

프로그램 개요:

이 문서는 다섯 개의 단계별 과제를 통해 공공 자전거 대여·반납·재고 관리 시스템을 점진적으로 구현하는 과정을 설명한다.

자전거 대여 비용은 이용 시간과 상관없이 **대여 1회당 고정 비용**이 적용되는 시스템이다.

이 시스템은 아래와 같이 두 가지 모드로 동작한다.

- 관리자 모드(admin mode)
- 사용자 모드(user mode)

역할:

- 관리자: 전체 자전거 재고 확인, 사용자 대여 기록 조회, 미반납 사용자 정보 조회
- 사용자: 대여 가능한 자전거 목록 조회, 자전거 선택 및 대여, 반납, 대여 확인서 발급

구현 지침:

- 문제 1~5는 이전 단계의 코드를 확장하는 방식으로 구성된다. 단, 각 단계별 결과물은 별도로 제출되어야 한다.
- 문제에서 요구하는 함수 이름은 반드시 그대로 사용해야 한다. 단, 매개변수와 반환값의 타입/구성은 명시되지 않은 경우 본인이 직접 설계한다.

채점 관련 안내사항:

- 각 단계마다 부분 점수가 존재한다.
- 출력의 세부 형식(띄어쓰기·줄바꿈·문장 부호 등)은 감점하지 않는다. 단, 요구된 정보는 모두 포함되어야 한다.
- 모든 아이디/비밀번호 등 입력받는 문자들의 경우, 대소문자를 구분한다.
- 반복이 명시되지 않은 경우, 본 코드는 단일 플로우만 수행하며, 추가적인 반복 실행이나 지속 실행은 필요하지 않다.
- 함수를 추가로 구현하여 사용하는 것 또한 가능하다.
- 할당한 동적 메모리를 해제하지 않으면 메모리 누수로 간주하며 감점된다.

<참고>

```
#define _CRT_SECURE_NO_WARNINGS
```

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```
#include <string.h>
```

1번 문제: 로그인 및 모드 구분

해당 시스템은 관리자/사용자 두가지 모드를 제공한다. 사용자가 입력한 로그인 아이디를 통해 관리자 모드와 서비스 모드를 구분한다.

아래와 같은 단계 및 조건으로 수행된다.

1. 사용자가 로그인 아이디를 입력한다.

로그인 아이디에 따른 각 모드는 다음과 같다. (비밀번호는 없다고 가정)

- 관리자 모드: admin
 - 사용자 모드: user.txt 파일에 등록된 아이디
 - 이때, user.txt는 한 줄에 사용자 아이디 정보만 존재한다.
 - 그 외의 경우, 잘못된 아이디임을 알리고 재입력받는다.
2. 현재 프로그램이 어떤 모드로 동작하는지 사용자에게 알려준다.
 3. 관리자/사용자 여부는 enum으로 관리한다.

구현 및 수정해야 할 함수:

check_access():

- 사용자로부터 로그인 아이디를 입력받는다.
- admin이면 관리자에 해당하는 enum의 멤버를 반환한다.
- user.txt안에 존재하는 아이디이면 사용자에게 해당하는 enum의 멤버를 반환한다.
- 그 외의 경우, 적절한 안내 메시지 출력 후 재입력받는다.

main():

- check_access()를 호출하여 enum 멤버(모드)를 받아온다.
- 받은 모드에 따라 아래와 같이 출력한다.
 - ◆ 관리자 모드: "관리자 모드입니다"
 - ◆ 사용자 모드: "사용자 모드입니다"

결과 예시:

관리자 모드 로그인		로그인 ID를 입력하세요 : admin 관리자 모드입니다 .	
사용자 모드 로그인		로그인 ID를 입력하세요 : wrong 잘못된 ID입니다 . 다시 시도하세요 . 로그인 ID를 입력하세요 : jully 사용자 모드입니다 .	

2번 문제: 자전거 목록 불러오기 대여 / 대여 기록 출력

해당 시스템은 사용자 모드일 때 대여 가능한 자전거 목록을 출력하고, 자전거 대여 서비스를 수행한다.

이를 위해 자전거 목록을 파일에서 읽어 메모리에 저장한 뒤 화면에 출력한다. 이후 사용자의 대여 요청을 받아 처리하며, 대여가 확정되면 대여 확인서를 출력한다.

