스터디룸 예약 프로그램

A04 팀

201910184 김민기
201911286 최민호
201911294 홍승택
202111294 박성준
202111309 손 본
202111332 유의진

목 차

1 개요

본 프로그램은 스터디룸 예약을 위한 사용자 기능과 스터디룸 관리를 위한 관리자 기능이 존재합니다. 명령어와 인자들을 문자열로 입력받아, 프로그램에서 지원하는 기능을 사용할 수 있습니다.

사용자는 스터디룸을 예약할 수 있고, 자신의 예약 현황을 확인할 수 있습니다. 관리자는 회원 정보를 조회할 수 있으며, 해당 회원의 예약 정보를 확인할 수 있습니다.

2 용어

이 문서에서 사용할 용어들의 의미를 약속합니다. 여기서 약속하는 용어들은 세간에서 통용되지 않는 의미일 수 있으며, 세간에서 사용하지 않는 용어가 새롭게 정의될 수 있습니다.

사용자(또는 회원)

이 프로그램의 예약 기능을 이용하는 사람들을 통칭하여 사용자(또는 회원)라 부릅니다.

관리자(또는 Admin)

이 프로그램을 관리하고, 운용하는 사람을 의미합니다. 이 프로그램은 사용자와 관리자가 사용할 수 있는 명령어가 구분되어 있으며, 9절부터 설명되어 있습니다.

이용자

사용자와 관리자를 통칭하는 용어입니다.

실행파일

배포되는 파일인 studyRoom.exe 파일을 의미합니다.

홈 경로

실행파일이 위치해 있는 디렉토리(폴더)의 경로를 홈 경로라고 부릅니다. 홈 경로를 기준으로 파일들이 생성, 저장됩니다. 이 문서에서 경로명 전체 혹은 시작 부분에 "{%HOME}"이라고 표기된 부분은 홈 경로를 의미합니다.

개행

키보드 자판 중 엔터 키(U+000A, Line Feed) 또는 리턴 키(U+000D, Carriage Return)으로 입력되는 문자를 의미합니다. 프롬프트 입출력이나 파일 형식의 예시에서는 ⏎으로 개행을 표시합니다.

데이터 파일(혹은 파일)

이 문서의 기획 대상인 프로그램이 데이터 정보를 저장 및 읽는데 사용되는 텍스트 문서 파일입니다. 확장자는 "txt"이며, 아래 6절에서 자세히 다룹니다.

숫자

표준 키보드로 직접 입력할 수 있는 아라비아 숫자들을 의미합니다. 유니코드에서 U+0030(0)부터 U+0039(9)에 해당하는 숫자들만 의미합니다. 예를 들어, "13"은 숫자 2개로 이루어져 있습니다.

공백

스페이스(SPC, U+0020)로 입력되는 문자를 의미합니다. 이 문서에서 " " 또는 "␣"로 표현됩니다.

문자

이 문서에서 "문자"란 Unicode에 존재하는 문자만을 의미합니다.

문자열

문자열이란 0개 이상의 문자의 순차 수열입니다. 이 문서에서 특정 단어나 문자열을 강조할 때에는 양끝에 "을 붙이기도 합니다. 즉, "abcd"는 문자열 abcd를 의미합니다.

(문자열의) 길이

문자열 속에 들어있는 문자의 개수입니다.

영문자(또는 대소문자)

a~z(U+0061 ~ U+007A), A~Z(U+0041+005A)에 해당하는 문자를 의미합니다. 또한 "소문자"는 a~z(U+0061 ~ U+007A)만을, "대문자"는 A~Z(U+0041+005A)만을 의미합니다.

명령어

사용자가 명령창에서 입력한 문자열에서 첫 번째 문자부터 공백이 나타나기 전까지의 부분 문자열은 명령어를 의미합니다. 해당 프로그램이 지원하는 기능을 실행하기 위해서는, 그에 맞는 명령어로 시작하는 문자열을 입력해야 합니다. 프로그램에서 사용하는 모든 명령어는 대소문자 구분 없이 사용합니다. 예를 들어, "book", "Book", "bOOk" 등의 입력은 모두 동일한 명령어로 인식됩니다. 명령어 종류와 그 기능은 9절부터 서술됩니다.

인자

사용자가 명령어를 입력하여 원하는 동작을 수행시킬 때, 해당 명령과 함께 지정하는 단어를 의미합니다. 인자는 명령어 뒤에 공백 1개로 구분됩니다. 이때 인자 순서에 따라 "인자1", "인자2"와 같이 뒤에 숫자를 붙여서 구분합니다. 예를 들어, "Book␣12␣abc"의 경우 명령어는 "Book", 인

자1은 "12", 인자2는 "abc"입니다. 또한 "Book␣␣"의 경우 명령어는 "Book", 인자1은 "␣"입니다.

인덱스

문자열에서 특정 문자를 지칭하기 위한 요소로, 왼쪽부터 1, 2, 3, ... 의 순서로 1씩 증가하는 양의 정수 값을 할당 받습니다. 예를 들어, 000839 에서 '8'의 인덱스는 4입니다.

프롬프트(창)

프로그램 실행 시 실행되는 창(화면)을 의미합니다. 사용자는 프롬프트를 통해 프로그램의 출력 결과를 확인하거나, 명령(어)을 입력하실 수 있습니다.

표준 형식

프로그램 내부에서 데이터 요소를 다룰 때 사용하는 형식입니다. 데이터 저장이나 화면 출력 시 표준 형식으로 저장되고, 데이터 요소를 사용자가 입력 시에도 표준 형식으로 변환하여 의미를 해석합니다.

선행 0

"0"으로 시작하는 문자열 중, 첫 문자를 포함하여 모든 문자가 "0"으로 이루어진 부분 문자열을 의미합니다. 예를 들어, "0013a"의 선행 0은 "00"이며, 선행 0을 무시한 문자열은 "13a"입니다.

날짜 구분자

날짜를 연(year), 월(month), 일(day)를 구분하기 위한 문자입니다. 날짜 구분자에는 "/", "-"가 있으며, 날짜를 표현하기 위해 날짜 구분자를 혼용하여 사용할 수 있습니다. 예를 들어, "2020/12-31"은 "2020년 12월 31일"을 의미합니다.

시간 구분자

시간을 시(hour), 분(minute)을 구분하기 위한 문자입니다. 시간 구분자에는 "/", "-", ":"가 있습니다. 예를 들어, "21:23"은 "21시 23분"을 의미합니다.

명령창

사용자가 명령어와 함께 인자를 사용하여, 특정 기능을 동작 시키기 위해 표준 키보드로 문자열을 입력하는 창을 의미합니다. 명령창은 메인 명령창, 사용자 명령창, 관리자 명령창으로 나눠집니다. 각 명령창에서 사용할 수 있는 명령어는 다릅니다. 이에 대한 자세한 설명은 7절에서 서술합니다.

1달(또는 1개월)

1달 혹은 1개월은 30일을 의미합니다. 즉, 1달, 1개월, 한 달, 30일은 모두 같은 의미이며, 한 달 뒤는 30일 뒤를 의미합니다. 3개월이란 3달을 의미합니다.

예약

이 프로그램에서 사용자가 스터디룸을 특정 시간대에 사용할 것임을 미리 명시해두는 것을 의

미합니다. 일반적으로 사용되는 "예약"과 동일한 의미를 가지지만, 본 프로그램에서는 더 많은 사용자가 스터디룸을 이용할 수 있도록 특수한 예약 최적화 방식을 채택합니다. 이에 대한 자세한 설명은 13절에서 서술합니다.

3 기본 사항 및 환경

3.1 개발 환경 및 작동 환경

본 프로그램은 MS Windows 10 환경에서 언어 C++ 20, 컴파일러 gcc/g++ 12.2.0을 이용하여 개발되었습니다. 아래의 작동 환경에서는 확실히 정상 작동하지만, 다른 운영체제, 혹은 Windows의 다른 버전, 다른 프롬프트를 사용할 경우 작동될 수도 있지만 보장되지는 않습니다.

- MS Windows 10에서 cmd 창, PowerShell 창에서 확실히 정상 작동합니다.

또한 시스템 또는 프롬프트창에서 유니코드를 지원하지 않으면, 문자가 정상적으로 출력되지 않을 수 있으며, 정상 작동하지 않을 수 있습니다. 또한 터미널 글꼴 및 문자 크기 등에 따라서 출력 간격, 배치 등이 정상적으로 출력되지 않을 수 있습니다.

3.2 프로그램 구성

이 프로그램의 배포 압축 파일(.zip)에는 주 실행파일인 studyRoom.exe 파일과, 프로그램의 초기 정보를 담고 있는 "meta.txt" 파일이 있습니다. 이 프로그램의 주 실행파일이 정상적으로 실행되면, "{%HOME}/resource/meta.txt"에 파일이 존재하는지, 존재한다면 올바른 형식을 가진 파일인지를 프로그램 내부에서 검증합니다. 만일 해당 경로에 파일이 존재하지 않거나, 올바른 저장 형식이 아니면, 오류 메시지를 출력하고 프로그램이 종료됩니다. 이 파일은 사용자가 직접 수동으로 지우기 전까지는 (즉, 프로그램을 통해서는) 지워지지 않습니다.

