ยืมอยู่ 0 = คืนแล้ว ฟิลด์นี้ช่วยบันทึกผลลัพธ์ของเหตุการณ์ยืมหรือคืนอย่างชัดเจน

### 2.3.7 ts timestamp เหตุการณ์ (เวลาที่เกิดการยืมหรือคืน)

เป็นฟิลด์แบบ timestamp (integer 4 bytes) ใช้เก็บวันที่และเวลาเกิดเหตุการณ์การ ยืม-คืน หนังสือข้อมูลนี้ช่วยให้สามารถบันทึกลำดับเวลาของเหตุการณ์ได้อย่างถูกต้อง และใช้ ตรวจสอบย้อนหลังได้

## 2.4 ไฟล์ report.txt

ไฟล์ report.txt ในระบบยืม - คืบหนังสือห้องสมุดของคุณประกอบด้วย 11 필ด์หลัก ซึ่ง แต่ละ 필ด์มีรายละเอียดและความสำคัญดังนี้

```

===== Main Menu =====
1) Member
2) Book
3) Loan
4) Report
0) Exit
Choose: 4

=== Report (Joined Only) ===
Save to file? (y/N): n

Library Management System - Lending Report
Generated At : 2025-10-05 14:42
App Version : 1.0
Encoding : UTF-8

=====
| LoanID | MemberID | Member Name | Email | BookID | Book Title | Author | Loan Date | Due Date | Return Date | Status |
=====
| LN175939679 | M001 | Anan Phonchai | anan@gmail.com | B002 | Data Structures in C | A. Kumar | 2025-10-02 | 2025-10-09 | 2025-10-05 | returned |
| LN1759396726 | M001 | Anan Phonchai | anan@gmail.com | B001 | Python Programming Basics | J. Smith | 2025-10-02 | 2025-10-09 | - | borrowed |
| LN1759649966 | M005 | Kanya Srichai | kanya@example.com | B003 | Electrical Circuits | T. Lee | 2025-10-05 | 2025-10-06 | 2025-10-05 | returned |
| LN1759649988 | M003 | Nida Wongpras | nida@gmail.com | B004 | Civil Engineering Handbook | W. Johnson | 2025-10-05 | 2025-10-07 | - | borrowed |
| LN1759650074 | M002 | Somchai Boonmee | somchai@gmail.com | B003 | Electrical Circuits | T. Lee | 2025-10-05 | 2025-10-09 | - | borrowed |
=====

Summary
- Total Lendings : 5
- Currently Borrowed: 3
- Overdue : 0
- Active Members : 5

===== Main Menu =====
1) Member
2) Book
3) Loan
4) Report
0) Exit
Choose:

```

รูปภาพที่ 2-1 ไฟล์ report

### 2.4.1 header\_text ส่วนหัวรายงาน

เป็นฟิลด์ข้อความ (string 100 bytes) ใช้เก็บข้อความส่วนหัวของรายงาน เช่น "LibraryBorrow System – Summary Report" ฟิลด์นี้แสดงชื่อหรือวัตถุประสงค์ของรายงาน เพื่อให้ผู้ใช้เข้าใจว่าเป็นรายงานประเภทใด

### 2.4.2 generated\_at วันที่และเวลาที่สร้างรายงาน (YYYY-MM-DD HH:MM)

เป็นฟิลด์ข้อความ (string 25 bytes) ใช้เก็บวันและเวลาที่รายงานถูกสร้างขึ้นในรูปแบบ YYYY-MM-DD HH:MM เช่น "2025-10-05 14:42" ฟิลด์นี้ช่วยในการติดตามและอ้างอิงว่าไฟล์รายงานถูกสร้างเมื่อ

#### 2.4.3 app\_version เวอร์ชันโปรแกรม เช่น "1.0"

เป็นฟิลด์ข้อความ (string 10 bytes) ใช้เก็บหมายเลขเวอร์ชันของโปรแกรมที่สร้างรายงาน เช่น "1.0", "2.1.5" ฟิลด์นี้มีประโยชน์ในการตรวจสอบว่าไฟล์รายงานถูกสร้างด้วยเวอร์ชันของระบบใด

#### 2.4.4 encoding การเข้ารหัสไฟล์

เป็นฟิลด์ข้อความ (string 20 bytes) ใช้ระบุรูปแบบการเข้ารหัสไฟล์ เช่น "UTF-8", "ISO-8859-1" เพื่อให้การอ่านไฟล์รายงานถูกต้องตรงกับภาษาที่ใช้งาน

#### 2.4.5 book\_table\_header หัวตาราง

book\_table\_header เป็นฟิลด์ข้อความ (string 80 bytes) ใช้เก็บหัวตารางสำหรับแสดงข้อมูลหนังสือ เช่น "LoansID | MemberID | Member Name | Email | BookID | Book Title | Author | Loans Date | Due Date | Return Date | Status " เพื่อกำหนดโครงสร้างของตารางในส่วนรายงาน

#### 2.4.6 book\_records ข้อมูลตาราง

เป็นฟิลด์ข้อความ (string  $120 * N$  bytes, โดย  $N$  = จำนวนหนังสือ) ใช้เก็บข้อมูลหนังสือแต่ละเล่มในรูปแบบความยาวคงที่ (fixed-length record) เช่น รายการรหัสหนังสือ, ชื่อเรื่อง, ผู้แต่ง, ปีพิมพ์, จำนวนเล่ม สมาชิกที่ยืมหนังสือ และสถานะหนังสือ

#### 2.4.6 summary\_section สรุปข้อมูล

เป็นฟิลด์ข้อความ (string 150 bytes) ใช้เก็บข้อมูลสรุปของระบบ เช่น Total Lending = จำนวนการยืมหนังสือทั้งหมดในระบบ Currently = หนังสือที่ใช้งานอยู่(ยังไม่ถูกคืนในระบบ) Overdue = หนังสือที่ยืมเกินกำหนด Active Members = จำนวนสมาชิกที่มีการยืมอยู่ในปัจจุบัน ฟิลด์นี้มีประโยชน์ต่อการวิเคราะห์พฤติกรรมการใช้งานของสมาชิกและการวางแผนจัดการหนังสือ

### บทที่ 3

#### การใช้งานระบบยืม – คืนหนังสือห้องสมุด

โปรแกรมการยืม – คืนหนังสือห้องสมุดคือการช่วยการยืม – คืนหนังสือให้สะดวกและง่ายขึ้น และยังช่วยจัดประเภทของหนังสือให้ดูง่ายขึ้นโดยการช่วยทำรายงานไปเก็บไว้ยังไฟล์Text/dat