단, 1번 문제의 기능(로그인 및 모드 구분)이 포함된 상태에서, 사용자 모드로 로그인했을 때만 해당 기능이 동작해야 한다.

다음과 같은 단계 및 조건으로 수행된다.

1. 사용자 모드일 때, bikes.txt 파일에서 자전거 목록을 읽어 메모리에 저장한다.
 - bikes.txt 파일에는 한 행에 하나의 자전거 이름만 존재한다.
2. 사용자에게 자전거 목록을 출력한다.
 - 자전거 목록을 출력할 때, 각 자전거에 자전거 번호를 붙여서 출력한다.
 - 자전거의 자전거 번호는 자전거 리스트를 저장한 배열의 인덱스와 동일하게 지정한다.
3. 사용자로부터 복수의 자전거 대여 요청을 받기 위해, 사용자가 대여 확정을 하기 전까지 반복하여 요청한다.
 - 이때, 사용자는 자전거 번호로 대여 요청을 진행한다.
 - 존재하지 않는 자전거 번호를 입력할 시, 예외처리(오류 메시지 출력) 후 재입력 받는다.
 - 사용자가 자전거 번호로 -1을 입력하는 경우, 대여 확정으로 처리한다.
4. 사용자가 대여를 완료하면, 사용자의 대여 확인서를 출력한다.

이때, 자전거 리스트와 사용자 대여 리스트는 **문자열 배열**로 관리하며, **반드시 동적 메모리 할당**을 이용해 관리해야 한다. (단순 고정 배열 시 감점)

구현 및 수정해야 할 함수:

read_bikes_list():

- bikes.txt에서 자전거 이름들을 읽어서 배열에 저장한다.

user_mode():

- 자전거 정보를 읽기 위한 read_bikes_list()를 호출한다.
- 자전거 목록을 화면에 출력한다.
- 사용자로부터 대여할 자전거 번호를 반복적으로 입력받는다.
- 사용자가 대여 확정을 선택하면, 이번 실행 동안 사용자가 대여한 자전거 목록(대여

확인서)를 화면에 출력한다.

main():

- 1단계 기능을 포함한다.
- 유저 모드로 로그인했을 경우, user_mode()를 호출한다.

결과 예시:

자전거 목록 출력	<pre>=== 자전거 목록 === [0] Mountain Bike A-Type [1] Folding Bike B-Type [2] Electric Bike Ultra-EV [3] Road Bike Speed-10 [4] Electric Bike C-Type [5] Fixie Bike Fix-01 [6] Urban Bike Urban-Alpha</pre>
대여 진행	<pre>대여할 자전거 번호 (-1 입력 시 대여 확정): 3 "Road Bike Speed-10" 자전거가 대여 목록에 추가되었습니다. 대여할 자전거 번호 (-1 입력 시 대여 확정): 3 "Road Bike Speed-10" 자전거가 대여 목록에 추가되었습니다. 대여할 자전거 번호 (-1 입력 시 대여 확정): 2 "Electric Bike Ultra-EV" 자전거가 대여 목록에 추가되었습니다. 대여할 자전거 번호 (-1 입력 시 대여 확정): 9 존재하지 않는 자전거 번호입니다. 다시 입력하세요. 대여할 자전거 번호 (-1 입력 시 대여 확정): -1 === 대여 확인서 === 1. Road Bike Speed-10 2. Road Bike Speed-10 3. Electric Bike Ultra-EV</pre>

3번 문제: 재고 및 결제 시스템 업그레이드

2단계까지 구현한 사용자 모드 기능을 확장하여 3단계에서는 자전거 브랜드별 대여 비용과 대여 가능 수량(재고량)를 고려하여 대여를 받도록 한다.

또한, 각 사용자의 대여 현황을 파일로 저장한다.

아래와 같은 단계 및 조건으로 수행된다.

1. 사용자 모드일 때, bikes_v2.txt파일에서 자전거 정보 목록을 읽어 메모리에 저장한다
 - bikes_v2.txt에는 자전거 이름, 대여 비용, 현재 대여 가능 수량(재고량) 정보가 존재한다.
 - 각 정보들은 'wt'(tab)으로 구분되어 있다.