3.3 프로그램 실행

프로그램 저장 및 실행하고 싶은 경로에 압축 파일을 풉니다. 해당 경로에 있는 주 실행파일인 "studyRoom.exe"를 탐색기 등 GUI 파일 관리창에서 더블클릭하여 실행할 수 있습니다.

추가로 MS Window의 경우:

- cmd창, PowerShell 창, 혹은 그에 준하는 터미널에서 주 실행 파일의 절대 경로나 상대 경로를 입력하여 실행할 수 있습니다. 이때 확장자(.exe)는 생략할 수도 있습니다.

```
C:₩work₩SoPrj> ./studyRoom[.exe]⏎
```

- 주 실행 파일이 위치한 디렉토리에서 start 명령어를 이용하여 파일을 실행할 수 있습니

다. 이때 확장자(.exe)는 생략할 수도 있습니다.

```
C:₩work₩SoPrj> start studyRoom[.exe]↵
```

4 프로그램 사용 흐름도

프로그램 사용 흐름, 즉, 프로그램 사용자 관점에서 프로그램을 사용하는 단계별 경로는 “그림 1: 사용 흐름도”와 같습니다. (데이터 파일을 외부 편집기로 직접 편집하는 방식은 허용된 사용 방식이지만, 프로그램 실행 중의 사용 흐름과는 별개이므로 흐름도에 포함시키지 않았습니다.)

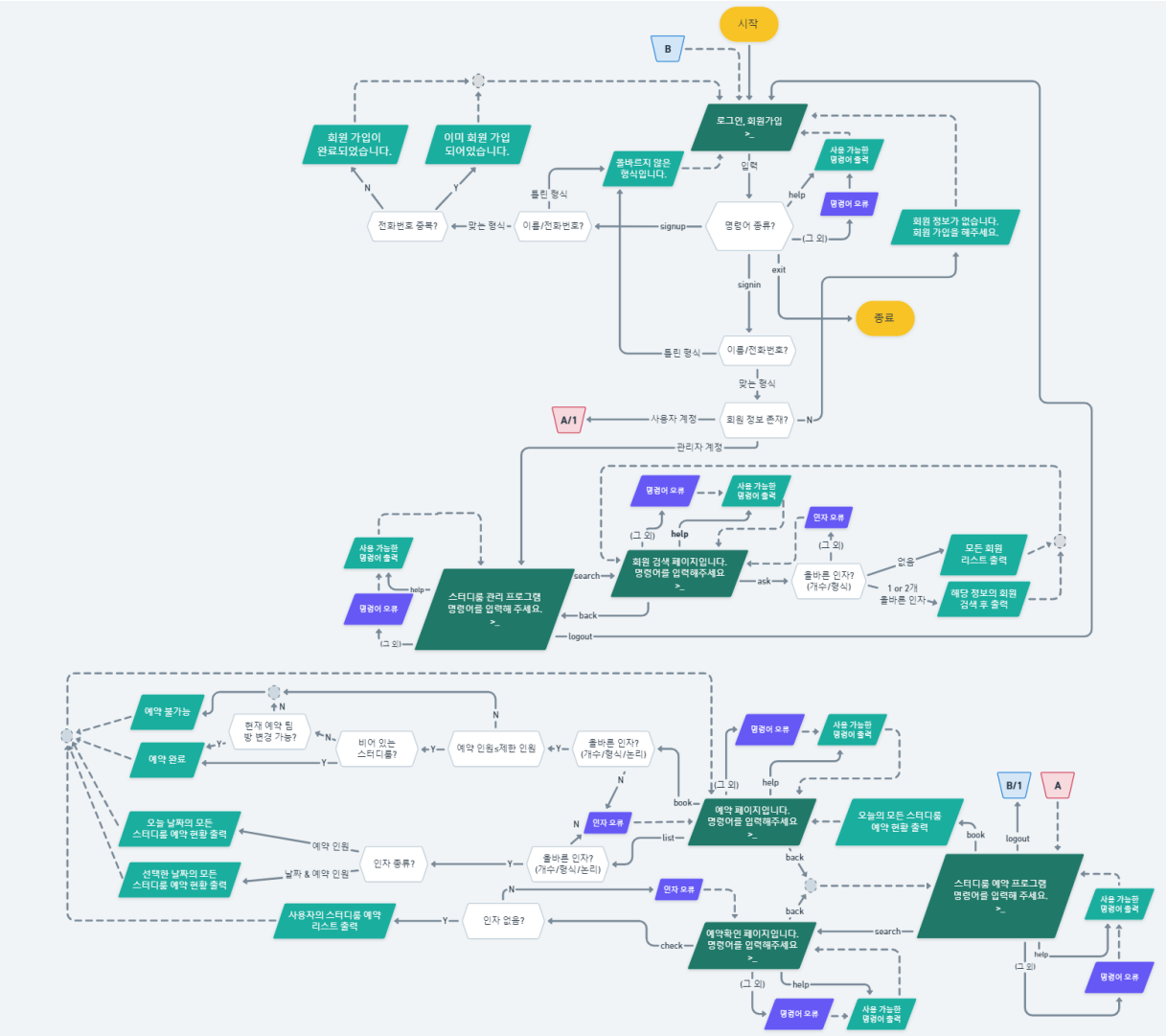


그림 1 : 사용 흐름도

5 데이터 요소

사용자가 문자열을 입력할 때 지켜야할 문법 형식과 의미 규칙을 정합니다. 또한 프로그램 내에서 사용될 표준 형식을 정합니다.

동치 비교: 표준 형식으로 변환된 데이터 요소의 문자열이 완전히 일치하면,
같은 데이터 요소로 간주합니다.

5.1 사용자

사용자는 프로그램에 자신의 정보를 등록하고 예약시스템을 이용하는 주체입니다. 사용자는 이름과 고유한 전화번호, 고유한 사용자 번호를 가지고 있습니다.
이때 어떤 두 사용자의 전화번호 또는 사용자 번호가 같다면, 동일한 사용자입니다.

5.1.1 이름

이 문서에서의 이름은 사용자의 인명과 같습니다.

표준 형식: 소문자로만 이루어진 길이 2 이상, 30 이하의 문자열

문법 형식: 아래의 조건을 모두 만족하는 문자열은 문법적으로 올바른 이름 표현입니다.

- 길이가 2 이상, 30 이하인 문자열
- 문자열은 영문자로만 이루어야 한다.
  즉, 공백이나 특수문자, 숫자 등 영문자가 아닌 문자는 문자열에 포함될 수 없습니다.

표준 형식 변환: 문자열에 있는 모든 대문자를 소문자로 변환합니다.
예를 들어, "AiYu"은 "aiyu"로 변환됩니다.

의미 규칙: 이름에는 추가적인 의미 규칙이 없습니다. 같은 이름을 갖는 둘 이상의 사용자가 존재해도 오류가 아닙니다.

5.1.2 전화번호

전화번호는 이용자들을 구분하는 하나의 식별자로 사용됩니다. 따라서 사용자의 전화번호는 유일(unique)해야 합니다.

표준 형식: 숫자로만 이루어진 길이 7 이상, 11 이하의 문자열

문법 형식: 아래의 조건을 모두 만족하는 문자열은 문법 상 올바른 전화번호 표현입니다.

- 길이가 7 이상, 20 이하인 문자열

- '-'(하이픈, U+002D)과 숫자로만 이루어진 문자열
- 첫 문자와 마지막 문자에 '-'(하이픈)은 올 수 없습니다.
- 숫자의 개수는 7개 이상, 11개 이하여야 합니다.

예를 들어, "0-1-0-2-6-3-5-0-3-0-3", "01—0—123-4-1234"도 모두 문법적으로 올바른 전화번호입니다.

또한 전화번호는 사용자를 구분하는 용도로 사용되기 때문에 "02-1234-1234" 또는 "010-123-1234", "070-1234"과 같이 문법 규칙에만 맞는다면, 문자열이 어떤 숫자로 시작하는지는 이 프로그램에서 중요하지 않습니다.

표준 형식 변환: 문자열 중 '-'(하이픈)은 무시되고, 숫자들만 변환됩니다.

예를 들어, "0-1-0-2-6-3-5-0-3-0-3"의 경우 "01026350303"로 변환됩니다.

의미 규칙: 중복된 전화번호를 가진 이용자는 있을 수 없습니다. 즉, 모든 이용자는 고유한 전화번호를 가져야 합니다.

5.1.3 사용자 번호

사용자 번호는 회원가입 완료 시 자동으로 생성되는 고유한 숫자입니다. 1부터 시작하여 1씩 증가하면서 사용자 번호가 생성됩니다.

표준 형식: 숫자로만 이루어진 길이 1 이상, 4 이하의 문자열

문법 형식: 숫자로만 이루어진 길이 1 이상, 4 이하의 문자열이면 문법적으로 올바른 사용자 번호입니다.

표준 형식 변환: 문자열 중 선행 0은 무시되어 변환됩니다.

예를 들어, "1", "001", "0001"은 모두 "1"로 변환됩니다.

의미 규칙: 중복된 사용자 번호를 가진 사용자는 있을 수 없습니다. 즉, 모든 사용자는 고유한 사용자 번호를 가져야 합니다.

5.2 방 번호

스터디룸마다 가지고 있는 고유의 숫자입니다. 스터디룸의 방 번호는 고유하며 사용자가 임의로 조작할 수 없고, 1부터 9까지의 자연수 값만 가질 수 있습니다.