โปรแกรมการยืม – คืนหนังสือห้องสมุดประกอบไปด้วย การเพิ่มข้อมูลที่ จะเก็บข้อมูล Book ID, Title, Author, Year , copies, Status แสดงข้อมูลหนังสือทั้งหมด ในโปรแกรมค้นหาข้อมูลโดยใช้ Book ID และ Title ในการค้นหาอัปเดตข้อมูลสามารถเปลี่ยนแปลงข้อมูลได้ แต่ถ้าไม่ต้องการแก้ไขข้อมูลบางส่วนสามารถกด Enter เพื่อข้ามการเปลี่ยนแปลงในข้อมูลนั้นๆ ได้เลยลบข้อมูลโดยการใช้เลข Book ID เพื่อลบข้อมูลทั้งหมดของเลข Book ID นั้น สร้างรายงานเพื่อทำสรุปการยืม – คืนหนังสือที่มีข้อมูลทั้งหมดในโปรแกรมจบการทำงานของโปรแกรม

สำหรับผู้ใช้งานโปรแกรม

#### 3.1 การใช้งานโปรแกรมระบบยืม – คืนห้องสมุด

3.1.1 กรอกรหัสเลข 2 ภายในช่อง Choose เพื่อเรียกใช้ฟังก์ชัน Book จะแสดงข้อมูลที่ประกอบไปด้วย Add, View, Edit, Delete, Back

```
===== Main Menu =====
1) Member
2) Book
3) Loan
4) Report
0) Exit
Choose: 2
```

รูปภาพ การเลือกใช้งานฟังก์ชัน

3.1.2 เมื่อเมนูฟังก์ชัน Book ขึ้นมาแล้วจากนั้นก็สามารัระบุเมนูที่ต้องการเลือกได้

```

===== Book Menu =====
1) Add
2) View
3) Edit
4) Delete (hard)
0) Back
Choose: █

```

รูปภาพเมนูของ Book

3.1.3 กรอกหมายเลข 1 ภายในช่อง Choose เพื่อเรียกใช้ฟังก์ชัน Member  
จะแสดงข้อมูลที่ประกอบไปด้วย Add, View, Edit, Delete, Back

```

===== Main Menu =====
1) Member
2) Book
3) Loan
4) Report
0) Exit
Choose:

```

รูปภาพ การเลือกใช้ฟังก์ชันของ Member

3.1.4 เมื่อฟังก์ชัน Member ขึ้นมาแล้วจากนั้นก็สมารถระบุเมนูที่ต้องการเลือกได้

```

===== Member Menu =====
1) Add
2) View
3) Edit
4) Delete (soft: active=False)
0) Back
Choose: █

```

รูปภาพ เมนูของ Member

3.1.5 กรอกหมายเลข 3 ภายในช่อง Choose เพื่อเรียกใช้ฟังก์ชัน Loan  
จะแสดงข้อมูลที่ประกอบไปด้วย Borrow, Return, View, Back

```

===== Main Menu =====
1) Member
2) Book
3) Loan
4) Report
0) Exit
Choose:

```

รูปภาพ การเรียกใช้ฟังก์ชันของ Loan

3.1.6 เมื่อเมนูฟังก์ชัน Loan ขึ้นมาแล้วจากนั้นก็สมารถระบุเมนูที่ต้องการเลือกได้

```

===== Loan Menu =====
1) Borrow
2) Return
3) View
0) Back
Choose: 

```

รูปภาพ เมนูของ Loan

3.1.7 กรอกหมายเลข 4 ภายในช่อง Choose เพื่อเรียกใช้ฟังก์ชัน Loan เพื่อเพิ่มไฟล์ report.txt ที่สามารถเขียน report ถึงห้องสมุดได้

```

===== Main Menu =====
1) Member
2) Book
3) Loan
4) Report
0) Exit
Choose: 4

```

รูปภาพ การเลือกใช้งานฟังก์ชันของ Report

3.1.8 กรอกหมายเลข 5 ภายในช่อง Choose เพื่อเรียกใช้ฟังก์ชัน Exit เพื่อออกจากโปรแกรม

```
===== Main Menu =====
1) Member
2) Book
3) Loan
4) Report
0) Exit
Choose: 5
Invalid option
```

รูปภาพ การเรียกใช้งานฟังก์ชัน Exit

## 3.2 การใช้งานโปรแกรมเพิ่มข้อมูล

3.2.1 เปิดโปรแกรมมาจะเจอหน้าเมนูให้กด 1 เพื่อเข้าสู่หน้า Member

```
===== Main Menu =====
1) Member
2) Book
3) Loan
4) Report
0) Exit
Choose:
```

3.2.2 จะเจอหน้า Member ให้กด 1 เพื่อเลือกเมนู Add

```
===== Member Menu =====
1) Add
2) View
3) Edit
4) Delete (soft: active=False)
0) Back
Choose: █
```



3.2.3 เข้าสู่หน้าเพิ่มสมาชิกให้กรอกข้อมูลให้ครบแล้วกด Enter เป็นอันเสร็จ

```
=== Member > Add ===
Member ID: M006
First name: Kittithat
Last name: Khemnak
Email: kittikhem@gmail.com
Phone: 0867890123
Major (optional): Computer
Year (0-255, optional): 21
Added.
```

### 3.3 ใช้งานโปรแกรมแสดงข้อมูล

3.3.1 เปิดโปรแกรมมาจะเจอหน้าเมนูให้กด 1 เพื่อเข้าสู่หน้า Member

```
===== Main Menu =====
1) Member
2) Book
3) Loan
4) Report
0) Exit
Choose:
```

3.3.2 จะเจอหน้า Member ให้กด 2 เพื่อเลือกเมนู View

```
===== Member Menu =====
1) Add
2) View
3) Edit
4) Delete (soft: active=False)
0) Back
Choose: █
```

### 3.3.3 โปรแกรมจะแสดงรายละเอียดของ Member ทั้งหมดมาให้

```
=== Member > View ===
```

| MemberID | Name              | Email               | Phone      | Major      | Year | Active |
|----------|-------------------|---------------------|------------|------------|------|--------|
| M001     | Anan Phonchai     | anan@gmail.com      | 0812345678 | Computer   | 2    | Yes    |
| M002     | Somchai Boonmee   | somchai@gmail.com   | 0823456789 | Electrical | 3    | Yes    |
| M003     | Nida Wongpras     | nida@gmail.com      | 0834567890 | Mechanical | 1    | Yes    |
| M004     | Prasert Thongdee  | prasert@gmail.com   | 0845678901 | Civil      | 1    | Yes    |
| M005     | Kanya Srichai     | kanya@example.com   | 0856789012 | Chemical   | 2    | Yes    |
| M006     | Kittithat Khemnak | kittikhem@gmail.com | 0867890123 | Computer   | 21   | Yes    |

Press Enter to continue...

