

# 데이터 구조 설계

## 2차 프로젝트

학번: 2018202084

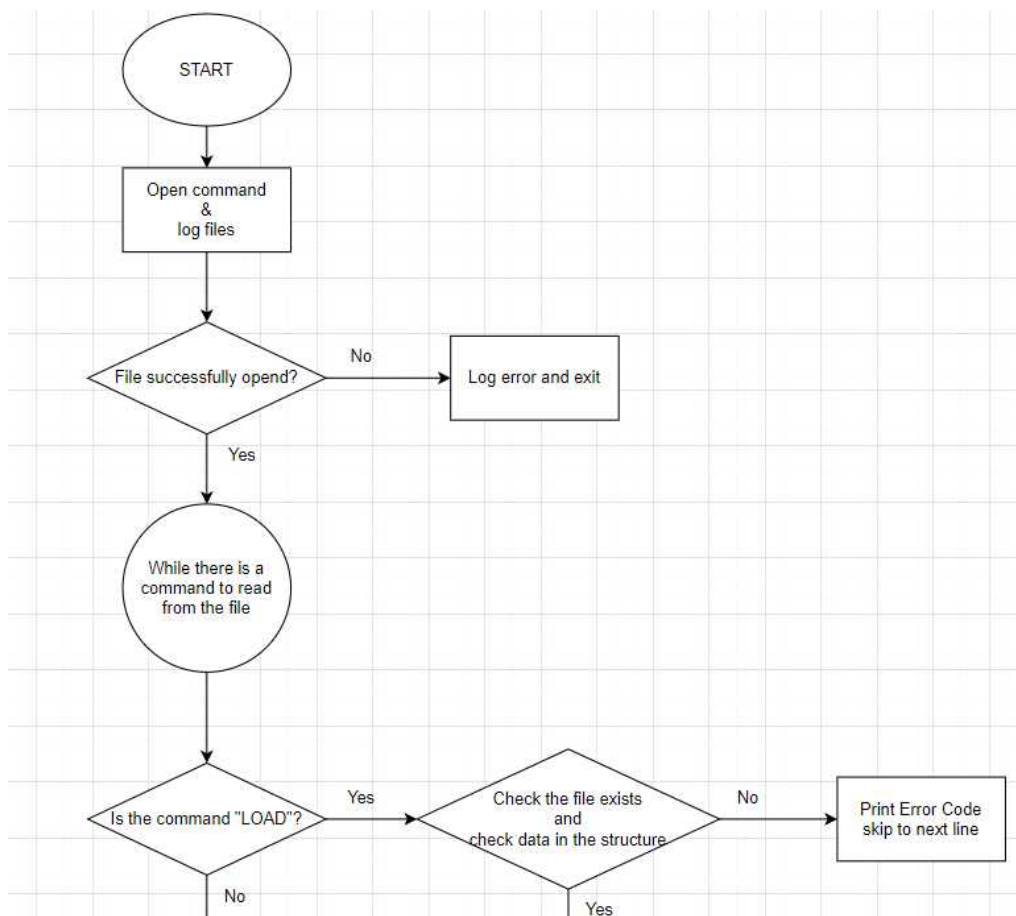
이름: 김민재

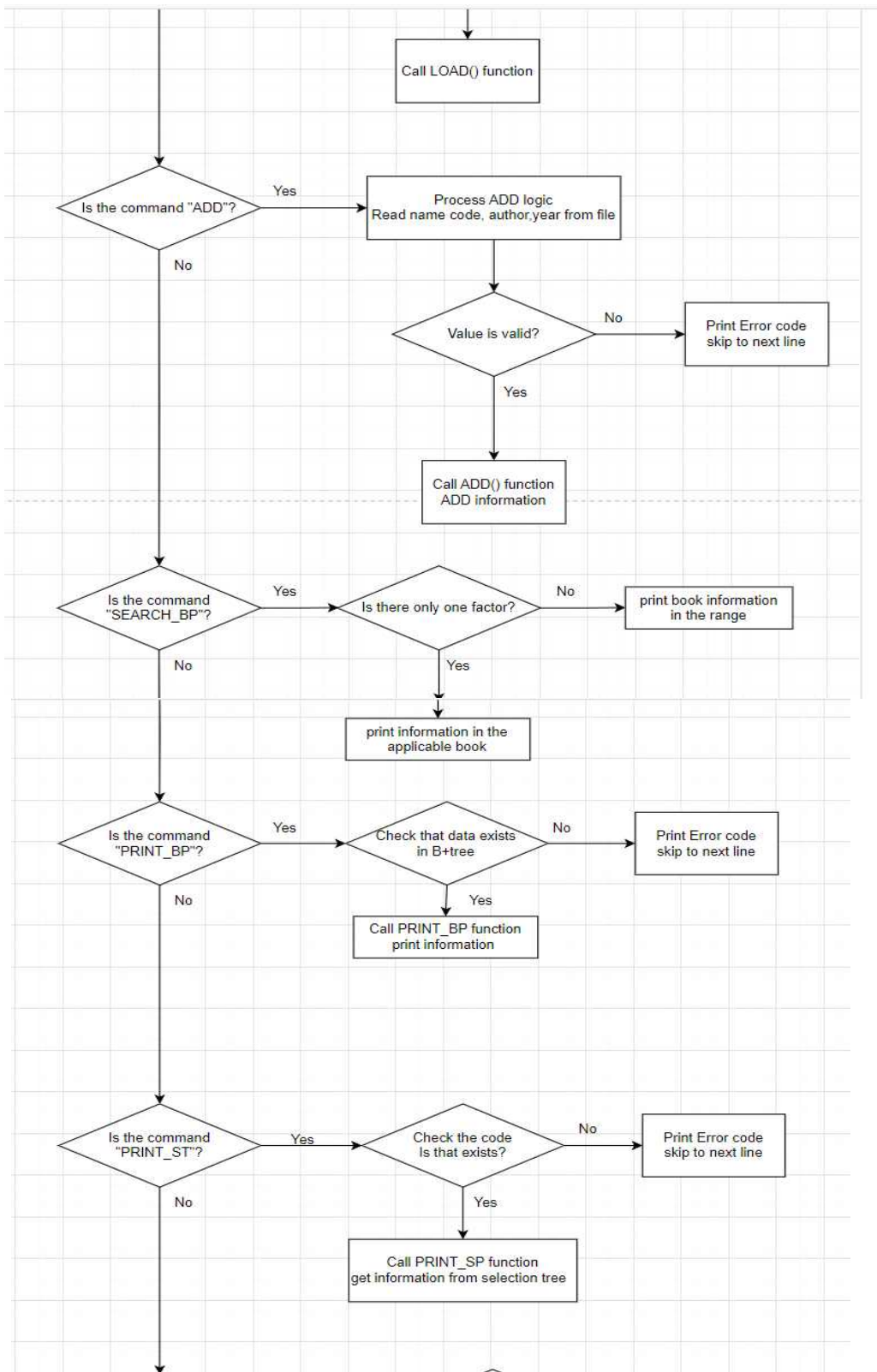
## - Introduction

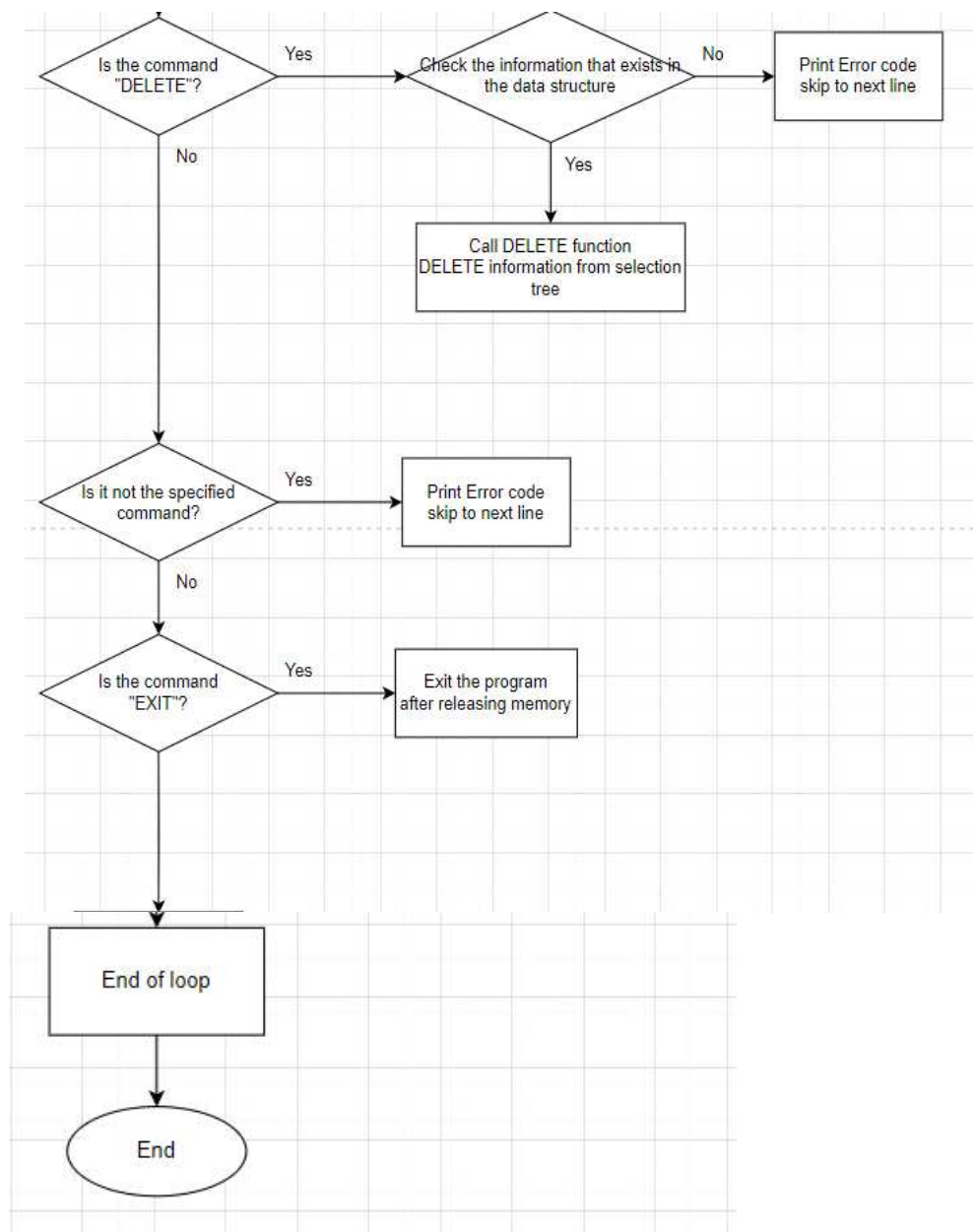
B+ tree와 selection tree 그리고 Heap을 사용하여 도서 대출 관리 프로그램을 구현해본다. 이번 프로그램은 도서의 이름과 분류 코드, 책의 발행 연도, 저자와 사용자의 대출 권수를 관리한다. 위와 같은 정보를 활용하여 사용자에게 대출하고 있는 도서와 아닌 도서에 대한 정보 제공이 가능하다. 대출하고 있는 도서의 경우는 B+ tree를 이용하여 관리하고, 대출 불가의 도서인 경우는 selection tree를 이용한다. 도서와 관련되어 있는 data들은 loan\_book.txt에 위에 해당 하는 모든 정보를 저장한다. 이후 LOAD 명령어를 이용하여 연관 정보를 class에 저장할 수 있도록 한다. 정보는 tab (\t)로 구분되어 있으며, 도서명의 중복은 없다. 