표준 형식: 숫자로만 이루어진 길이가 1인 문자열

문법 형식: 0이 아닌 숫자로 이루어진, 길이가 1인 문자열만 올바른 문법 형식입니다.

의미 규칙: 중복된 방 번호를 가진 스터디룸은 있을 수 없습니다.

5.3 예약 번호

모든 사용자의 예약을 구분하는 고유의 숫자입니다. 스터디룸을 예약할 경우, 예약 번호는 1부터 1씩 증가되면서 자동으로 할당됩니다.

표준 형식: 숫자로만 이루어진 길이가 1 이상인 문자열

문법 형식: 숫자로만 이루어진 문자열만 올바른 문법 형식입니다.

의미 규칙: 중복된 예약 번호를 가진 예약은 있을 수 없습니다.

5.4 날짜

날짜는 특정 일(day)을 표현하기 위한 데이터 요소를 의미합니다.

표준 형식: <4자리 숫자>-<2자리 숫자>-<2자리 숫자>로 이루어진 문자열

이때, 앞 4자리 숫자는 연(year), 중간 2자리 숫자는 월(month), 끝 2자리 숫자는 일(day)로 해석합니다.

예를 들어, "2022-10-09"는 표준 형식으로 나타낸 날짜이며, 2022년 10월 09일을 의미합니다.

문법 형식: 아래 두 조건 중 한 개의 조건에 만족하는 문자열은 문법상 올바른 날짜입니다.

- <2개 또는 4개의 숫자><날짜 구분자 1개><1개 또는 2개의 숫자><날짜 구분자 1개><1개 또는 2개의 숫자>로 이루어진 문자열
- 숫자로만 이루어진 길이 6 또는 8인 문자열

표준 형식 변환:

1. 날짜 구분자가 포함된 경우의 해석은 다음과 같습니다.
   - 첫 번째 날짜 구분자 이전의 숫자 : 연(year)으로 해석합니다.
     - 숫자 2개로 이루어져 있다면, 앞에 "20"을 붙여 변환합니다.
     - 숫자 4개로 이루어져 있다면, 그대로 변환합니다.
   - 첫 번째 구분자부터 두 번째 구분자 전까지의 숫자 : 월(month)로 해석합니다.
     - 숫자 1개로 이루어져 있다면, 앞에 "0"을 붙여 변환합니다.

- 숫자 2개로 이루어져 있다면, 그대로 변환합니다.

■ 두 번째 구분자부터 문자열 끝까지(개행 전까지): 일(day)로 해석합니다.
- 숫자 1개로 이루어져 있다면, 앞에 "0"을 붙여 변환합니다.
- 숫자 2개로 이루어져 있다면, 그대로 변환합니다.

2. 날짜 구분자가 사용되지 않은 문자열의 해석은 다음과 같습니다.
■ 문자열 길이가 6이라면, 인덱스 1, 2는 연(year), 3, 4는 월(month), 5, 6은 일(day)로 해석합니다.
예를 들어, 문자열이 "221009"이면, "2022-10-09"로 변환됩니다.
■ 문자열 길이가 8이라면, 인덱스 1, 2, 3, 4는 연(year), 5, 6는 월(month), 7, 8은 일(day)로 해석합니다.
예를 들어, 문자열이 "20221009"이면, "2022-10-09"로 변환됩니다.

의미 규칙: 그레고리력에 부합하는 날짜가 아니면 오류입니다.

5.5 시간

특정 시간(분 단위)를 표현합니다. 엄밀하게 말하자면 시간의 어느 한 시점을 의미하므로 시각(時刻)이라고 표현하는 것이 더 정확하지만, 이 문서에서는 시간과 시각을 같은 의미로 사용합니다.

5.5.1 시간

'시'와 '분' 이루어진 문자열을 24시간제로 표현하기 위한 데이터 요소를 의미합니다.

표준 형식: <숫자 2개>:<숫자 2개>
이때, 앞 2개 숫자는 시(hour), 끝 2개 숫자는 분(minute)로 해석합니다.
예를 들어, "13:04"는 표준 형식으로 나타낸 시간이며, 13시 04분을 의미합니다.

문법 형식: 아래 형식 중 하나를 만족하는 문자열은 문법적으로 올바른 시간입니다.
■ <1개 또는 2개의 숫자><시간 구분자 1개><1개 또는 2개의 숫자>
■ 길이가 1 이상, 4 이하인 숫자로만 이루어진 문자열

표준 형식 변환:
1. 시간 구분자가 포함된 경우
■ 구분자는 ":"(콜론, U+003A)으로 변환합니다.
■ 구분자 앞, 뒤에 있는 숫자는 그대로 변환합니다.
2. 시간 구분자가 사용되지 않은 문자열의 해석은 다음과 같습니다.
■ 문자열의 길이가 1이나 2라면, 시(hour)로 변환합니다. 이때 분은 0으로 변환합니다.
예를 들어, "1"은 "01:00"로, 14는 "14:00"로 변환합니다.
■ 문자열의 길이가 3이면, 문자열 맨 앞에 "0"을 붙여서 문자열 길이가 4인 것(바로 아래 조건)과 같이 변환합니다.

예를 들어, "330"은 "0330"으로 변환 후, 문자열의 길이가 4인 것과 같이 해석합니다.

- 문자열의 길이가 4이면, 인덱스 1, 2를 시로 변환하고, 인덱스 3, 4를 분으로 변환합니다.
  예를 들어, 1139는 "11:39"로 변환합니다.

표준 형식으로 변환 후 '분'이 30분 단위가 아니라면, 아래 5.5.2의 시간 보정 규칙에 따라 자동으로 30분 단위의 시간으로 보정됩니다.

의미 규칙: 해석 및 보정의 결과가 24시간 표현으로 가능한 시간이 아니면 오류입니다. 가능한 시간이란 '시'는 0~23까지의 숫자이고 '분'은 0~59까지의 숫자를 의미합니다.

5.5.2 시간 보정

본 프로그램에서 가정한 스터디룸 매장은 각 스터디룸의 사용시간을 30분 단위로 관리합니다. 따라서 사용자의 편의를 위해, 위에서 정의한 시간으로부터 가장 가까운 30분 단위의 시간으로 변환해줍니다.

보정 규칙: 아래와 같은 순서로 보정을 진행합니다.

1. 분(minute)을 60으로 나눈 몫이 1 이상이라면, 시(hour)에 몫을 더하고, 분에는 나머지를 저장합니다.
   예를 들어, 시간이 "09:78"이라면 "10:18"으로 보정합니다.
2. 해석된 '분(minute)' 값에 대하여 다음과 같은 규칙을 따라 30분 단위의 시간으로 보정됩니다.
   - 00 ~ 14 → 00
   - 15 ~ 44 → 30
   - 45 ~ 59 → 시(hour)에 1을 더하고, 분은 0으로 변환

   예를 들어, 시간이 "09:08"이라면 "09:00"으로 보정합니다.

5.6 예약 인원

각 예약에서 스터디룸을 사용할 인원 수입니다.

표준 형식: <숫자 1개> 또는 <숫자 2개>

표준형식으로 변환하는 경우 예약인원의 길이가 2일 때, 선행 0은 무시됩니다.
예를 들어, 예약인원이 "09"라면 "9"로 보정하며, "00"이라면 "0"으로 보정합니다.

문법 형식: 길이가 1 이상, 2 이하인 숫자로만 이루어진 문자열은 문법적으로 올바른 예약 인원입니다.

의미 규칙:

- 예약 인원은 0명이 될 수 없습니다.
- 스터디룸이 허용하는 최대 인원수보다 더 많은 인원은 예약인원이 될 수 없습니다.

6 데이터 파일

6.1 파일 종류

이 프로그램이 사용하는 데이터 파일은 4 5 종류이며, 모두 텍스트 파일(.txt)입니다.

이때 [YYYYMMDD]는 특정 날짜를 의미합니다. "{%HOME}/book/" 내에는 여러 데이터 파일이 존재할 수도 있고, 그 파일 이름과 확장자는 "20221005.txt", "20221006.txt" 과 같은 형식입니다.

또한 [userId]는 사용자 번호를 의미합니다. 위와 마찬가지로 "{%HOME}/user/" 폴더 내에는 여러 데이터 파일이 존재할 수도 있고, 그 파일 이름과 확장자는 "1.txt", "2.txt" 과 같은 형식입니다.

경로 및 파일 이름	설명	예시
/book/[YYYYMMDD].txt	특정 날짜에 대한 예약 파일	/book/20220928.txt
/resource/userdata.txt	모든 사용자들의 정보를 저장하는 파일	/resource/userdata.txt
/user/[userId].txt	해당 사용자의 예약 내역 저장 파일	/user/13.txt
/resource/meta.txt	자원의 관리를 용이하게 하기 위한 메타 데이터 ■ 관리자 계정 정보 ■ 사용된 예약 번호 개수 ■ 사용된 사용자 번호 개수 ■ 스터디룸 별 제한 인원	/resource/meta.txt
/resource/resernum.txt	예약 번호 당 예약 인원을 저장하는 파일	/resource/resernum.txt

6.2 파일 저장 형식

이 절에서 "행"은 txt 파일에서 개행으로 구분되는 라인 하나를 의미합니다. 즉, txt 파일의 시작은 항상 1행에서 시작합니다. 행 번호는 해당 행의 순번을 의미하며, 개행 문자가 등장할 때마다 행 번호가 올라갑니다. 예를 들어, 1행, 2행, 3행과 같이 표현합니다.