## 3.4 การใช้งานโปรแกรมแก้ไขข้อมูล

### 3.4.1 เปิดโปรแกรมมาจะเจอหน้าเมนูให้กด 1 เพื่อเข้าสู่หน้า Member

```
===== Main Menu =====
1) Member
2) Book
3) Loan
4) Report
0) Exit
Choose:
```

### 3.4.2 จะเจอหน้า Member ให้กด 3 เพื่อเลือกเมนู Edit

```
===== Member Menu =====
1) Add
2) View
3) Edit
4) Delete (soft: active=False)
0) Back
Choose: █
```

### 3.4.3 เข้าสู่หน้าแก้ไขสมาชิกให้กรอกข้อมูลที่จะแก้ไขครบแล้วกด Enter เป็นอันเสร็จ

```

=== Member > Edit ===
Member ID to edit: M006
Leave blank to keep current.
First name [Kittithat]: Ethan
Last name [Khemnak]: cole
Email [kittikhem@gmail.com]: Ethacol@gmail.com
Phone [0867890123]: 0887514758
Major [Computer]: Main
Year [21]: 20
Active true/false [True]: true
Updated.

Press Enter to continue...

```

### 3.4.4 เช็กผลการแก้ไขได้ที่เมนู View

| MemberID | Name             | Email             | Phone      | Major      | Year | Active |
|----------|------------------|-------------------|------------|------------|------|--------|
| M001     | Anan Phonchai    | anan@gmail.com    | 0812345678 | Computer   | 2    | Yes    |
| M002     | Somchai Boonmee  | somchai@gmail.com | 0823456789 | Electrical | 3    | Yes    |
| M003     | Nida Wongpras    | nida@gmail.com    | 0834567890 | Mechanical | 1    | Yes    |
| M004     | Prasert Thongdee | prasert@gmail.com | 0845678901 | Civil      | 1    | Yes    |
| M005     | Kanya Srichai    | kanya@example.com | 0856789012 | Chemical   | 2    | Yes    |
| M006     | Ethan cole       | Ethacol@gmail.com | 0887514758 | Main       | 20   | Yes    |

## 3.5 การใช้งานโปรแกรมลบข้อมูล

### 3.5.1 เปิดโปรแกรมมาจะเจอหน้าเมนูให้กด 1 เพื่อเข้าสู่หน้า Member

```

===== Main Menu =====
1) Member
2) Book
3) Loan
4) Report
0) Exit
Choose:

```

### 3.5.2 จะเจอหน้า Member ให้กด 4 เพื่อเลือกเมนู Delete

```
==== Member Menu ====
1) Add
2) View
3) Edit
4) Delete (soft: active=False)
0) Back
Choose: █
```

### 3.5.3 ให้ใส่เลขที่ IDMember ที่จะลบจากนั้นกด Enter เป็นอันเสร็จ

```
=== Member > Delete (soft) ===
Member ID to deactivate: M006
Deactivated (active = False).

Press Enter to continue... █
```

## 3.6 วิธีเพิ่ม Book

### 3.6.1 เปิดโปรแกรมมาจะเจอหน้าเมนูให้กด 2 เพื่อเข้าสู่หน้า Book

```
===== Main Menu =====
1) Member
2) Book
3) Loan
4) Report
0) Exit
Choose: █
```

### 3.6.2 จะเจอหน้า Book ให้กด 1 เพื่อเลือกเมนู Add

```
==== Book Menu ====
1) Add
2) View
3) Edit
4) Delete (hard)
0) Back
Choose: █
```

### 3.6.3 เข้าสู่หน้าเพิ่ม Book ให้กรอกข้อมูลให้ครบแล้วกด Enter เป็นอันเสร็จ

```
=== Book > Add ===
Book ID: B006
Title: To Kill a Mockingbird
Author: Harper Lee
Category: Classic Literature
Year: 2503
ISBN: 9780061120084
Total copies: 6
Added.

Press Enter to continue... █
```

## 3.7 วิธีดูข้อมูล Book

### 3.7.1 เปิดโปรแกรมมาจะเจอหน้าเมนูให้กด 2 เพื่อเข้าสู่หน้า Book

```
===== Main Menu =====
1) Member
2) Book
3) Loan
4) Report
0) Exit
Choose:
```

### 3.7.2 จะเจอหน้า Book ให้กด 2 เพื่อเลือกเมนู View

```
==== Book Menu ====
1) Add
2) View
3) Edit
4) Delete (hard)
0) Back
Choose: █
```

### 3.7.3 โปรแกรมจะแสดงรายละเอียดของ Book ทั้งหมดมาให้

```
=== Book > View ===
```

| BookID | Title                          | Author     | Year | Copies | Avail |
|--------|--------------------------------|------------|------|--------|-------|
| B001   | Python Programming Basics      | J. Smith   | 2022 | 3      | 2     |
| B002   | Data Structures in C           | A. Kumar   | 2021 | 2      | 1     |
| B003   | Electrical Circuits            | T. Lee     | 2020 | 5      | 5     |
| B004   | Civil Engineering Handbook     | W. Johnson | 2019 | 4      | 4     |
| B005   | Introduction to Thermodynamics | M. Brown   | 2023 | 3      | 3     |

```
Press Enter to continue... █
```

## 3.8 วิธีแก้ไขข้อมูล Book

### 3.8.1 เปิดโปรแกรมมาเจอหน้าเมนูให้กด 2 เพื่อเข้าสู่หน้า Book

```
===== Main Menu =====
1) Member
2) Book
3) Loan
4) Report
0) Exit
Choose:
```

### 3.8.2 จะเจอหน้า Book ให้กด 3 เพื่อเลือกเมนู Edit

```
==== Book Menu ====
1) Add
2) View
3) Edit
4) Delete (hard)
0) Back
Choose: █
```

### 3.8.3 เข้าสู่หน้าแก้ไขข้อมูลหนังสือให้กรอกข้อมูลที่จะแก้ไขครบแล้วกด Enter เป็นอันเสร็จ

```
=== Book > Edit ===
Book ID to edit: B005
Leave blank to keep current.
Title [Introduction to Thermodynamics]: The Great Gatsby
Author [M. Brown]: F. Scott Gatsby
Category [Mechanical]:
Year [2023]: 1925
ISBN [9785555555]: 9780743273565
Total copies [3]: 6
Updated.

Press Enter to continue... █
```

## 3.9 วิธีลบข้อมูล Book

### 3.9.1 เปิดโปรแกรมมาจะเจอหน้าเมนูให้กด 2 เพื่อเข้าสู่หน้า Book

```
===== Main Menu =====
1) Member
2) Book
3) Loan
4) Report
0) Exit
Choose:
```

### 3.9.2 จะเจอหน้า Book ให้กด 4 เพื่อเลือกเมนู Delete

```
==== Book Menu ====
1) Add
2) View
3) Edit
4) Delete (hard)
0) Back
Choose: █
```

### 3.9.3 ให้ใส่เลขที่ IDBook ที่จะลบจากนั้นกด Enter เป็นอันเสร็จ

```
=== Book > Delete (hard) ===
Book ID to delete: B006
Deleted.

Press Enter to continue... █
```