 - 자전거 번호, 자전거 이름, 대여 비용, 재고량 정보를 저장하기 위해 구조체를 사용한다.
 - 자전거 리스트는 여러 종류의 자전거가 존재하기 때문에 구조체 배열을 사용한다.
2. 사용자에게 대여 가능 자전거 목록을 출력한다.
 - 자전거의 번호, 이름, 대여 비용 및 대여 가능 수량(재고량)을 모두 출력한다.
 - 이때, 대여 가능 수량(재고량)이 0인 자전거는 목록 출력에 포함하지 않는다.
3. 사용자로부터 복수의 자전거 대여 요청을 받기 위해, 사용자가 대여 확정을 하기 전까지 반복하여 요청을 받는다.
 - 사용자는 자전거 번호로 대여를 진행한다.
 - 사용자가 자전거 번호로 -1을 입력하는 경우, 대여 확정으로 처리한다.
 - 이 외의 존재하지 않는 자전거 번호를 입력할 시, 적절한 메시지를 출력하여 예외처리를 진행한다.
 - 사용자가 하나의 자전거를 선택하면, 대여 내역에 추가한다.
 - ◆ 대여 내역 또한 자전거 번호, 자전거 이름, 대여 비용, 대여 중인 자전거 수량 정보를 저장하는 구조체로 구성되어야 한다.
 - ◆ 이미 해당 사용자가 대여 중인 자전거 종류인 경우, 수량만을 증가시킨다.
 - ◆ 재고가 없는 자전거 종류일 경우, 자전거의 재고가 부족하다는 메시지를 출력하고, 대여 내역에 추가하지 않는다.
4. 사용자가 대여를 확정하면, 현재 실행(세션)의 대여 내역 정보(대여 확인서)를 출력하고, 이 정보를 이용해 사용자의 대여 현황을 파일에 저장한다.
 - 대여 기록은 rental_history.txt 파일에 저장한다.
 - ◆ 파일이 존재하지 않으면 생성한다.
 - 모든 사용자의 대여 현황을 rental_history.txt 하나의 파일에서 누적 관리한다.
 - 이미 같은 사용자아이디로 같은 자전거 종류의 대여 기록이 있으면 수량만을 누적 수정하고, 없으면 새 레코드를 추가한다.

- 파일 내부 형식은 다음과 같다.
 - ◆ 사용자아이디\자전거이름\사용자가 대여중인 수량
- 5. 사용자가 자전거를 대여한 후, 변화한 대여가능 자전거 수량(재고)를 bikes_v2.txt 파일에 반영한다.
 - 대여된 수량만큼 각 자전거의 재고를 감소시켜 파일에 다시 쓴다.

이때, 자전거 리스트와 사용자 대여 내역은 구조체 배열로 관리하며, 반드시 동적 메모리 할당을 이용해 관리해야 한다. 또한, 자전거 구조체 내부의 자전거 이름 또한 동적 메모리 할당을 사용한다. (단순 고정 배열 시 감점)

구현 및 수정해야 할 함수:

read_bikes_list_v2():

- bikes_v2.txt에서 자전거 정보들(이름, 대여 비용, 대여 가능 수량(재고))을 읽어서 구조체 배열에 저장한다.
 - ◆ 이때, 각 요소에 자전거 번호를 부여한다.

save_rental_history():

- 사용자가 현재 세션에서 대여한 자전거 내역을 이용하여 rental_history.txt에 기록되어 있는 사용자의 대여 현황을 수정한다.
 - ◆ 만약, 해당 파일이 존재하지 않는다면 생성하여 기록한다.
- 이미 해당 사용자 아이디에 대하여, 같은 자전거 종류에 대한 대여 기록이 존재한다면,
 - ◆ 수량만을 누적 수정하고, 없으면 새 레코드를 추가한다.
- 사용자아이디, 자전거이름, 대여중인 수량을 기록한다.
 - ◆ 기록 형식은 'Вт'(tab)으로 구분된 텍스트 형태로 저장한다.

write_updated_bikes():

- 자전거 대여 내역을 기준으로, 변경된 대여 가능 자전거 수량(재고)을 bikes_v2.txt에 반영한다.

user_mode():

2단계의 user_mode()를 확장한다.

- 자전거 정보를 읽기 위한 read_bikes_list_v2()를 호출한다.