또한 "요소"는 각 행에서 탭(Horizontal Tabulation, U+0009)으로 구분된, 데이터 값을 의미합니다.

공통 파일 구성 규칙

- 각 요소들은 탭으로 구분되어 있습니다.
- 각 요소는 사전에 정의한 표준 형식을 따릅니다.

6.2.1 각 날짜 예약 정보 파일

해당 날짜의 예약 정보를 저장하는 파일입니다.

파일 경로: {%HOME}/book/[YYYYMMDD 형식의 날짜].txt

파일 구성 규칙

- n행은 n번 스터디룸의 예약 정보를 담고 있습니다.
- 각 행의 k번째 요소는 "매장 운영시간 + 30*(k-1)분 ~ 매장 운영시간 + 30*k분"의 예약 정보를 담고 있습니다.
- 예약 정보의 기본 세팅 값은 0입니다.
  즉, 예약 정보가 0이면 예약이 가능합니다.
- 예약 성공 시, 예약 대상의 예약 정보에는 예약 번호가 저장됩니다.

예시[1]

```
<20220905.txt>
<09:00~09:30> <09:00~09:30> …(생략)… <19:30~20:00>에 해당하는 예약 정보
<1번방>0      0      0      0      0      0      …(생략)…⏎
<2번방>0      41     12     0      0      0      …(생략)…⏎
<3번방>0      0      0      0      0      0      …(생략)…⏎
(생략)
```

6.2.2 유저 데이터 종합 파일

모든 사용자들의 정보를 저장하는 파일

파일 경로: {%HOME}/resource/userdata.txt

파일 구성 규칙

- 한 행마다 한 사용자의 정보를 의미합니다.
- 각 행은 순서대로 사용자 번호, 사용자 이름, 사용자 전화번호가 탭으로 구분되어 있습니다.
- 사용자 번호 기준 오름차순 정렬되어 있습니다.

예시

```
[사용자 번호]   [사용자 이름]   [사용자 전화번호]
1       honggildong     010123456789⏎
2       nagildong       010987654321⏎
```

[1] 이 문서의 예시에 포함된 녹색 텍스트는 이해를 돕기 위한 설명이며, 실제 출력/저장되지 않습니다.

6.2.3 사용자 개인 데이터 파일

해당 사용자의 예약 내역 저장 파일입니다.

파일 경로: {%HOME}/user/[사용자 번호].txt

파일 구성 규칙

- 한 행은 한 개의 예약 정보를 의미합니다.
- 각 행은 순서대로 예약 번호, 예약 날짜, 방 번호, 시작 시각, 종료 시각이 탭으로 구분되어 있습니다.
- 예약 정보 정렬에 사용되는 규칙 우선순위는 아래와 같습니다. 우선순위가 높은 정보의 행 번호가 작습니다.
    1. 예약 날짜가 빠른 순
    2. 시작 시각이 빠른 순
    3. 종료 시각이 빠른 순
    4. 방 번호가 작은 순
- 기존의 예약이 예약 최적화로 인해 2개 이상으로 분리될 경우, 해당 예약 번호가 저장된 행에 시작시간, 종료시간, 방 번호가 시간 순으로 나열되어 저장됩니다.

예시

```
[예약 번호]  [예약 날짜] [시작시각] [종료시각] [방 번호] [시작시각] [종료시각] [방 번호] ..
12      2022-10-09     14:00   14:30   1       14:30   15:30   6       15:30   16:00   1⏎
14      2022-10-11     09:00   10:00   2⏎
13      2022-10-11     09:30   10:00   3⏎
9       2022-10-11     09:30   10:00   4       10:00   10:30   7⏎
11      2022-10-11     09:30   10:30   5⏎
```

6.2.4 메타 데이터 파일

파일 경로: {%HOME}/resource/meta.txt

파일 구성 규칙

- 1행에는 관리자 이름, 전화번호가 저장되어 있습니다.
- 2행에는 현재 가입된 사용자 수가 저장되어 있습니다.
- 3행에는 현재 사용된 예약 번호 수가 저장되어 있습니다.
  예약할 때마다 1씩 증가하기 때문에 해당 숫자는 지금까지 예약된 수와 동일합니다.
- 4행에는 1~9번 스터디룸의 제한 인원이 구분되어 저장되어 있습니다.

예시

```
Honggildong    01011112222⏎ [관리자 이름] [전화번호]
4⏎ [현재 회원 수]
13⏎ [현재까지 예약건수]
3       2       4       6       6       8       …(생략)… ⏎[스터디룸별 제한 인원]
```

초기 배포될 때는 아래와 같은 내용을 갖습니다.

```
admin  01012345678⏎ [관리자 이름] [전화번호]
0⏎ [현재 회원 수]
0⏎ [현재까지 예약건수]
3      2      4      6      6      8      …(생략)… ⏎[스터디룸별 제한 인원]
```

6.2.5 예약 인원 데이터 파일

파일 경로: {%HOME}/resource/resernum.txt

파일 구성 규칙

- 한 개의 행에는 한 개의 예약 정보가 존재합니다.
- 각 행은 예약 인원과 해당하는 사용자 번호가 탭으로 구분되어 저장됩니다.
- N번째 행에는 예약 번호 N번에 대한 정보를 담고 있습니다.

예시

```
          [예약 인원]   [사용자 번호]
<예약 번호 1번>1        4⏎
<예약 번호 2번>2        2⏎
<예약 번호 3번>3        6⏎
<예약 번호 4번>4        4⏎
```

7 명령창

이 절에서는 프로그램의 기능을 동작하는 명령(어)와 오류를 알아보기 전에, 명령창에 대해 설명합니다.

이 프로그램은 크게 메인 명령창, 사용자 명령창, 관리자 명령창으로 나눠집니다. 메인 명령창에서 사용자로 로그인하면 사용자 명령창으로 이동하고, 관리자로 로그인하면 로그인 명령창으로 이동합니다. 아래는 각 창의 프롬프트 예시입니다.

```
■ 메인 명령창
Main >

■ 사용자 명령창
{username} >

■ 관리자 명령창
Admin >
```

사용자 명령창 예시에 보이는 {username}에는 현재 로그인 된 사용자의 이름이 들어갑니다. 만약 사용자 이름 문자열의 길이가 8 초과일 경우, 앞선 8 개의 문자만 표시합니다.
예를 들어, 사용자 이름이 "honggildong"일 경우에 사용자 명령창에서는 "honggild > " 로 출력됩니다.

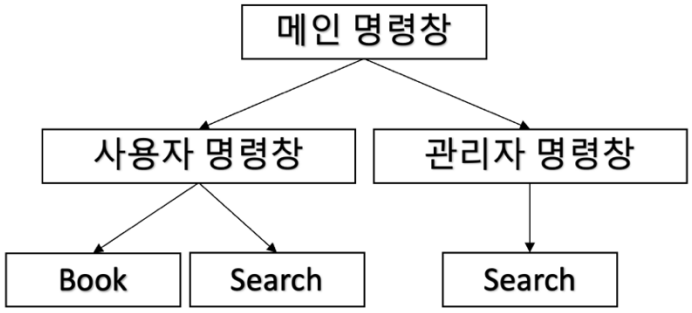
사용자 명령창과 관리자 명령창에서 명령어를 이용해 세부적인 명령창으로 이동할 수 있습니다. 사용자 명령창 내의 세부 명령창에는 "Book", "Search"가 존재하고, 관리자 명령창 내의 세부 명령창은 "Search"가 존재합니다. 세부 명령창으로의 이동과 출력 예시는 아래와 같습니다.

```
1. 사용자 명령창 > Book
{username} > Book⏎
{username}/Book >

2. 사용자 명령창 > Search
{username} > Search⏎
{username}/Search >

3. 관리자 명령창 > Search
Admin > Search⏎
Admin/Search>
```

명령창의 구성은 아래 그림과 같습니다.



8 오류 처리

본 절은 프로그램 이용 시 발생할 수 있는 공통 오류를 정의하고 해당 오류가 발생했을 때 출력할 오류 메시지와 비상 동작에 대해 정의합니다. 본 절에서 언급되지 않은 오류는 특정 명령어를 사용할 때만 발생하는 오류로, 각 명령어를 정의할 때 함께 정의됩니다. 모든 오류 메시지는 앞에 "[오류] " 문자열과 함께 출력합니다.

8.1 명령 입력 오류

8.1.1 명령어 오류

사용자가 공백을 기준으로 첫 번째로 입력한 문자열을 명령어로 해석하고, 명령어 해석 결과가 현재 사용할 수 있는 명령어에 존재하지 않는 경우 오류가 발생합니다. 발생한 오류 종류에 따라 다음과 같은 오류 메시지를 출력하고, 현재 상태에서 사용할 수 있는 명령어 목록을 출력한 후 재입력을 받습니다. 사용 가능한 명령어 목록을 출력하는 것은 9.2절에서 인자 없이 help 명령어를 사용하는 것과 동일한 기능을 합니다.