## 4.0 วิธียืมหนังสือ

### 4.0.1. เปิดโปรแกรมมาจะเจอหน้าเมนูให้กด 3 เพื่อเข้าสู่หน้า Loan

```
===== Main Menu =====
1) Member
2) Book
3) Loan
4) Report
0) Exit
Choose: █
```



#### 4.0.2 จะเจอหน้า Loan ให้กด 1 เพื่อเลือกเมนู Borrow

```
==== Loan Menu ====
1) Borrow
2) Return
3) View
0) Back
Choose: █
```

#### 4.0.3 กรอกข้อมูลสมาชิกและข้อมูลหนังสือที่จะยืมและกด Enter เป็นอันเสร็จ

```
=== Loan > Borrow ===
Member ID: M001
Book ID: B005
Days (default 14): 5
Borrowed.

Press Enter to continue... █
```

#### 4.0.4 ดูข้อมูลการยืมได้ที่หน้า View จะแสดง LoanID ที่เรายืมไว้ใส่ตอนคืนหนังสือ

```
=== Loan > View ===
```

| LoanID       | Member Name   | BookID | Title                     | LoanDate   | DueDate    | ReturnDate | Status   |
|--------------|---------------|--------|---------------------------|------------|------------|------------|----------|
| LN1759396679 | Anan Phonchai | B002   | Data Structures in C      | 2025-10-02 | 2025-10-09 | -          | borrowed |
| LN1759396726 | Anan Phonchai | B001   | Python Programming Basics | 2025-10-02 | 2025-10-09 | -          | borrowed |
| LN1759652979 | Anan Phonchai | B005   | The Great Gatsby          | 2025-10-05 | 2025-10-10 | 2025-10-05 | returned |

```

Borrowed Today
- Anan Phonchai | LoanID: LN1759652979 | The Great Gatsby (borrow & return today)

Returned Today
- Anan Phonchai | LoanID: LN1759652979 | The Great Gatsby (same-day)

Summary today -> Borrowers: 1 | Returns: 1 | Overdue: 0

```

```

=== Loan > View ===



| LoanID       | Member Name   | BookID | Title                     | LoanDate   | DueDate    | ReturnDate | Status   |
|--------------|---------------|--------|---------------------------|------------|------------|------------|----------|
| LN1759396679 | Anan Phonchai | B002   | Data Structures in C      | 2025-10-02 | 2025-10-09 | -          | borrowed |
| LN1759396726 | Anan Phonchai | B001   | Python Programming Basics | 2025-10-02 | 2025-10-09 | -          | borrowed |
| LN1759652979 | Anan Phonchai | B005   | The Great Gatsby          | 2025-10-05 | 2025-10-10 | 2025-10-05 | returned |



Borrowed Today
- Anan Phonchai | LoanID: LN1759652979 | The Great Gatsby (borrow & return today)

Returned Today
- Anan Phonchai | LoanID: LN1759652979 | The Great Gatsby (same-day)

Summary today -> Borrowers: 1 | Returns: 1 | Overdue: 0

Choose:

```

#### 4.1.2 จะเจอน้ำ Loan ให้กด 2 เพื่อเลือกเมนู Return

```

===== Loan Menu =====
1) Borrow
2) Return
3) View
0) Back
Choose: 

```

#### 4.1.3 กรอก LoanID ที่ได้มาตอนยืมแล้วกด Enter เป็นอันเสร็จ

```

=== Loan > Return ===
Loan ID: LN1759652979
Returned.

Press Enter to continue...

```

#### 4.1.4 เช็กข้อมูลการคืนได้ที่หน้า View จะขึ้นวันที่คืนในช่อง ReturnData

## 4.2 วิธีดูข้อมูลการคืน-ยืมหนังสือ

### 4.2.1 เปิดโปรแกรมมาจะเจอหน้าเมนูให้กด 3 เพื่อเข้าสู่หน้า Loan

```
===== Main Menu =====
1) Member
2) Book
3) Loan
4) Report
0) Exit
Choose:
```

### 4.2.2 จะเจอหน้า Loan ให้กด 3 เพื่อเลือกเมนู View

```
==== Loan Menu ====
1) Borrow
2) Return
3) View
0) Back
Choose: █
```

### 4.2.3 โปรแกรมจะแสดงรายละเอียดของการคืน-ยืมทั้งหมดมาให้

```
=== Loan > View ===
```

| LoanID       | Member Name   | BookID | Title                     | LoanDate   | DueDate    | ReturnDate | Status   |
|--------------|---------------|--------|---------------------------|------------|------------|------------|----------|
| LN1759396679 | Anan Phonchai | B002   | Data Structures in C      | 2025-10-02 | 2025-10-09 | -          | borrowed |
| LN1759396726 | Anan Phonchai | B001   | Python Programming Basics | 2025-10-02 | 2025-10-09 | -          | borrowed |
| LN1759652979 | Anan Phonchai | B005   | The Great Gatsby          | 2025-10-05 | 2025-10-10 | 2025-10-05 | returned |

Borrowed Today

- Anan Phonchai | LoanID: LN1759652979 | The Great Gatsby (borrow & return today)

Returned Today

- Anan Phonchai | LoanID: LN1759652979 | The Great Gatsby (same-day)

Summary today -> Borrowers: 1 | Returns: 1 | Overdue: 0



## บทที่ 4

### อธิบายการทำงานของ Code

#### 4.1 ฟังก์ชันไบนารีพื้นฐานในระบบยืม-คืนหนังสือห้องสมุด

4.1.1 Module Struct เป็นโมดูลใน Python ที่ใช้สำหรับการจัดการข้อมูลแบบไบนารี เช่นการแปลงข้อมูลจากรูปแบบ Python (เช่นinteger, float) ไปเป็นไบนารีหรือการแปลงข้อมูลจากไบนารี กลับมาเป็นรูปแบบPython อีกครั้งโมดูลนี้สำคัญเมื่อเราต้องการทำงานกับไฟล์หรือข้อมูลที่อยู่ใน รูปแบบไบนารีเช่นไฟล์

```
import struct
```

ภาพที่ 4.1 Code module struct

4.1.2 import time คือคำสั่งในภาษา Python ที่ใช้สำหรับนำเข้า(import) โมดูลtime มาใช้งานโมดูลtime จะช่วยให้เราสามารถทำงานที่เกี่ยวข้องกับเวลา เช่นหน่วงเวลา (delay) จับเวลา การ ทำงาน แปลงเวลาให้อ่านง่ายimport time จะเรียกใช้โมดูลที่เกี่ยวข้องกับเวลา เพื่อหยุดเวลา,จับเวลา, หรือจัดการเวลา ได้สะดวกขึ้น

```
import time
```