- 자전거 리스트를 출력할 때, 자전거 번호, 자전거 이름, 대여 비용과 대여 가능 자전거 수량(재고량)을 함께 출력한다.
 - ◆ 단, 대여 가능 자전거 수량(재고)이 0인 자전거는 출력하지 않는다.

- 사용자로부터 대여할 자전거 번호를 입력받는다.
 - ◆ -1(대여 확정)을 선택할 때까지 반복적으로 입력받는다.
 - ◆ 잘못된 번호 입력에 대한 예외처리를 수행한다.
 - ◆ 대여 내역에 정보를 추가한다.
 - ◆ 선택한 자전거의 재고가 부족하면, 관련 메시지를 출력하며 대여 내역에 추가하지 않는다.
- 사용자가 대여 확정을 선택하면, 현재 실행(세션)의 대여 내역(사용자가 대여한 자전거들)을 화면에 출력한다.
- save_rental_history()를 호출하여 해당 사용자의 대여 현황을 갱신한다.
- 자전거 재고 변화를 반영하기 위해 write_updated_bikes()를 호출한다.

결과 예시:

자전거 대여 및 그 결과(1)	로그인 ID를 입력하세요: jully 사용자 모드입니다.			
	=== 대여 가능 자전거 목록 ===			
	번호	자전거 이름	비용	재고
	0	Mountain Bike A-Type	20000	4
	1	Folding Bike B-Type	50000	2
	2	Electric Bike Ultra-EV	30000	8
	4	Electric Bike C-Type	80000	4
	6	Urban Bike Urban-Alpha	70000	6
	대여할 자전거 번호 (-1 입력 시 대여 확정): 1 "Folding Bike B-Type" 자전거가 대여 목록에 추가되었습니다. 대여할 자전거 번호 (-1 입력 시 대여 확정): 1 "Folding Bike B-Type" 자전거가 대여 목록에 추가되었습니다. 대여할 자전거 번호 (-1 입력 시 대여 확정): 9 존재하지 않는 자전거 번호입니다. 다시 입력하세요. 대여할 자전거 번호 (-1 입력 시 대여 확정): 1 재고가 부족한 자전거입니다. 대여할 자전거 번호 (-1 입력 시 대여 확정): 2 "Electric Bike Ultra-EV" 자전거가 대여 목록에 추가되었습니다. 대여할 자전거 번호 (-1 입력 시 대여 확정): -1			
	=== 이번 세션 대여 내역 ===			
	번호	자전거 이름	수량	비용
	0	Folding Bike B-Type	2	50000
	1	Electric Bike Ultra-EV	1	30000
	총 결제 금액: 130000			

	<div data-bbox="619 224 1182 649"> <div>bikes_v2.txt</div> <div>파일 편집 보기 H1 ≡ ...</div> <table> <tr><td>Mountain Bike A-Type</td><td>20000</td><td>4</td></tr> <tr><td>Folding Bike B-Type</td><td>50000</td><td>0</td></tr> <tr><td>Electric Bike Ultra-EV</td><td>30000</td><td>7</td></tr> <tr><td>Road Bike Speed-10</td><td>10000</td><td>0</td></tr> <tr><td>Electric Bike C-Type</td><td>80000</td><td>4</td></tr> <tr><td>Fixie Bike Fix-01</td><td>30000</td><td>0</td></tr> <tr><td>Urban Bike Urban-Alpha</td><td>70000</td><td>6</td></tr> </table> </div> <div data-bbox="619 694 1182 1003"> <div>rental_history.txt</div> <div>파일 편집 보기 H1 ≡ ...</div> <pre>jully Folding Bike B-Type 2 jully Electric Bike Ultra-EV 1</pre> </div>	Mountain Bike A-Type	20000	4	Folding Bike B-Type	50000	0	Electric Bike Ultra-EV	30000	7	Road Bike Speed-10	10000	0	Electric Bike C-Type	80000	4	Fixie Bike Fix-01	30000	0	Urban Bike Urban-Alpha	70000	6									
Mountain Bike A-Type	20000	4																													
Folding Bike B-Type	50000	0																													
Electric Bike Ultra-EV	30000	7																													
Road Bike Speed-10	10000	0																													
Electric Bike C-Type	80000	4																													
Fixie Bike Fix-01	30000	0																													