예시

■ 현재 명령창에서 사용할 수 없는 명령어를 입력했을 때

기본형식: 현재 명령창 [명령창]에서 [COMMAND]은(는) 사용할 수 없는 명령어입니다.

- 현재 명령창 : 현재 사용자가 위치한 명령창을 의미합니다.
- COMMAND : 사용자가 입력한 잘못된 명령어를 의미합니다.

```
{userName}/Book > exit 3⏎
[오류] 현재 명령창 Book에서 exit(는) 사용할 수 없는 명령어입니다.
-----------------------------------------------
  명령어    |     기능
-----------------------------------------------
help        |  도움말 출력
back        |  이전 프롬프트로 되돌아감
(생략)
{userName}/Book >
```

■ 프로그램 자체에 존재하지 않는 명령어를 입력했을 때

즉, 어느 명령창에서도 지원하지 않는 명령어를 입력할 경우입니다.

기본형식: 명령어 [COMMAND]은(는) 존재하지 않는 명령어입니다.

```
{userName}/Book > shutdown 3asdf01⏎
[오류] 명령어 shutdown은(는) 존재하지 않는 명령어입니다.
-----------------------------------------------
  명령어     |     기능
-----------------------------------------------
help         |  도움말 출력
back         |  이전 프롬프트로 되돌아감
(생략)
{userName}/Book >
```

8.1.2 인자 문법 오류

사용자가 명령어는 정상적으로 사용하였으나, 전달한 인자가 해당 명령어의 인자 문법 규칙을 만족하지 못한 경우 오류가 발생합니다. 발생한 오류 종류에 따라 다음과 같은 오류 메시지를 출력하고 재입력 받습니다.

■ 해당 명령어가 받을 수 있는 인자의 수와 다른 인자들을 전달하였을 때

기본형식: 명령어 [COMMAND]이(가) 가질 수 있는 인자의 개수와 다릅니다.

```
{userName}/Book > Book 3 2⏎
[오류] 명령어 Book이(가) 가질 수 있는 인자의 개수와 다릅니다.
{userName}/Book >
```

■ 전달한 인자의 수는 정확하나 그 인자가 가질 수 없는 문자를 가지는 경우

기본형식 : 입력 인자 [ARGUMENT]에 사용할 수 없는 문자 '[CHAR]'이(가) 존재합니다.

- ARGUMENT : 오류가 발생한 첫번째 인자를 의미합니다.
- CHAR : 인자에서 문법 오류의 원인이 되는 첫 번째 문자를 의미합니다.

```
Admin/Search > ask 010jdk1929⏎
[오류] 입력 인자 010jdk1929에 사용할 수 없는 문자 'j'이(가) 존재합니다.
Admin/Search >
```

■ 전달한 인자의 수는 정확하나 그 인자가 가질 수 있는 문자열의 길이를 벗어나는 입력인 경우

기본형식: 입력 인자가 허용가능한 길이 [LEN]을(를) 넘어섰습니다.

- LEN : 해당 인자가 가질 수 있는 길이의 한계로 정의해둔 값

```
Admin/Search > ask 1209309823740918723⏎
[오류] 입력 인자가 허용가능한 길이 20을(를) 넘어섰습니다.
Admin/Search >
```

■ 전달한 인자의 수와 형태는 정확하나 잘못된 문법을 가질 때
즉, 위에서 정의한 오류 이외의 문법 오류를 의미합니다.
기본형식: 입력 인자 [ARGUMENT](이)가 잘못된 문법을 가집니다.

- ARGUMENT : 오류가 발생한 첫 번째 인자를 의미합니다.

```
{userName}/Search > list wrong⏎
[오류] 입력 인자 wrong이(가) 잘못된 문법을 가집니다.
{userName}/Search >
```

8.1.3 인자 의미 오류

각 인자의 문법은 정확하나, 데이터 요소나 특정 명령어에서 정의한 의미규칙을 만족하지 못한 경우 오류 메시지를 출력하고 재입력 받습니다.

기본 형식: 입력 인자 [ARGUMENT]이(가) 의미 규칙을 만족하지 못합니다. > [RULE]

- ARGUMENT : 오류가 발생한 첫번째 인자를 의미합니다.
- RULE : 데이터 요소나 명령어에 대해 정의해 둔 의미 규칙을 설명하는 문자열입니다.

```
{userName}/Book > book 221002 1 3010 4500⏎
[오류] 입력 인자 3010(가) 의미 규칙을 만족하지 못합니다. > 존재하지 않는 시간입니다.
{userName}/Book >
```

8.2 파일 관련 오류

이 프로그램은 여러가지 파일들을 다루기 때문에, 파일을 관리하는데 있어서 다양한 오류들이 발생할 수 있습니다. 파일상에 오류가 존재할 시 프로그램을 사용할 수 없기 때문에 발생한 오류의 종류에 따라 다음과 같은 오류 메시지를 출력하고 프로그램을 종료합니다.

■ 프로그램 실행 시 "{%HOME}/resource/meta.txt" 파일이 존재하지 않는 경우

```
프로그램을 시작합니다.
meta.txt 이(가) 존재하는지 확인합니다.
[오류] meta.txt가 존재하지 않으므로, 프로그램을 종료합니다.
```

■ 프로그램 실행 시 "{%HOME}/resource/meta.txt" 파일 형식 검증에 실패한 경우

```
프로그램을 시작합니다.
meta.txt 이(가) 존재하는지 확인합니다. ...성공
meta.txt 이(가) 올바른 형식을 만족하는지 검증합니다.
[오류] meta.txt가 올바른 형식이 아니므로, 프로그램을 종료합니다.
```

- 존재하지 않는 파일을 열고자 하였을 때
  기본 형식: '[FILEPATH]'은(는) 존재하지 않는 파일입니다.
  - FILEPATH: 프로그램이 열고자 했던 파일의 경로를 나타냅니다. 이때 홈경로는 제외합니다.

```
{userName}/Search > list 4 231011⏎
[오류] '/book/231011.txt'은(는) 존재하지 않는 파일입니다.
프로그램을 종료합니다.
```

- 파일을 사용하고자 할 때, 파일에 저장된 내용의 형식이 6.2에서 정의한 것과 다른 경우
  기본 형식 : '[FILEPATH]'에 저장된 내용의 형식에 문제가 존재합니다.
  - FILEPATH : 오류가 발생한 파일의 경로를 나타냅니다. 이때 홈경로는 제외합니다.

```
{userName}/Search > list 4 231011⏎
[오류] '/book/231011.txt'에 저장된 내용의 형식에 문제가 존재합니다.
프로그램을 종료합니다.
```

8.3 기타

이 절에서 정의하지 않은 예기치 못한 모든 오류에 대해서는 아래와 같은 메시지를 출력하고 프로그램을 강제로 종료합니다. 이때 어떤 예외로 인해 프로그램이 종료되었는지를 알기 위해서 오류 메시지에는 발생한 표준 C++ Exception class에 대한 정보를 출력합니다.

기본형식: <C++> [EXCEPTION] 예외가 발생하여 프로그램이 종료되었습니다.

- EXCEPTION : 표준 C++ 라이브러리에서 정의한 예외 클래스의 이름을 의미합니다.

```
{userName}/Book > ⏎
[오류] <C++> bad_typeid 예외가 발생하여 프로그램이 종료되었습니다.
```

9 공통 명령

9.1 뒤로 가기

현재 명령창에서 이전 명령창으로 되돌아가기 위해 사용됩니다.

명령어: back
인자 문법 규칙: 이 명령은 인자를 가지지 않습니다.
인자 의미 규칙: 해당 명령어는 별도의 의미규칙을 가지지 않습니다.
사용 조건: 해당 명령어는 다음 조건을 만족해야 합니다.

- 메인 명령창에서는 뒤로 돌아가는 명령창이 존재하지 않기 때문에 사용할 수 없습니다.
- 회원이나 관리자로 로그인한 상태에서 메인 명령창으로 돌아가기 위해서는 로그아웃을 필요로 합니다. 즉, 사용자 명령창이나 관리자 명령창 화면에서는 back을 이용하여 메인 명령창으로 되돌아갈 수 없습니다.

아래 표는 명령창 이동시 필요한 명령어와 동작 설명입니다. "exit"와 "logout" 명령어는 아직 서술되어 있지 않았고, 각각 **10.3**, **11.3**, **12.2**에서 서술하지만, 편의(이해)를 위해 미리 서술합니다.

현재 명령창	이전 명령창	명령어 및 동작
메인 명령창	없음	"exit" 명령어로 프로그램 종료
사용자 명령창	메인 명령창	"logout" 명령어로 이동
사용자 명령창 > Book	사용자 명령창	"back" 명령어로 이동
사용자 명령창 > Search	사용자 명령창	"back" 명령어로 이동
관리자 명령창	메인 명령창	"logout" 명령어로 이동
관리자 명령창 > Search	관리자 명령창	"back" 명령어로 이동

9.2 도움말

명령어: help
인자 문법 규칙: 이 명령은 인자를 최대 1개까지 가질 수 있습니다.
즉, 인자 없이 단독으로 사용되거나, 단 1개의 인자만을 가질 수 있습니다.
인자 의미 규칙: 인자를 가지는 경우, 인자는 아래 조건을 모두 만족해야 합니다.

- 프로그램에 존재하는 명령어이어야 합니다.
- 현재 명령창에서 사용 가능한 명령어이어야 합니다.