ภาพที่ 4.2 Code module time

4.1.3 import shutil คือคำสั่งในภาษา Python ที่นำเข้า (import) โมดูล shutil ซึ่งเป็นโมดูลมาตรฐานที่ใช้สำหรับการจัดการไฟล์และโฟลเดอร์ เช่น การคัดลอกไฟล์ ลบไฟล์ ย้ายไฟล์ บีบอัดไฟล์ หรือดูขนาดของ terminal window โดยไม่ต้องใช้คำสั่งระบบภายนอก

```
import shutil
```

ภาพที่ 4.3 Code module shutil

4.1.4 import unicodedata คือคำสั่งในภาษา Python ที่นำเข้า (import) โมดูล unicodedata ซึ่งเป็นโมดูลมาตรฐานมีไว้สำหรับ จัดการและตรวจสอบตัวอักษรในระบบ Unicode

```
import shutil
```

ภาพที่ 4.4 Code module unicodedata

4.1.5 filesystem & time คือฟังก์ชันที่ช่วยให้โปรแกรมสามารถจัดการไฟล์และเวลาได้สะดวกขึ้น filesystem คือในส่วนที่ใช้สำหรับการตรวจสอบหรือสร้างไฟล์ เช่น ฟังก์ชัน `ensure_file(path: str)` ที่ใช้เช็คว่ามีไฟล์อยู่หรือไม่ ถ้าไม่มีจะสร้างไฟล์เปล่าขึ้นมา และ time คือในส่วนที่ใช้สำหรับการจัดการกับเวลา เช่น `now_epoch()` คือการคืนค่าเวลาปัจจุบันในรูปแบบ epoch (จำนวนวินาทีตั้งแต่ปี 1970) และ `fmt_date(ts: int)` สำหรับแปลง timestamp เป็นวันที่ในรูปแบบ YYYY-MM-DD

```
# ===== Helpers: filesystem & time =====
def ensure_file(path: str):
    if not os.path.exists(path):
        with open(path, "wb") as f:
            pass

def now_epoch() -> int:
    return int(time.time())

def fmt_date(ts: int) -> str:
    try:
        return datetime.fromtimestamp(int(ts)).strftime("%Y-%m-%d")
    except Exception:
        return "-"
```

ภาพที่ 4.5 Code function filesystem & time

4.1.6 fixed-size string คือเทคนิคการจัดเก็บข้อความให้มีขนาดเท่ากันทุก record เพื่อให้อ่าน/เขียนไฟล์ไบนารีได้ง่ายและตรงตำแหน่ง สตริงที่มีขนาด (จำนวนไบต์) คงที่เสมอ ไม่ว่าจะ มีข้อมูลจริงยาวเท่าไร ในโปรแกรมนี้นี้ใช้สำหรับจัดเก็บข้อมูลในไฟล์ไบนารีแบบ record ที่แต่ละฟิลด์ ต้องมีขนาดแน่นอน เช่น 12, 32, 64 ไบต์

```
# ===== Helpers: fixed-size string =====
def pack_str(s: str, size: int) -> bytes:
    b = (s or "").encode("utf-8")
    if len(b) > size:
        b = b[:size]
    return b + b"\x00" * (size - len(b))

def unpack_str(b: bytes) -> str:
    return b.split(b"\x00", 1)[0].decode("utf-8", "ignore")
```

ภาพที่ 4.6 Code function fixed-size string

4.1.7 table rendering คือการแปลงข้อมูลให้อยู่ในรูปแบบตารางข้อความที่สวยงามและอ่านง่าย โดยใช้ฟังก์ชันเหล่านี้ในการคำนวณความกว้าง ตัดข้อความ และจัดตำแหน่งแต่ละคอลัมน์ เช่น `_disp_width` ใช้สำหรับ คำนวณความกว้างที่แสดงจริงของข้อความ `_truncate_to_width` ตัดข้อความให้พอดีกับความกว้างที่กำหนด `_pad` เพิ่มช่องว่างให้ข้อความมีความกว้างตามที่ต้องการ และจัดตำแหน่ง (ซ้าย/ขวา/กลาง) และ `render_table` ฟังก์ชันหลักสำหรับสร้างตารางข้อความ

```
# ===== Helpers: table rendering =====
def _disp_width(text: str) -> int:
    w = 0
    for ch in str(text):
        if unicodedata.combining(ch):
            continue
        ea = unicodedata.east_asian_width(ch)
        w += 2 if ea in ("W", "F") else 1
    return w

def _truncate_to_width(text: str, width: int) -> str:
    # ตัดข้อความให้พอดีความกว้าง width พร้อมเติม '...' ถ้าจำเป็น
    if _disp_width(text) <= width:
        return text
    if width <= 1:
        return "...[:width]"
    out = []
    used = 0
    for ch in str(text):
        cw = 0 if unicodedata.combining(ch) else (2 if unicodedata.east_asian_width(ch) in ("W", "F") else 1)
        if used + cw >= width:
            break
        out.append(ch)
        used += cw
        if used >= width - 1:
            break
    return "".join(out) + "..."
```

ภาพที่ 4.7 Code function rendering

```

def _pad(text: str, width: int, align: str = "left") -> str:
    # เติมช่องว่างให้ได้ความกว้างที่ "แสดง" เท่ากับ width
    raw = str(text)
    raw = _truncate_to_width(raw, width)
    gap = width - _disp_width(raw)
    if align == "right":
        return " " * gap + raw
    elif align == "center":
        left = gap // 2
        right = gap - left
        return " " * left + raw + " " * right
    else:
        return raw + " " * gap

def render_table(headers, rows, aligns=None, max_total=None, sep=" | ", hline_char="-"):
    """
    headers: [str, ...]
    rows:      [[cell,...], ...]
    aligns:    ["left"/"right"/"center", ...] (ถ้าไม่ระบุ default = left)
    max_total: จำกัดความกว้างตารางรวม (ไม่รวม margin ซ้ายขวา) ถ้า None จะใช้ความกว้างจอ
    """
    if aligns is None:
        aligns = ["left"] * len(headers)

    # ความกว้างขั้นต่ำของคอลัมน์ = max(ความกว้าง header และค่ามากสุดในแถว)
    cols = len(headers)
    widths = [0] * cols
    for i, h in enumerate(headers):
        widths[i] = max(widths[i], _disp_width(h))
    for r in rows:
        for i in range(cols):
            if i < len(r):
                widths[i] = max(widths[i], _disp_width("" if r[i] is None else r[i]))