Urban Bike Urban-Alpha	70000	6																													
<div>자전거 대여 및 그 결과(2)</div>	<div data-bbox="426 1050 1374 1637"> <p>로그인 ID를 입력하세요: jully 사용자 모드입니다.</p> <p>=== 대여 가능 자전거 목록 ===</p> <table> <tr> <th>번호</th> <th>자전거 이름</th> <th>비용</th> <th>재고</th> </tr> <tr><td>0</td><td>Mountain Bike A-Type</td><td>20000</td><td>4</td></tr> <tr><td>2</td><td>Electric Bike Ultra-EV</td><td>30000</td><td>7</td></tr> <tr><td>4</td><td>Electric Bike C-Type</td><td>80000</td><td>4</td></tr> <tr><td>6</td><td>Urban Bike Urban-Alpha</td><td>70000</td><td>6</td></tr> </table> <p>대여할 자전거 번호 (-1 입력 시 대여 확정): 2 "Electric Bike Ultra-EV" 자전거가 대여 목록에 추가되었습니다. 대여할 자전거 번호 (-1 입력 시 대여 확정): -1</p> <p>=== 이번 세션 대여 내역 ===</p> <table> <tr> <th>번호</th> <th>자전거 이름</th> <th>수량</th> <th>비용</th> <th>합계</th> </tr> <tr> <td>0</td> <td>Electric Bike Ultra-EV</td> <td>1</td> <td>30000</td> <td>30000</td> </tr> </table> <p>총 결제 금액: 30000</p> </div> <div data-bbox="614 1684 1185 1957"> <div>rental_history.txt</div> <div>파일 편집 보기 H1 ≡ ...</div> <pre>jully Folding Bike B-Type 2 jully Electric Bike Ultra-EV 2</pre> </div>	번호	자전거 이름	비용	재고	0	Mountain Bike A-Type	20000	4	2	Electric Bike Ultra-EV	30000	7	4	Electric Bike C-Type	80000	4	6	Urban Bike Urban-Alpha	70000	6	번호	자전거 이름	수량	비용	합계	0	Electric Bike Ultra-EV	1	30000	30000
번호	자전거 이름	비용	재고																												
0	Mountain Bike A-Type	20000	4																												
2	Electric Bike Ultra-EV	30000	7																												
4	Electric Bike C-Type	80000	4																												
6	Urban Bike Urban-Alpha	70000	6																												
번호	자전거 이름	수량	비용	합계																											
0	Electric Bike Ultra-EV	1	30000	30000																											

자전거 대여
및
그 결과(3)

로그인 ID를 입력하세요 : kim
사용자 모드입니다.

=== 대여 가능 자전거 목록 ===

번호	자전거 이름	비용	재고
0	Mountain Bike A-Type	20000	4
2	Electric Bike Ultra-EV	30000	6
4	Electric Bike C-Type	80000	4
6	Urban Bike Urban-Alpha	70000	6

대여할 자전거 번호 (-1 입력 시 대여 확정): 4
"Electric Bike C-Type" 자전거가 대여 목록에 추가되었습니다.
대여할 자전거 번호 (-1 입력 시 대여 확정): -1

=== 이번 세션 대여 내역 ===

번호	자전거 이름	수량	비용	합계
0	Electric Bike C-Type	1	80000	80000

총 결제 금액: 80000

rental_history.txt		
파일	편집	보기
jully	Folding Bike B-Type	2
jully	Electric Bike Ultra-EV	2
kim	Electric Bike C-Type	1

4번 문제: 관리자 및 사용자 반납 기능

3단계까지 구현한 상태에서, 관리자 동작 및 사용자의 반납 기능을 추가하여 프로그램을 확장한다. 각각의 모드에서는 다음과 같이 적용된다.

- 관리자 모드: 사용자의 모든 대여 현황을 조회하고, 미반납 사용자들을 확인
- 사용자 모드: 대여중인 자전거 반납

아래와 같은 단계 및 조건으로 수행된다.

1. 관리자 모드로 로그인 했을 때, 전체 사용자의 대여 현황이 저장된 rental_history.txt를 모두 읽어 메모리에 저장한다.
 - 파일이 존재하지 않으면 에러 메시지를 출력하고, 종료한다.