정상 동작 1: 인자를 사용하지 않은 경우
현재 명령창에서 사용할 수 있는 명령어들의 목록을 출력합니다.

```
{userName}/Book > help⏎
-------------------------------------------------
  명령어     |     기능
-------------------------------------------------
help         |  도움말 출력
back         |  이전 프롬프트로 되돌아감
(생략)
```

정상 동작 2: 인자를 정상적으로 사용한 경우

해당 명령어를 사용하는 방법에 대한 설명을 출력합니다.

```
{userName}/Book > help list⏎
명령어
        list : 스터디룸의 예약 현황을 출력합니다.
가능한 인자의 형태
        list [인자 1]
        list [인자 2]
        list [인자 1] [인자 2]
각 인자의 규칙
        [인자 1] : 예약할 인원수를 입력합니다.
        [인자 2] : 스터디룸의 예약 현황을 확인할 날짜를 입력합니다.
(생략)
```

10 메인 명령

10.1 회원가입

해당 프로그램은 사용자의 회원가입만을 지원합니다. 관리자 계정은 프로그램 내에서 회원가입이 불가능하고, 별도의 "{%HOME}/resource/meta.txt" 파일의 1행에서 관리자 이름과 전화번호를 탭(Tab)으로 구분해서 작성 후 이용 가능합니다.

회원가입 후, 회원정보는 별도의 사용자 번호와 함께 "{%HOME}/resource/userdata.txt" 파일에 저장합니다. 또한 "{%HOME}/user/"에 해당 사용자 데이터 파일이 생성되며, "{%HOME}/resource/meta.txt"의 2행의 값도 1 증가합니다.

명령어: signup

인자 문법 규칙: 이 명령어는 반드시 2개의 인자를 가져야 합니다.

- 인자1은 위에서 정의한 이름의 문법 형식을 따릅니다.
- 인자2는 위에서 정의한 전화번호의 문법 형식을 따릅니다.

인자 의미 규칙:

- 인자1은 위에서 정의한 이름의 의미 규칙을 따릅니다.
- 인자2는 위에서 정의한 전화번호의 의미 규칙을 따릅니다.
- 사용자 전화번호는 다른 사용자 또는 관리자 전화번호와 중복될 수 없습니다.

정상 동작 예시

```
Main > signup username 010-1234⏎
회원가입이 완료되었습니다.
사용자 이름: username
사용자 전화번호: 0101234
```

비정상 동작: 이미 회원 가입된 전화번호인 경우

```
Main > signup username 7777777⏎
[오류] 26WrongRuleArgumentException 예외가 발생하였습니다.
입력 인자 7777777(이)가 의미규칙을 만족하지 못합니다.
규칙 : 이미 회원가입이 되어있는 전화번호입니다.
```

10.2 로그인

이름과 전화번호를 입력하여 로그인합니다. "{%HOME}/resource/userdata.txt"에 있는 사용자 정보와 인자로 입력한 이름과 전화번호가 동일한 사용자가 존재할 경우, 해당 사용자로 로그인하고 사용자 명령창으로 이동합니다. 또는 "{%HOME}/resource/meta.txt"의 1행에 저장되어 있는 관리자 이름과 전화번호와 모두 동일하면 관리자 명령창으로 이동합니다.

명령어: signin

인자 문법 규칙: 이 명령어는 반드시 2개의 인자를 가져야 합니다.

- 인자1은 위에서 정의한 이름의 문법 형식을 따릅니다.
- 인자2는 위에서 정의한 전화번호의 문법 형식을 따릅니다.

인자 의미 규칙:

- 인자1은 위에서 정의한 이름의 의미 규칙을 따릅니다.
- 인자2는 위에서 정의한 전화번호의 의미 규칙을 따릅니다.

정상 동작 1: 입력한 계정이 사용자 계정일 경우

사용자 명령창으로 이동합니다.

```
Main > signin user 02-1234-56⏎
사용자로 로그인 되었습니다.
{userName} >
```

정상 동작 2: 입력한 계정이 관리자 계정일 경우

관리자 명령창으로 이동합니다.

```
Main > signin admin 0101234-13⏎
관리자로 로그인 되었습니다.
Admin >
```

비정상 동작: 존재하지 않는 이용자일 경우

```
Main > signin wrong 0101234-13⏎
[오류] 존재하지 않는 회원입니다.
Main >
```

10.3 프로그램 종료

프로그램을 종료하기 위해 사용됩니다.

명령어: exit
인자 문법 규칙: 이 명령은 인자를 가지지 않습니다.
인자 의미 규칙: 해당 명령어는 별도의 의미규칙을 가지지 않습니다.

정상 동작: 메인 명령창에 올바르게 입력한 경우, 프로그램이 종료됩니다.

```
Main > exit⏎
프로그램을 종료합니다.
```

11 사용자 명령

이 프로그램에서 이용자는 사용자와 관리자로 나누어집니다. 본 절은 사용자가 사용할 수 있는 명령들을 서술했습니다.

11.1 예약하기

예약하려는 시간의 스터디룸의 상태가 예약가능상태이면 사용자 예약을 할 수 있는 기능입니다. 스터디룸의 예약가능상태의 조건입니다.

명령어: book

인자 문법 규칙: 이 명령은 인자가 반드시 45개 이어야 합니다. 인자1은 예약할 날짜, 인자2는 예약할 스터디룸의 번호, 인자3은 이용 시작 시간, 인자4는 이용 종료 시간, 인자5는 예약 인원을 의미합니다.

- 인자1은 위에서 정의한 날짜의 문법에 맞아야 합니다.
- 인자2, 인자5는 위에서 정의한 숫자의 문법에 맞아야 합니다.
- 인자3, 인자4는 위에서 정의한 시간의 문법에 맞아야 합니다.

인자 의미 규칙:

- 인자1은 위에서 정의한 날짜의 의미 규칙과 동일합니다.
- 인자2는 위에서 정의한 스터디룸 방 번호의 의미규칙과 동일합니다.
- 인자3, 인자4는 위에서 정의한 '시간'의 의미 규칙과 동일합니다. 또한 스터디룸 운영시간인 09:00(오픈시간) ~ 20:00(마감시간) 사이의 시간이어야 합니다.

해당 명령어에서는 아래와 같은 추가적인 의미 규칙이 적용됩니다.

- 예약 날짜는 현재의 날짜보다 미래이여야 하며, 현재 날짜 기준 3개월 이내이어야 합니다.
- 입력한 날짜, 스터디룸 번호, 예약시작시간, 예약종료시간에 맞는 스터디룸의 상태가 예약가능 상태여야 합니다.
- 인자4는 인자3보다 30분 이상 커야 합니다. 또한 독서실 운영 시간을 벗어날 수 없습니다.
- 인자5는 예약할 스터디룸의 최대 예약 가능 인원 수보다 작거나 같아야 합니다.

정상 동작: 정상적으로 예약이 된 경우

```
{userName}/Book > book 221002 1 0900 1100 3⏎
2022-10-02 1번 스터디룸 09:00 ~ 11:00 로 정상 예약되었습니다.
{userName}/Book >
```

비정상 동작1: 선택한 스터디룸 방의 시간에 예약 불가능한 상태인 경우

```
{userName}/Book > book 221002 1 0900 1100 6⏎
[오류] 2022-10-02 1번 스터디룸 09:00 ~ 11:00는 예약 불가능한 상태입니다.
{userName}/Book >
```

비정상 동작2: 선택한 스터디룸 방의 최대 예약 가능 인원 수보다 큰 수를 입력했을 경우

```
{userName}/Book > book 221002 1 0900 1100 50⏎
[오류] 2022-10-02 1번 스터디룸의 최대 예약 가능 인원수는 3명입니다.
{userName}/Book >
```

11.2 예약 확인

11.2.1 스터디룸 예약 상태 확인

입력한 날짜와 예약 인원에 맞는 스터디룸의 시간대별 예약 여부를 출력합니다. 각 시간대에 예약이 되어 있지 않거나 이미 예약이 존재하지만 최적화가 가능하다면 "가능"으로 출력되고, 이미 예약이 존재하고 최적화가 불가능하다면 "X"가 출력됩니다.

명령어: list

인자 문법 규칙: 이 명령은 인자가 ~~0~~1개 또는 ~~1~~2개 이어야 합니다.

- 인자1은 스터디룸에 예약할 인원을 의미합니다.
- (만일 인자가 ~~존재할 시~~2개일 때) 인자~~1~~2는 예약 리스트를 볼 날짜를 의미합니다. 이 때 인자~~1~~2는 위에서 정의한 날짜의 문법에 맞아야 합니다.

인자 의미 규칙: 인자가 존재한다면 아래 규칙을 만족해야 합니다.

- 인자1은 위에서 정의한 예약 인원의 의미규칙과 동일합니다.
- 인자~~1~~2는 위에서 정의한 날짜의 의미 규칙과 동일합니다.
- 추가로, 인자2가 현재 날짜 기준 3개월보다 앞서면 오류이며, 과거 날짜를 입력해도 오류입니다.