```

ภาพที่ 4.7 Code function rendering (ต่อ)



```

# เพื่อช่องคั่น
sep_w = _disp_width(sep)
table_content_width = sum(widths) + sep_w * (cols - 1)

# จำกัดให้พอดีกับจอ
term_w = shutil.get_terminal_size(fallback=(120, 30)).columns
limit = max_total or max(60, term_w - 2) # กันเผื่อเล็กน้อย
if table_content_width > limit:
    # ตัดคอลัมน์แบบกระจาย
    min_col = [max(6, min(30, w)) for w in widths]
    need = table_content_width - limit
    pairs = sorted(list(enumerate(widths)), key=lambda x: x[1], reverse=True)
    for idx, _w in pairs:
        can_reduce = widths[idx] - min_col[idx]
        if can_reduce <= 0:
            continue
        step = min(can_reduce, need)
        widths[idx] -= step
        need -= step
        if need <= 0:
            break

```

ภาพที่ 4.7 Code function rendering (ต่อ)

```

def hline(ch=hline_char):
    return ch * (sum(widths) + sep_w * (cols - 1))

header_line = sep.join(_pad(headers[i], widths[i], "center") for i in range(cols))

body_lines = []
for r in rows:
    cells = []
    for i in range(cols):
        val = "" if i >= len(r) or r[i] is None else str(r[i])
        cells.append(_pad(val, widths[i], aligns[i] if i < len(aligns) else "left"))
    body_lines.append(sep.join(cells))

top = hline()
bottom = hline()
return "\n".join([top, header_line, top] + body_lines + [bottom])

```

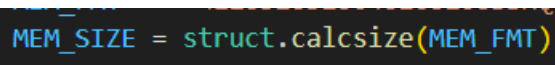
ภาพที่ 4.7 Code function rendering (ต่อ)

4.1.8 MEM\_FMT คือการกำหนดโครงสร้างข้อมูลสมาชิกแต่ละ record ในไฟล์ members.dat ให้มีขนาดและลำดับฟิลด์ที่แน่นอน เพื่อให้สามารถอ่าน/เขียนข้อมูลได้อย่างถูกต้องและรวดเร็ว

```
MEM_FMT = "<12s32s32s64s16s16sBBHQ"
```

ภาพที่ 4.8 Code MEM\_FMT

4.1.9 MEM\_SIZE คือการคำนวณขนาด (จำนวนไบต์) ของข้อมูลสมาชิก (Member) 1 record ตามรูปแบบที่กำหนดใน MEM\_FMT ด้วยฟังก์ชัน struct.calcsize() ผลลัพธ์ที่ได้คือ 184 ไบต์



```
MEM_SIZE = struct.calcsize(MEM_FMT)
```

ภาพที่ 4.9 Code MEM\_SIZE

4.1.10 MemberStore คือคลาสสำหรับการจัดการข้อมูลสมาชิกในระบบห้องสมุด โดยทำงานกับไฟล์ไบนารีที่เก็บข้อมูลสมาชิกแต่ละคนแบบ fixed-size record (ขนาด 184 ไบต์/record) ฟังก์ชันหลักแต่ละตัวในคลาสนี้มีหน้าที่ดังนี้:

1. ฟังก์ชัน init คือการกำหนด path ของไฟล์ข้อมูลสมาชิก และสร้างไฟล์เปล่าหากยังไม่มี
2. ฟังก์ชัน pack คือการรับ dict ข้อมูลสมาชิก แปลงเป็น bytes ตามรูปแบบ MEM\_FMT เพื่อบันทึกลงไฟล์
3. ฟังก์ชัน unpack คือการรับ bytes จากไฟล์ แปลงกลับเป็น dict ข้อมูลสมาชิก
4. ฟังก์ชัน append คือการเพิ่มสมาชิกใหม่ (เช็คว่า member\_id ไม่ซ้ำก่อน)
5. ฟังก์ชัน iter\_all คือการอ่านสมาชิกทุกคนในไฟล์ทีละ record (ใช้ yield ส่งออกทีละ dict)
6. ฟังก์ชัน get\_by\_id คือการค้นหาสมาชิกจาก member\_id ถ้าพบคืนค่า (dict, index) ถ้าไม่พบคืน None
7. ฟังก์ชัน update\_at คือการอัปเดตข้อมูลสมาชิกที่ตำแหน่ง index ที่ระบุ
8. ฟังก์ชัน soft\_delete คือการปิดการใช้งานสมาชิก (active=False) โดยค้นหา member\_id แล้วอัปเดตฟิลด์ active

```

class MemberStore:
    def __init__(self, path=MEMBERS_DAT):
        self.path = path
        ensure_file(self.path)

    def pack(self, d: dict) -> bytes:
        return struct.pack(
            MEM_FMT,
            pack_str(d["member_id"], 12),
            pack_str(d["first_name"], 32),
            pack_str(d["last_name"], 32),
            pack_str(d["email"], 64),
            pack_str(d["phone"], 16),
            pack_str(d.get("major", ""), 16),
            int(d.get("year", 0)) & 0xFF,
            1 if d.get("active", True) else 0,
            0, # reserved
            int(d.get("created_at", now_epoch())))
        )

    def unpack(self, b: bytes) -> dict:
        (mid, first, last, email, phone, major, year, active, _res, created) = struct.unpack(MEM_FMT, b)
        return {
            "member_id": unpack_str(mid),
            "first_name": unpack_str(first),
            "last_name": unpack_str(last),
            "email":      unpack_str(email),
            "phone":      unpack_str(phone),
            "major":      unpack_str(major),
            "year":        int(year),
            "active":      bool(active),
            "created_at": int(created),
        }

```

ภาพที่ 4.10 Code class MemberStore

```

def append(self, d: dict):
    # unique member_id check
    if self.get_by_id(d["member_id"]) is not None:
        raise ValueError(f"Member ID '{d['member_id']}' already exists")
    with open(self.path, "ab") as f:
        f.write(self.pack(d))

def iter_all(self):
    with open(self.path, "rb") as f:
        while True:
            chunk = f.read(MEM_SIZE)
            if not chunk or len(chunk) < MEM_SIZE:
                break
            yield self.unpack(chunk)

def get_by_id(self, member_id: str):
    with open(self.path, "rb") as f:
        idx = 0
        while True:
            chunk = f.read(MEM_SIZE)
            if not chunk or len(chunk) < MEM_SIZE: break
            rec = self.unpack(chunk)
            if rec["member_id"] == member_id:
                return rec, idx
            idx += 1
    return None

def update_at(self, index: int, d: dict):
    with open(self.path, "r+b") as f:
        f.seek(index * MEM_SIZE)
        f.write(self.pack(d))

```

ภาพที่ 4.10 Code class MemberStore (ต่อ)

```

def soft_delete(self, member_id: str):
    res = self.get_by_id(member_id)
    if not res: raise ValueError("Member not found")
    rec, idx = res
    rec["active"] = False
    self.update_at(idx, rec)

```

ภาพที่ 4.10 Code class MemberStore (ต่อ)