이는 도서 관련 정보 데이터로 text 파일로 저장된다. 분류코드에 따른 대출 권수 또한 나뉘어 진다. 300~400 번대의 분류 코드인 경우는 3권 이하의 대출이고, 불가 도서는 대출 권수가 4인 도서를 의미하는 것이다. B+ tree의 경우는 loan\_book.txt에 저장되어 있던 data를 읽어 대출 도서를 저장하게 된다. ADD 명령어에 의하여 추가되어지는 data를 읽어 B+ tree에 저장하게 되고, tree에 없는 경우는 새 node를 추가하고, 있는 경우는 대출 권수만 증가한다. B+ tree는 LoanBookData 클래스로 정의되어 있고, 도서명, 분류코드, 저자, 연도, 대출 권수는 member 변수로 저장되어 있다. data node는 Map 컨테이너 형식으로 설정되어 있고, index와 data 노드는 B+ node 클래스를 상속 받게 된다. Selection tree는 도서명으로 하여 최소 승자 트리를 형성한다. ADD 명령어를 입력 받아 data의 대출 권수를 갱신하여 대출이 어려운 도서는 분류 코드에 따라서 Min heap으로 구성되어진 Selection Tree에 저장한다. LoanBookHeap의 경우는 책의 제목을 기준으로 정렬하게 한다. ADD 명령어를 입력 받아 B+ tree의 데이터 중에 대출이 어려운 도서는 heap으로 구현하며, 기존 heap에 없으면 새로 구축하도록 한다. ADD 명령어를 이용하여 도서 대출이 불가하면 heap을 다시 정렬하도록 한다. Heap을 다시 정렬하는 경우 parent node의 도서명이 child process 보다 작거나 같은 경우는 정렬을 완료한다. LOAD 명령어는 text 파일의 data를 읽는 명령어로 text 파일에 데이터가 있으면 B+ tree에 저장한다. text 파일이 없거나 이미 data가 들어가 있는 경우 에러코드를 출력하게 된다. ADD 명령어는 B+ tree에 도서를 직접 추가한다. 도서명, 분류 코드, 저자, 발행 년도를 가져야 하며, 1개라도 존재하지 않는 경우 에러 코드를 출력한다. SEARCH\_BP의 명령어의 경우 B+ tree에 있는 data를 검색한다. 1개 혹은 2개의 인자를 입력 받으며 1개인 경우는 도서명의 결과를 보여준다. 2개인 경우는 도서명의 범위를 나타내며 해당 범위 내에 속하는 도서를 출력