정상 동작 1: 모든 규칙에 올바른 명령을 입력할 경우

인자1의 예약 인원을 고려한, 인자~~1~~2에 해당하는 날짜의 스터디룸 예약 가능 여부가 리스트로 출력됩니다.

```
{userName}/Book > list 3 221011⏎
2022-10-11 스터디룸 상태입니다.
스터디룸        09:00   09:30   10:00   10:30   ...(생략)...   19:30
1 (2인실)       X       X       X       X       ...(생략)...   X
2 (3인실)       가능    X       가능    X       ...(생략)...   가능
3 (3인실)       가능    가능    X       X       ...(생략)...   가능
4 (4인실)       가능    가능    X       X       ...(생략)...   가능
5 (6인실)       가능    가능    X       X       ...(생략)...   가능
6 (6인실)       가능    가능    X       X       ...(생략)...   가능
(생략)
{userName}/Book >
```

정상 동작 2: ~~인자를 입력하지 않은 경우~~ 인자가 1개일 경우 ~~인자가 0개일 경우~~ 인자1의 예약 인원을 고려한, 당일 날짜의 스터디룸 예약 가능 여부가 리스트로 출력됩니다.

```
{userName}/Book > list 3⏎
2022-10-11 스터디룸 상태입니다.
스터디룸        09:00   09:30   10:00   10:30   ...(생략)...   19:30
1 (2인실)       X       X       X       X       ...(생략)...   X
2 (3인실)       가능    X       가능    X       ...(생략)...   가능
3 (3인실)       가능    가능    X       X       ...(생략)...   가능
4 (4인실)       가능    가능    X       X       ...(생략)...   가능
5 (6인실)       가능    가능    X       X       ...(생략)...   가능
6 (6인실)       가능    가능    X       X       ...(생략)...   가능
(생략)
{userName}/Book >
```

비정상 동작 1: 조회 날짜가 오늘 날짜 기준 3개월을 초과했을 경우

```
{userName}/Book > list 4 231011⏎
[오류] 현재 날짜 기준 3개월까지만 확인 가능합니다.
{userName}/Book >
```

비정상 동작 2: 조회 날짜가 현재보다 과거인 경우

```
{userName}/Book > list 4 201011⏎
[오류] 과거 날짜는 조회할 수 없습니다.
{userName}/Book >
```

11.2.2 사용자 예약 내역 확인

사용자의 예약된 스터디룸을 나열 우선 순위에 맞춰 나열합니다.
나열 우선 순위는 아래와 같습니다.

1. 날짜가 빠른 순
2. 예약시작시간이 빠른 순
3. 예약종료시간이 느린 순
4. 예약된 방 번호가 빠른

명령어: check
인자 문법 규칙: 이 명령은 인자가 반드시 0개 이어야 합니다.
인자 의미 규칙: 이 명령은 별도의 의미 규칙이 없습니다.

정상 동작 1: 예약 내역이 1개 이상일 경우

```
{userName}/Search > check⏎
예약번호    예약날짜    인원    스터디룸 번호    시작시간    종료시간
  1.     2022-10-11   4명     1번 스터디룸     09:00     11:00
  2.     2022.10-12   3명     2번 스터디룸     09:00     10:30
         2022-10-12   3명     4번 스터디룸     10:30     11:00
  3.     2022-10-12   2명     5번 스터디룸     10:00     11:00
  4.     2022-10-13   6명     3번 스터디룸     09:00     11:00
  5.     2022-10-14   3명     1번 스터디룸     09:00     11:00
{userName}/Search >
```

정상 동작 2: 예약 내역이 0개일 경우

```
{userName}/Search > check⏎
예약번호        예약날짜        인원    스터디룸 번호        시작시간        종료시간
{userName}/Search >
```

11.3 로그아웃

사용자 명령창에서 메인 명령창으로 이동하기 위한 명령어입니다.

명령어: logout
인자 문법 규칙: 이 명령은 인자가 반드시 0개 이어야 합니다.
인자 의미 규칙: 이 명령은 별도의 의미 규칙이 없습니다.

정상 동작: 9.1절에서 설명한 것처럼, 메인 명령창으로 이동합니다.

```
{userName} > logout⏎
로그아웃 되었습니다. 메인 명령창으로 이동합니다.
Main >
```

12 관리자 명령

12.1 회원 정보 검색

관리자가 회원 정보를 검색할 수 있는 명령어입니다. 관리자는 전화번호 또는 회원의 이름으로 해당하는 회원을 조회할 수 있습니다. 관리자는 조회한 회원의 전화번호, 이름, 예약 현황, 과거 예약 내역을 확인할 수 있습니다.

명령어: ask

인자 문법 규칙: 이 명령은 인자가 최대 2개까지 가질 수 있습니다.

- 인자가 1개인 경우에 다음 3가지가 인자로 들어올 수 있습니다.
  - 이름만 입력되는 경우
  - 전화번호만 입력되는 경우
  - 이름과 전화번호가 공백 없이 연속적으로 입력되는 경우
- 인자가 2개인 경우에 두 인자는 이름과 전화번호를 의미하며, 인자 순서는 상관없습니다.
- 이름과 전화번호는 사전에 정의한 문법 규칙에 맞아야 합니다.

인자 의미 규칙:

- 입력한 전화번호는 현재 존재하는 사용자의 전화번호이어야 합니다.
- 전화번호와 이름을 인자로 입력할 경우, 전화번호에 해당하는 사용자 이름과 입력한 이름이 일치해야 합니다.

정상 동작 1: 인자가 없는 경우

모든 회원 리스트를 출력합니다.

```
Admin/Search > ask⏎
이름     전화번호
hong    01012341234
kang    01011112222
kang    01022223456
Admin/Search >
```

정상 동작 2: 인자가 1개로 이름을 입력하는 경우

해당 이름과 일치하는 사용자의 목록을 출력합니다.

```
Admin/Search > ask kang⏎
이름    전화번호
kang    01011112222
kang    01022223456
Admin/Search >
```

정상 동작 3: 인자가 1개로 전화번호를 입력하고, 검색 결과가 존재할 경우

전화번호는 사용자별 유일(unique)하므로, 해당 사용자의 이름, 전화번호와 예약 현황을 출력합니다.

```
Admin/Search > ask 010-1111-2222⏎
이름    전화번호
kang    010-1111-2222

예약 현황
2022-11-20 | 13:00 ~ 15:00 | 스터디 룸 번호 : 1, 2, 5 | 예약 번호 : 9
2022-11-24 | 14:00 ~ 16:00 | 스터디 룸 번호 : 4 | 예약 번호 : 12
2022-12-10 | 12:00 ~ 15:00 | 스터디 룸 번호 : 5 | 예약 번호 : 15

과거 이용 내역
2022-08-20 | 13:00 ~ 15:00 | 스터디 룸 번호 : 1 | 예약 번호 : 1
2022-08-24 | 14:00 ~ 16:00 | 스터디 룸 번호 : 4 | 예약 번호 : 2
2022-08-10 | 12:00 ~ 15:00 | 스터디 룸 번호 : 5 | 예약 번호 : 4
Admin/Search >
```

정상 동작 4: 인자가 1개로 이름과 전화번호를 연속적으로 입력하는 경우

연속된 숫자와 숫자가 아닌 연속된 문자를 기준으로 2개의 문자열로 나눠서

숫자가 포함되지 않는 문자열은 이름으로, 그 외의 문자열은 전화번호로 처리합니다.

```
Admin/Search > ask kang010-1111-2222⏎
이름    전화번호
kang    01011112222

예약 현황
2022-11-20 | 13:00 ~ 15:00 | 스터디 룸 번호 : 1, 2, 5 | 예약 번호 : 9
2022-11-24 | 14:00 ~ 16:00 | 스터디 룸 번호 : 4 | 예약 번호 : 12
2022-12-10 | 12:00 ~ 15:00 | 스터디 룸 번호 : 5 | 예약 번호 : 15

과거 이용 내역
2022-08-20 | 13:00 ~ 15:00 | 스터디 룸 번호 : 1 | 예약 번호 : 1
2022-08-24 | 14:00 ~ 16:00 | 스터디 룸 번호 : 4 | 예약 번호 : 2
2022-08-10 | 12:00 ~ 15:00 | 스터디 룸 번호 : 5 | 예약 번호 : 4
```

정상 동작 5: 인자가 2개로 이름과 전화번호를 입력하는 경우

```
Admin/Search > ask kang 010-1111-2222⏎
이름    전화번호
kang    01011112222

예약 현황
2022-11-20 | 13:00 ~ 15:00 | 스터디 룸 번호 : 1, 2, 5 | 예약 번호 : 9
2022-11-24 | 14:00 ~ 16:00 | 스터디 룸 번호 : 4 | 예약 번호 : 12
2022-12-10 | 12:00 ~ 15:00 | 스터디 룸 번호 : 5 | 예약 번호 : 15

과거 이용 내역
2022-08-20 | 13:00 ~ 15:00 | 스터디 룸 번호 : 1 | 예약 번호 : 1
2022-08-24 | 14:00 ~ 16:00 | 스터디 룸 번호 : 4 | 예약 번호 : 2
2022-08-10 | 12:00 ~ 15:00 | 스터디 룸 번호 : 5 | 예약 번호 : 4
Admin/Search >
```

비정상 동작 1: 이름과 전화번호를 혼합해서 입력하는 경우

[정상 동작 4]와 같이 숫자와 영문자를 문자열 2개로 나눌 수 없기 때문에 오류입니다.

```
Admin/Search > ask k010-1ang111-2222⏎
[오류] 입력 형식이 올바르지 않습니다.
Admin/Search >
```

비정상 동작 2: 인자 중 전화번호가 있지만, 해당 전화번호와 일치하는 사용자가 없는 경우

```
Admin/Search > ask 010-12436-1⏎
[오류] 해당 전화번호와 일치하는 사용자를 찾을 수 없습니다.
Admin/Search >
```

12.2 로그아웃

관리자 명령창에서 메인 명령창으로 이동하기 위한 명령어입니다.