4.1.11 BookStore คือคลาสสำหรับการจัดการข้อมูลหนังสือในระบบห้องสมุด โดยเก็บข้อมูลแต่ละเล่มในไฟล์ไบนารี (books.dat) แบบ fixed-size record (ขนาด 158 ไบต์/record) ฟังก์ชันหลักแต่ละตัวมีหน้าที่ดังนี้

1. ฟังก์ชัน init คือการกำหนด path ของไฟล์ข้อมูลหนังสือ และสร้างไฟล์เปล่าหากยังไม่มี

2. ฟังก์ชัน pack คือการรับ dict ข้อมูลหนังสือ แปลงเป็น bytes ตามรูปแบบ BOOK\_FMT เพื่อบันทึกลงไฟล์
3. ฟังก์ชัน unpack คือการรับ bytes จากไฟล์ แปลงกลับเป็น dict ข้อมูลหนังสือ
4. ฟังก์ชัน append คือการเพิ่มหนังสือใหม่ (เช็คว่า book\_id ไม่ซ้ำก่อน)
5. ฟังก์ชัน iter\_all คือการอ่านหนังสือทุกเล่มในไฟล์ทีละ record (ใช้ yield ส่งออกทีละ dict)
6. ฟังก์ชัน get\_by\_id คือการค้นหาหนังสือตาม book\_id ถ้าพบคืนค่า (dict, index) ถ้าไม่พบคืน None
7. ฟังก์ชัน update\_at คือการอัปเดตข้อมูลหนังสือที่ตำแหน่ง index ที่ระบุ
8. ฟังก์ชัน delete\_hard คือการลบหนังสือแบบถาวร (hard delete) โดยเขียนไฟล์ใหม่โดยไม่รวม record ที่ต้องการลบ

```
class BookStore:
    def __init__(self, path=BOOKS_DAT):
        self.path = path
        ensure_file(self.path)

    def pack(self, d: dict) -> bytes:
        return struct.pack(
            BOOK_FMT,
            pack_str(d["book_id"], 12),
            pack_str(d["title"], 64),
            pack_str(d["author"], 32),
            pack_str(d.get("category", ""), 16),
            int(d.get("year", 0)) & 0xFFFF,
            pack_str(d.get("isbn", ""), 20),
            int(d.get("total_copies", 1)) & 0xFFFF,
            int(d.get("available_copies", d.get("total_copies", 1))) & 0xFFFF,
            int(d.get("created_at", now_epoch()))
        )

    def unpack(self, b: bytes) -> dict:
        (bid, title, author, category, year, isbn, total, avail, created) = struct.unpack(BOOK_FMT, b)
        return {
            "book_id": unpack_str(bid),
            "title": unpack_str(title),
            "author": unpack_str(author),
            "category": unpack_str(category),
            "year": int(year) if year else None,
            "isbn": unpack_str(isbn),
            "total_copies": int(total),
            "available_copies": int(avail),
            "created_at": int(created),
        }
```

ภาพที่ 4.11 Code class BookStore

```

def append(self, d: dict):
    if self.get_by_id(d["book_id"]) is not None:
        raise ValueError(f"Book ID '{d['book_id']}' already exists")
    with open(self.path, "ab") as f:
        f.write(self.pack(d))

def iter_all(self):
    with open(self.path, "rb") as f:
        while True:
            chunk = f.read(BOOK_SIZE)
            if not chunk or len(chunk) < BOOK_SIZE: break
            yield self.unpack(chunk)

def get_by_id(self, book_id: str):
    with open(self.path, "rb") as f:
        idx = 0
        while True:
            chunk = f.read(BOOK_SIZE)
            if not chunk or len(chunk) < BOOK_SIZE: break
            rec = self.unpack(chunk)
            if rec["book_id"] == book_id:
                return rec, idx
            idx += 1
    return None

def update_at(self, index: int, d: dict):
    with open(self.path, "r+b") as f:
        f.seek(index * BOOK_SIZE)
        f.write(self.pack(d))

```

ภาพที่ 4.11 Code class BookStore (ต่อ)

```

def delete_hard(self, book_id: str):
    # rewrite without the target (hard delete)
    records = [r for r in self.iter_all() if r["book_id"] != book_id]
    with open(self.path, "wb") as f:
        for r in records:
            f.write(self.pack(r))

```

ภาพที่ 4.11 Code class BookStore (ต่อ)

LOAN\_FMT กำหนดรูปแบบข้อมูล 1 รายการยืม-คืนในไฟล์ไบนารี  
 LOAN\_SIZE คือขนาดของข้อมูล 1 รายการ (64 ไบต์) ตามรูปดังนี้

```

LOAN_FMT = "<12s12s12sQQQB3x"
LOAN_SIZE = struct.calcsize(LOAN_FMT)

```

```

def get_by_id(self, loan_id: str):
    with open(self.path, "rb") as f:
        idx = 0
        while True:
            chunk = f.read(LOAN_SIZE)
            if not chunk or len(chunk) < LOAN_SIZE: break
            rec = self.unpack(chunk)
            if rec["loan_id"] == loan_id:
                return rec, idx
            idx += 1
    return None

def update_at(self, index: int, d: dict):
    with open(self.path, "r+b") as f:
        f.seek(index * LOAN_SIZE)
        f.write(self.pack(d))

```

4.1.1.2 ฟังก์ชัน `get_by_id(self, loan_id: str)` หน้าที่ค้นหา “เรคคอร์ด (record)” ในไฟล์ ที่มี `loan_id` ตรงกับค่าที่กำหนดการทำงาน: เปิดไฟล์แบบอ่านไบนารี (rb) อ่านข้อมูลทีละก้อน (ขนาด `LOAN_SIZE` ไบต์) แปลงข้อมูลแต่ละก้อนเป็น dict (`self.unpack`) ถ้าเจอ `loan_id` ที่ตรงกัน → คืนข้อมูลเรคคอร์ดและตำแหน่ง (`rec, idx`) ถ้าอ่านจนจบแล้วยังไม่เจอ → คืน `None`

ฟังก์ชัน `update_at(self, index: int, d: dict)` อัปเดตข้อมูลของเรคคอร์ดที่ตำแหน่ง `index` ในไฟล์ ทำงาน: เปิดไฟล์แบบอ่าน/เขียนไบนารี (r+b) คำนวณตำแหน่งไบต์ที่จะเขียน = `index * LOAN_SIZE` เลื่อนไปตำแหน่งนั้น (`seek`) เขียนข้อมูลใหม่ (หลังแปลงเป็นไบต์ด้วย `self.pack`) ทับข้อมูลเดิม

```

class LoanStore:
    def __init__(self, path=LOANS_DAT):
        self.path = path
        ensure_file(self.path)

    def pack(self, d: dict) -> bytes:
        return struct.pack(
            LOAN_FMT,
            pack_str(d["loan_id"], 12),
            pack_str(d["member_id"], 12),
            pack_str(d["book_id"], 12),
            int(d.get("loan_date", now_epoch())),
            int(d.get("due_date", now_epoch()+14*86400)),
            int(d.get("return_date", 0)),
            int(d.get("status", 0)) & 0xFF,
        )

    def unpack(self, b: bytes) -> dict:
        (lid, mid, bid, loan_dt, due_dt, ret_dt, status) = struct.unpack(LOAN_FMT, b)
        return {
            "loan_id": unpack_str(lid),
            "member_id": unpack_str(mid),
            "book_id": unpack_str(bid),
            "loan_date": int(loan_dt),
            "due_date": int(due_dt),
            "return_date": int(ret_dt),
            "status": int(status),
        }

    def append(self, d: dict):
        with open(self.path, "ab") as f:
            f.write(self.pack(d))

    def iter_all(self):
        with open(self.path, "rb") as f:
            while True:
                chunk = f.read(LOAN_SIZE)
                if not chunk or len(chunk) < LOAN_SIZE: break
                yield self.unpack(chunk)