하게 한다. 인자가 없거나, B+ tree에 data가 없으면, 에러코드를 출력한다. PRINT\_BP 명령어는 B+ tree에 있는 데이터를 도서 이름을 기준으로 정렬하여 출력하도록 하고, 데이터가 없으면 에러코드를 출력하게 한다. PRINT\_ST 명령어는 selection tree로부터 받은 분류 코드에 해당하는 데이터를 출력한다. 1개의 인자를 입력 받으며 이는 분류 코드이다. 도서 이름을 기준으로 출력하고, 잘못된 분류 코드를 입력하거나 데이터가 없으면 에러 코드를 출력하도록 한다. DELETE 명령어는 selection tree에서 제거 하는 명령어로 HEAP의 대소 관계가 변경되면 HEAP을 다시 정렬 하도록 한다. 마지막으로 EXIT 명령어는 메모리를 해제하고, 프로그램을 종료시킨다.

## - Flowchart

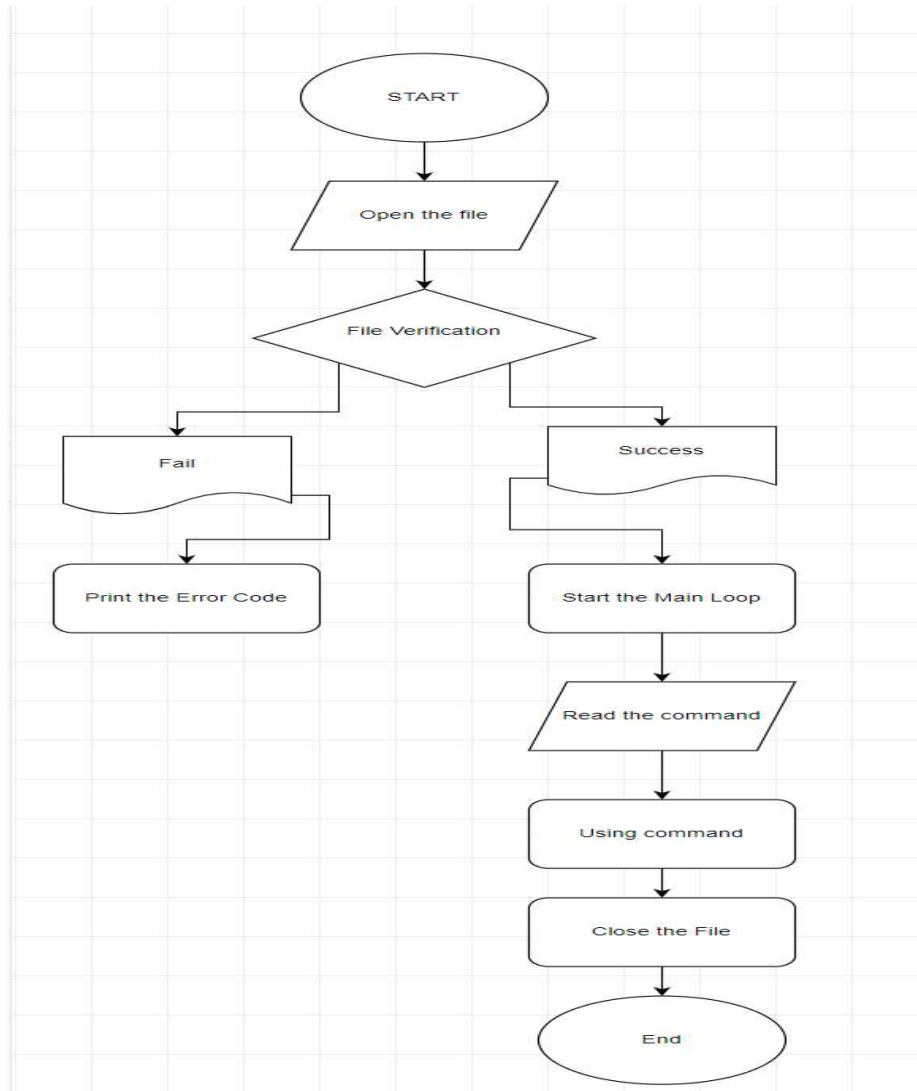






## - Algorithm

전반적인 알고리즘:



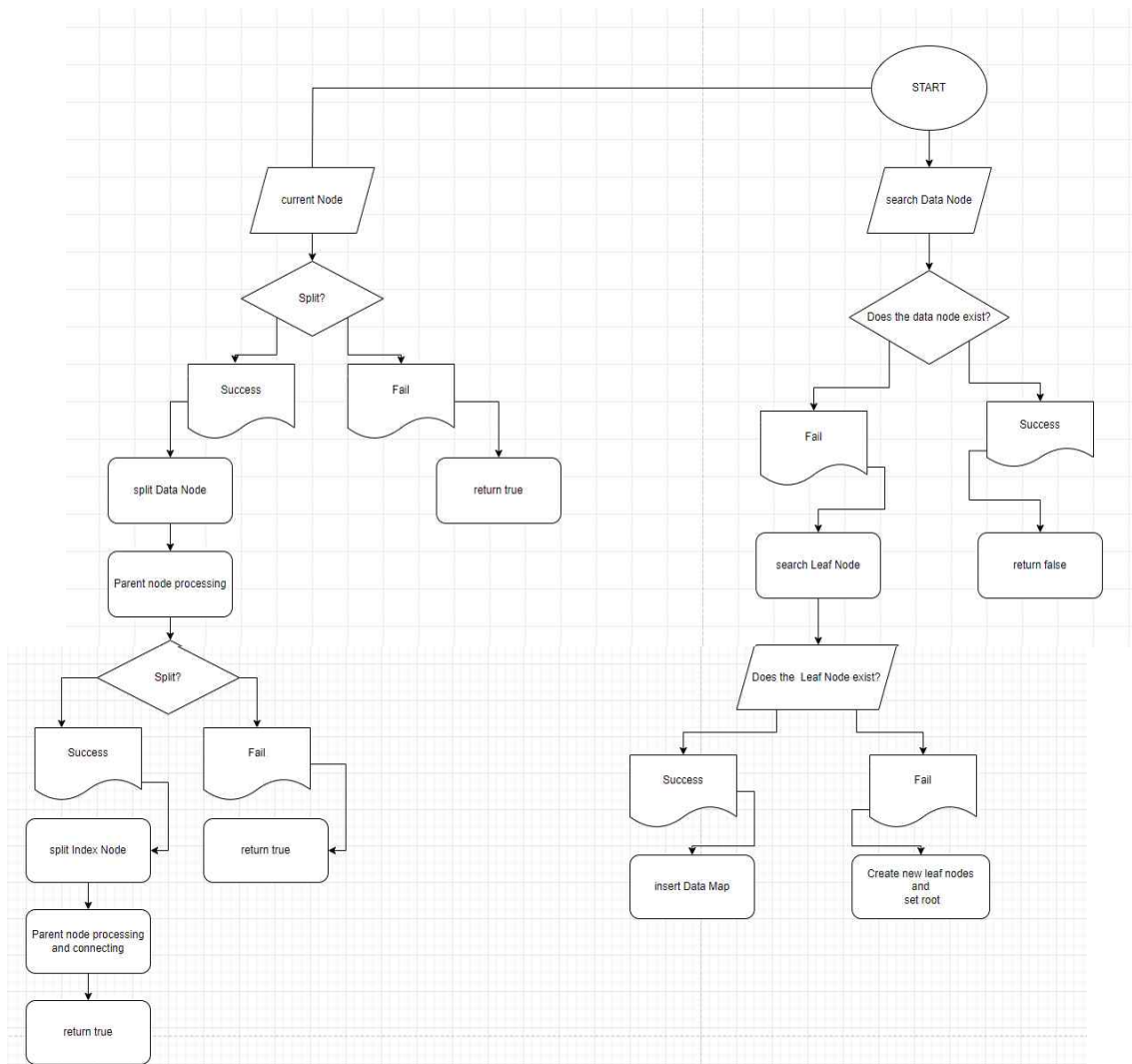
BpTree 클래스는 B+ 트리 구조를 사용하여 데이터를 관리한다. B+ 트리는 키-값 쌍을 저장하는데 사용되는 트리 구조로, 데이터베이스 및 파일 시스템에서 인덱싱에 자주 사용된다. B+ 트리의 주요 특징은 모든 값이 리프 노드에 저장되고, 내부 노드는 키를 인덱싱하는 데만 사용된다.