명령어: logout
인자 문법 규칙: 이 명령은 인자가 반드시 0개 이어야 합니다.
인자 의미 규칙: 이 명령은 별도의 의미 규칙이 없습니다.

정상 동작: 9.1절에서 설명한 것처럼, 메인 명령창으로 이동합니다.

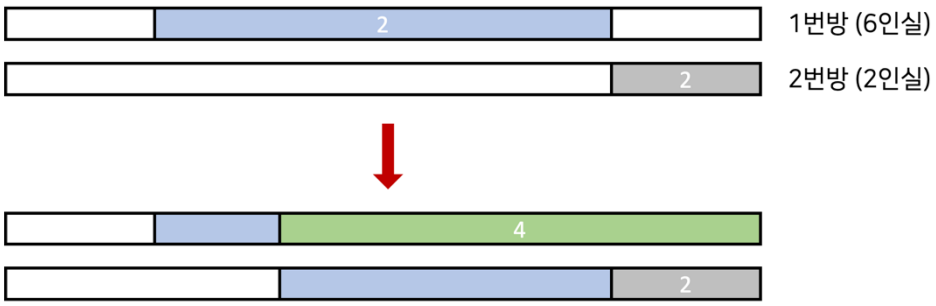
```
Admin > logout⏎
로그아웃 되었습니다. 메인 명령창으로 이동합니다.
Main >
```

13 예약 최적화

13.1 예약 최적화 방법

본 프로그램에서는 최대한 많은 인원이 스터디룸을 이용할 수 있도록 특수한 예약 최적화 방법을 사용하고 있습니다. 어떤 스터디룸에 대하여 기존의 예약 인원보다 더 많은 사람이 해당 시간대에 스터디룸을 사용하고자 한다면, 기존 예약의 스터디룸을 자동 변경하여 새로운 예약을 할 수 있도록 합니다. 이러한 최적화가 가능하다면, 그 시간대(30분 단위)에 스터디룸이 예약되어 있더라도 "사용 가능" 상태로 표현됩니다.

예를 들어 제한 인원이 6명인 1번 스터디룸에 현재 2명이 예약 중일 때, 4명이 동일한 시간대에 1번 스터디룸을 사용하고자 한다면, 중복되는 시간대의 기존 예약을 2인실로 변경하여 4명이 스터디룸을 사용할 수 있도록 합니다.



기본적으로 하나의 예약은 예약 시 선택한 하나의 스터디룸만 사용합니다. 스터디룸이 변경되거나, 동일한 예약(예약 번호가 같은 예약) 내에서 2개 이상의 스터디룸을 사용하게 된다면, 해당 예약에 대하여 자동 변경 알고리즘이 수행된 경우입니다.

13.2 스터디룸 자동 변경 알고리즘

제한 인원이 r명인 스터디룸에 예약 인원이 n명인 새로운 예약 B를 하기 위해, 예약 인원이 m명인 기존 예약 A의 스터디룸을 변경하기 위한 탐색 과정은 다음과 같습니다. (단, $r \geq n > m > 0$, 예약 A가 옮겨질 스터디룸의 제한 인원은 r'명)

알고리즘의 규칙

- **조건**
  - 옮겨질 스터디룸의 제한 인원이 A의 예약 인원보다 같거나 커야 합니다. ($r' \geq m$)

- 이미 예약 되어있는 경우,
  - 기존 예약 C의 예약 인원 k 명이 예약 A의 예약 인원보다 작아야 합니다. ($m > k$)
  - 그 스터디룸에 대하여 다시 자동 변경 알고리즘을 적용하였을 때 최적화가 가능해야 합니다.

■ **우선순위**: 선택 가능한 스터디룸이 여러 개 있는 경우, 다음과 같은 순서로 변경할 방을 선택합니다.

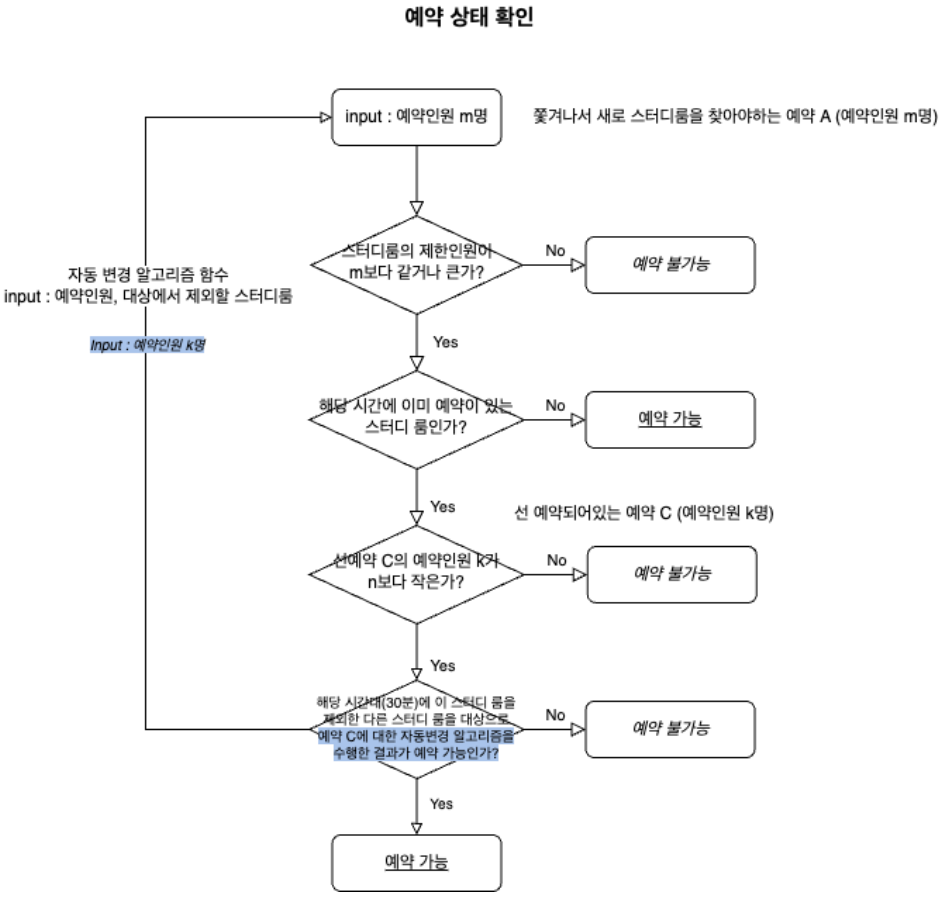
1. 해당 시간에 대한 예약이 존재하지 않는 방
2. 스터디룸의 제한 인원과 예약 인원의 차이가 적은 방 ($r' - m$)
3. 스터디룸 번호가 작은 방

탐색 과정

변경되어야 하는 시간대에 대하여 30분 단위로 모든 스터디룸을 탐색하여 새로운 스터디룸을 할당합니다.

1. 스터디룸의 제한 인원이 A의 예약 인원 미만이면 후보에서 제외합니다.
2. 해당 시간대의 예약 유무에 따라서
   a. 기존 예약이 없는 스터디룸이 존재한다면, 우선순위가 가장 빠른 스터디룸을 선택하여 그 스터디룸으로 예약을 변경합니다. <종료>
   b. 기존 예약이 없는 스터디룸이 존재하지 않는다면, 예약이 있는 스터디룸에 대하여 다음 과정을 수행합니다.
3. 기존 예약 C의 예약 인원 k 명이 A의 예약 인원 m명 이상인 스터디룸은 후보에서 제외합니다.
4. 각 스터디룸에 대하여 다시 예약 C의 스터디룸 자동 변경 알고리즘을 수행하였을 때, 최적화가 가능하지 않은 스터디룸은 후보에서 제외합니다.
5. 후보 스터디룸이 존재한다면, 우선순위가 가장 빠른 스터디룸을 선택하여 그 스터디룸으로 예약을 변경합니다. <종료>

순서도


예약 상태 확인
input : 예약인원 m명
쫓겨나서 새로 스터디룸을 찾아야하는 예약 A (예약인원 m명)
스터디룸의 제한인원이 m보다 같거나 큰가?
No
예약 불가능
Yes
자동 변경 알고리즘 함수
input : 예약인원, 대상에서 제외할 스터디룸
Input : 예약인원 k명
해당 시간에 이미 예약이 있는 스터디 룸인가?
No
예약 가능
Yes
선 예약되어있는 예약 C (예약인원 k명)
선예약 C의 예약인원 k가 n보다 작은가?
No
예약 불가능
Yes
해당 시간대(30분)에 이 스터디 룸을 제외한 다른 스터디 룸을 대상으로 예약 C에 대한 자동변경 알고리즘을 수행한 결과가 예약 가능인가?
No
예약 불가능
Yes
예약 가능