 - 이때, 사용자 아이디, 자전거 번호, 자전거 이름, 해당 사용자가 대여중인 수량을 저장하기 위한 새로운 구조체 및 구조체 배열을 정의한다.
 - ◆ 자전거 번호의 경우 배열의 인덱스와 동일하게 부여한다.
2. 관리자에게 다음 메뉴를 제공한다.
 - 전체 사용자의 대여 현황 조회
 - ◆ "전체 사용자의 대여 현황 조회"를 선택한 경우, 메모리에 저장된 모든 대여 현황을 출력한다.
 - 미반납 사용자 조회
 - ◆ "미반납 사용자 조회"를 선택한 경우, 대여중인 자전거가 존재하는(대여중인 자전거 수량이 0이 아닌) 기록들을 찾아서 해당 사용자 아이디, 자전거 이름, 대여중인 수량을 출력한다.
 - ◆ 이때, 사용자별로 정렬하여 출력하지 않아도 된다.
3. 사용자 모드로 로그인 후, 다음과 같은 메뉴 선택지를 가진다.
 - 자전거 대여
 - ◆ 사용자가 "자전거 대여" 메뉴를 선택했을 때, 이전 단계에서 구현한 자전거 대여 과정을 수행한다.
 - 자전거 반납
 - ◆ 사용자가 "자전거 반납"선택 시 진행한다.
 - ◆ rental_history.txt을 읽어 해당 사용자의 대여 현황을 모두 메모리에 저장한다.
 - 파일이 존재하지 않으면 에러 메시지를 출력하고, 종료한다.
 - 이때, 이 사용자 아이디, 자전거 번호, 자전거 이름, 해당 사용자가 대여중인 수량을 저장하기 위하여 구조체를 이용하고, 구조체 배열을 정의한다.
 - 자전거 번호의 경우 배열의 인덱스와 동일하게 부여한다.
 - ◆ bikes_v2.txt파일에서 자전거 정보 목록을 읽어 메모리에 저장한다
 - ◆ 사용자의 대여 현황을 출력한다.

- 해당 사용자가 대여중인 자전거의 정보를 출력한다.
 - 자전거 번호, 자전거 이름, 사용자가 해당 자전거를 대여중인 수량을 출력한다.
- 대여중인 자전거 수량이 0인 경우 출력하지 않는다.
- ◆ 자전거 반납 진행
 - 사용자에게 반복적으로 반납할 자전거 번호와 반납할 자전거 수량을 입력받는다.
 - 사용자가 자전거 번호로 -1을 입력하는 경우, 반납 종료로 처리한다.
 - 선택된 자전거의 대여 중 수량과 대여 가능 수량(재고)을 수정한다.
 - 반납한 자전거 수량만큼 메모리의 자전거 리스트에서 해당 자전거 대여 가능 수량(재고)을 증가시킨다.
 - 반납한 자전거 수만큼 메모리에 저장된 해당 사용자의 해당 자전거 대여 중 수량을 감소시킨다.
 - 사용자가 입력한 반납 대수가 실제 대여 중인 대수보다 많을 경우, 관련 메시지를 출력하고, 가능한 수량만큼만 반납 처리한다.
 - 잘못된 자전거 번호의 경우 알맞은 메시지를 출력한 뒤 재입력받는다.
- ◆ 변경 내용을 rental_history.txt와 bikes_v2.txt에 반영한다.

이때, 자전거 리스트, 대여 현황은 **구조체 배열**로 관리하며, **반드시 동적 메모리 할당**을 이용해 관리해야 한다. 또한, **구조체 내부의 자전거 이름** 또한 **동적 메모리 할당**을 사용한다. (단순 고정 배열 시 감점)

구현 및 수정해야 할 함수:

본 단계에서는 함수 구조를 직접 설계하여 구현한다.

이전 단계에서 구현한 함수들은 재사용하여 기능을 완성하도록 한다.

결과 예시:

[사용자]
자전거 반납

로그인 ID를 입력하세요 : jully

--- 사용자 메뉴 ---
1. 자전거 대여
2. 자전거 반납
선택 : 2

=== jully 님의 미반납 목록 ===
[0] Folding Bike B-Type - 2대 미반납
[1] Electric Bike Ultra-EV - 2대 미반납
반납할 번호(-1 종료): 3
잘못된 번호입니다. 다시 입력하세요.