```

รูปภาพที่ 4- 11

#### 4.1.1.3 LoanStore คือคลาสสำหรับจัดการข้อมูลการยืม-คืนหนังสือ (loan) ในระบบ

ห้องสมุด โดยเก็บข้อมูลแต่ละรายการในไฟล์ไบนารี (loans.dat) แบบ fixed- size record (ขนาด 64 ไบต์/record) ฟังก์ชันหลักแต่ละตัวมีหน้าที่ดังนี้: `__init__`: กำหนด path ของไฟล์ข้อมูลยืม-คืน และสร้างไฟล์เปล่าหากยังไม่มี `pack`: รับ dict ข้อมูลการยืม-คืน แปลงเป็น bytes ตามรูปแบบ LOAN\_FMT เพื่อบันทึกลงไฟล์ `unpack`: รับ bytes จากไฟล์ แปลงกลับเป็น dict ข้อมูลการยืม-คืน `append`: เพิ่มข้อมูลการยืม-คืนใหม่ (เขียนต่อท้ายไฟล์) `iter_all`: อ่านข้อมูลการยืม-คืนทุก record ในไฟล์ทีละรายการ (ใช้ yield ส่งออกทีละ dict) `get_by_id`: ค้นหาข้อมูลการยืม-คืนตาม loan\_id ถ้าพบคืนค่า (dict, index) ถ้าไม่พบคืน None `update_at`: อัปเดตข้อมูลการยืม-คืนที่ตำแหน่ง index ที่ระบุ



## บทที่ 5

### สรุปผลการดำเนินงาน

ในบทนี้กล่าวถึงสรุปผลการดำเนินงานของโครงการ สามารถแบ่งออกเป็น 4 ส่วน ดังนี้

#### 5.1 สรุปผลการดำเนินงานโครงการ

ระบบยืม – คืนหนังสือห้องสมุด ที่พัฒนาขึ้นสามารถช่วยจัดการข้อมูลหนังสือ ข้อมูลสมาชิก และข้อมูลการยืม – คืนได้อย่างมีประสิทธิภาพ โดยใช้การจัดเก็บข้อมูลแบบไฟล์ไบนารี พร้อมเมนูสำหรับเพิ่ม แก้ไข ลบ และแสดงผลข้อมูล ระบบยังรองรับการตรวจสอบสถานะหนังสือที่ถูกยืมหรือยังว่างอยู่ การควบคุมจำนวนเล่มที่ถูกยืม ตลอดจนการสร้างรายงานสรุปผลการดำเนินงาน เช่น จำนวนหนังสือที่ถูกยืมมากที่สุด รายชื่อผู้ยืมปัจจุบัน และสถิติการใช้งานโดยรวม ซึ่งช่วยให้การบริหารจัดการห้องสมุดสะดวก รวดเร็ว และลดความผิดพลาดจากการบันทึกแบบเดิมที่ใช้เอกสารกระดาษ

#### 5.2 ปัญหาและอุปสรรคในการดำเนินงาน

ในการพัฒนาระบบยืม – คืนหนังสือห้องสมุด พบปัญหาหลักคือ ความซับซ้อนของการจัดการไฟล์ไบนารีที่ต้องใช้โครงสร้างข้อมูลคงที่ (struct) ซึ่งอาจเกิดข้อผิดพลาดหากการเข้ารหัสหรือถอดรหัสไม่ถูกต้อง นอกจากนี้ยังพบข้อจำกัดด้านการแสดงผลข้อมูลความยาวของชื่อหนังสือหรือชื่อผู้ใช้ที่ต้องถูกจำกัดตามขนาดที่กำหนดไว้ อีกทั้งระบบยังไม่มีการเชื่อมต่อฐานข้อมูลจริง ทำให้การจัดการข้อมูลจำนวนมากหรือการเข้าถึงพร้อมกันจากหลายผู้ใช้งานยังไม่สามารถทำได้เต็มที่

#### 5.3 ข้อเสนอแนะ

เพื่อให้ระบบสมบูรณ์และพร้อมใช้งานจริงในอนาคต ควรปรับปรุงดังนี้

5.3.1 พัฒนาให้รองรับฐานข้อมูลเชิงสัมพันธ์ (Relational Database) เช่น MySQL หรือ SQLite เพื่อรองรับข้อมูลจำนวนมากและการเข้าถึงหลายผู้ใช้งาน

5.3.2 เพิ่มฟังก์ชันค้นหาและกรองข้อมูล เช่น ค้นหาหนังสือตามชื่อ ผู้แต่ง หรือปีที่พิมพ์

5.3.3 ปรับปรุงระบบยืนยันตัวตนสมาชิก และจำกัดสิทธิ์การเข้าถึงของผู้ใช้แต่ละกลุ่ม

5.3.4 พัฒนาเป็นโปรแกรมที่มีส่วนติดต่อผู้ใช้แบบกราฟิก (GUI) หรือเว็บแอปพลิเคชัน เพื่อความสะดวกในการใช้งานจริง

#### 5.4 สิ่งที่ได้จัดทำได้รับในการพัฒนาโครงการ

จากการพัฒนาโครงการครั้งนี้ ผู้จัดทำได้รับความรู้และประสบการณ์ด้านการออกแบบระบบ การเขียนโปรแกรมด้วยภาษา Python การใช้โครงสร้างข้อมูลแบบไบนารี รวมถึงการคิดวิเคราะห์และแก้ไขปัญหาเชิงตรรกะ นอกจากนี้ยังได้ฝึกทักษะการทำงานเป็นทีม การแบ่งหน้าที่รับผิดชอบ และการจัดการเวลาให้สอดคล้องกับแผนงานทำให้ผู้จัดทำมีความเข้าใจในกระบวนการพัฒนาระบบซอฟต์แวร์มากยิ่งขึ้น และสามารถนำไปประยุกต์ใช้ในโครงการหรืองานจริงในอนาคตได้