반납할 번호(-1 종료): 0
반납할 수량 입력: 3
반납 요청 수량이 대여 중인 대수보다 많습니다. 가능한 2대만 반납합니다.
"Folding Bike B-Type" 2대 반납 완료
반납할 번호(-1 종료): 1
반납할 수량 입력: 1
"Electric Bike Ultra-EV" 1대 반납 완료
반납할 번호(-1 종료): -1

rental_history.txt

파일 편집 보기 H1 ≡ ...

jully	Folding Bike B-Type	0
jully	Electric Bike Ultra-EV	1
kim	Electric Bike C-Type	1

bikes_v2.txt

파일 편집 보기 H1 ≡ ...

Mountain Bike A-Type	20000	4
Folding Bike B-Type	50000	2
Electric Bike Ultra-EV	30000	7
Road Bike Speed-10	10000	0
Electric Bike C-Type	80000	3
Fixie Bike Fix-01	30000	0
Urban Bike Urban-Alpha	70000	6

<p>[관리자] 전체 사용자 의 대여 현 황 출력</p>	<pre>로그인 ID를 입력하세요 : admin --- 관리자 모드 --- 1. 전체 대여 기록 조회 2. 미반납 사용자 조회 선택 : 1 === 전체 대여 기록 === jully - Folding Bike B-Type : 0대 jully - Electric Bike Ultra-EV : 1대 kim - Electric Bike C-Type : 1대</pre>
<p>[관리자] 미반납 사용 자 조회</p>	<pre>로그인 ID를 입력하세요 : admin --- 관리자 모드 --- 1. 전체 대여 기록 조회 2. 미반납 사용자 조회 선택 : 2 === 미반납 사용자 === jully - Electric Bike Ultra-EV : 1대 미반납 kim - Electric Bike C-Type : 1대 미반납</pre>

5번 문제: 사용자 보안 강화

지금까지는 단순히 admin/user의 아이디만 입력하면 각 모드로 진입 가능하게 설정되어 있었기 때문에, 1단계의 check_access()를 보안 버전으로 확장한다.

이를 위해 관리자는 프로그램을 실행할 때 main의 매개변수로 admin을 입력하게 되며, 이 경우 인증 키를 추가로 입력하여 접근 허용을 받는다.

사용자의 경우, 아이디 입력 후 비밀번호를 입력 받아야 하며, 함께 계정 상태 또한 고려되어야 한다.

아래와 같은 단계 및 조건으로 수행된다.

1. 관리자 모드로 진입하기 위해, 프로그램 실행 시 매개변수로 "admin"을 입력한다.
2. main 함수의 매개변수로 "admin"이 입력되었다면, 인증 코드를 입력받는다.
 - 관리자의 올바른 인증 코드는 admin.txt에 저장되어 있다.
 - 입력한 인증 코드가 admin.txt와 일치하지 않는 경우, 다시 입력받는다.
3. 사용자 모드로 진입하기 위해, 아이디와 비밀번호를 입력받는다.
 - 사용자 정보는 user_v2.txt에 존재한다.
 - ◆ 해당 파일에는 아이디, 비밀번호, 계정 상태가 저장되어 있다.
 - ◆ 각 정보는 'wt'(tab)으로 구분되어 있다.
 - 입력받은 아이디가 user_v2.txt에 존재하지 않는 경우, 재입력받는다.
 - 아이디가 존재하는데, 비밀번호가 틀린 경우 재입력받는다.
 - 아이디와 비밀번호가 user_v2.txt의 정보와 일치하고, 계정 상태가 활성 상태이면 사용자 모드로 진입한다.
 - 비밀번호가 3회 틀릴 경우, 계정을 잠금 처리하고, 해당 사용자의 계정 상태를 user_v2.txt에 업데이트한 후, 종료한다.
 - 계정이 잠금 처리된 경우, 아이디와 비밀번호가 user_v2.txt 정보와 일치해도 계정 상태가 활성 상태가 아니면 종료한다.
 - 로그인 결과에 따라 적절한 안내 메시지를 출력한다.

구현 및 수정해야 할 함수:

read_admin():

- admin.txt에서 인증 코드를 읽고, 프로그램 사용자의 입력과 비교하여 결과를 반환한다.

read_user():

- user_v2.txt를 읽어와서 아이디/비밀번호/계정 상태를 메모리에 저장한다.
- 사용자가 입력한 아이디가 user_v2.txt에 존재하는지 확인한다.
 - ◆ 존재하는 경우, 비밀번호를 입력받는다.
 - ◆ 존재하지 않는 경우, 아이디를 다시 입력받는다.
- 비밀번호를 입력받은 경우
 - ◆ 아이디에 해당하는 비밀번호인 경우, 계정 상태가 활성 상태인지 확인한다.
 - 계정 상태가 활성 상태라면, 사용자 모드 진입을 위해 결과를 반환한다.
 - 계정 상태가 비활성 상태라면, 프로그램을 종료한다.
 - ◆ 아이디에 해당하는 비밀번호가 아닌 경우, 다시 입력받는다.
 - 만약 세 번 틀린다면, 계정 상태를 잠금 상태로 변경한다.
 - 사용자의 변경된 상태를 user_v2.txt에 저장하고, 종료한다.

check_access():

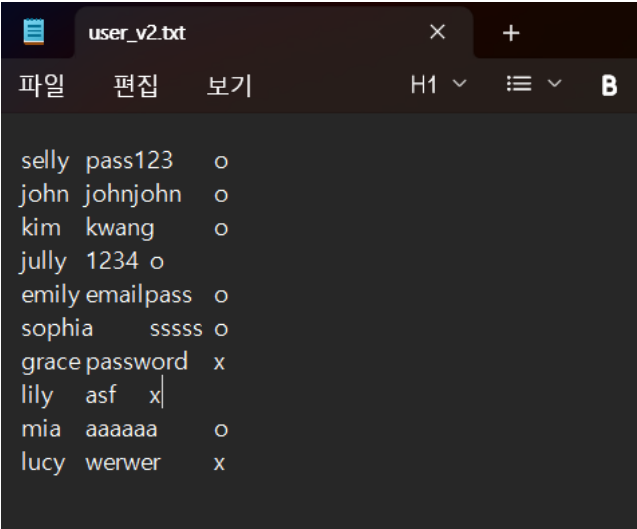
- 1번 문제의 check_access()를 확장한 버전이다.
- main()에서 전달받은 값에 따라 내부에서 read_admin()과 read_user()를 적절히 호출한다.

main():

- main 함수는 실행 시 전달된 매개변수를 확인하여, 관리자로 실행되는지 판단한다.
- 매개변수가 "admin"인 경우, main은 check_access()에 관리자 모드 정보를 넘겨 호출한다.

결과 예시:

관리자 모드 진입	<pre> 관리자 모드로 접속을 시도했습니다. 관리자 인증 코드를 입력하세요 : XXXXX 잘못된 코드입니다. 다시 입력하세요. 관리자 인증 코드를 입력하세요 : XD2Za30 관리자 인증 성공! 관리자 모드입니다. </pre>
-----------	---

<p>사용자 모드 진입 (로그인 성공)</p>	<pre>로그인 ID를 입력하세요 : wrong 존재하지 않는 아이디입니다 . 다시 로그인 시도 . 로그인 ID를 입력하세요 : jully 비밀번호 입력 : 1111 비밀번호가 틀렸습니다 . (1/3) 비밀번호 입력 : 1234 로그인 성공 !</pre>
<p>사용자 모드 진입 (실패 - 계정 잠김)</p>	<pre>로그인 ID를 입력하세요 : lily 비밀번호 입력 : asdf 비밀번호가 틀렸습니다 . (1/3) 비밀번호 입력 : aaaa 비밀번호가 틀렸습니다 . (2/3) 비밀번호 입력 : ssss 비밀번호가 틀렸습니다 . (3/3) 3회 실패로 계정이 잠깁니다 . 계정이 잠겨 프로그램을 종료합니다 . 로그인 ID를 입력하세요 : lily 비밀번호 입력 : asf 계정이 잠겨 있습니다 . 프로그램을 종료합니다 .</pre>
<p>user_v2.txt 변화</p>	 <pre>user_v2.txt 파일 편집 보기 H1 B selly pass123 o john johnjohn o kim kwang o jully 1234 o emily emailpass o sophia sssss o grace password x lily asf x mia aaaaaa o lucy werwer x</pre>