### B+ Tree Structure

```

[IndexNode] [IndexNode]
 /   \       /   \
[Data] [Data] [Data] [Data]

```



SelectionTree 클래스의 Insert와 printBookData 메서드는 BST의 동작 원리를 따르고 있다. BST의 각 노드는 최대 두 개의 자식 노드를 가질 수 있으며, 왼쪽 자식 노드는 부모 노드보다 작은 값을, 오른쪽 자식 노드는 부모 노드보다 큰 값을 가진다.

Insert(LoanBookData newData\*)

새 노드를 생성하여 주어진 데이터를 저장한다. 트리가 비어 있으면 새 노드를 루트 노드로 설정하고, 비어 있지 않으면 새 노드를 삽입할 위치를 찾아서 삽입한다. 삽입할 위치는 새 데이터의 코드와 기존 노드의 코드를 비교하며 찾는다. 삽입 과정은 왼쪽 또는 오른쪽 서브트리로 이동하면서 적절한 위치를 찾아나간다.

Insert 5:

```
      7
     /\
    3  8
   /\
  2  6
```

printBookData(int bookCode)

주어진 도서 코드(bookCode)를 가진 노드를 찾는다. 루트 노드부터 시작하여 도서 코드를 기준으로 왼쪽 또는 오른쪽으로 이동하며 노드를 찾는다. 도서 코드와 일치하는 노드를 찾으면 해당 도서의 데이터를 출력하고, 일치하는 노드가 없으면 함수는 false를 반환한다.

printBookData 6:

```
      7
     /\
    3  8
   /\
  2  6
```



LoanBookHeap 클래스의 코드는 min-heap 구조를 구현하고 있다. 최소 힙은 부모 노드의 키가 자식 노드의 키보다 작거나 같은 완전 이진 트리이다.

heapifyUp(LoanBookHeapNode pN\*)

새로운 노드를 삽입한 후 힙의 속성을 유지하기 위해 위로 이동시킨다. 새 노드를 부모 노드와 비교하여, 부모 노드보다 작으면 부모 노드와 위치를 바꾼 후 루트 노드에 도달하거나, 부모 노드가 더 작을 때까지 반복한다.

heapifyDown(LoanBookHeapNode pN\*)

루트 노드를 제거한 후 힙의 속성을 유지하기 위해 아래로 이동시킨다. 루트 노드를 자식 노드들과 비교하여, 더 작은 자식 노드와 위치를 바꾼다.

Insert(LoanBookData data\*)

새로운 데이터를 heap에 삽입, 새 노드를 힙의 마지막 위치에 추가한 후, heapifyUp을 호출하여 heap 속성을 유지

heapifyUp:

```
      4
    /  \
   9    [NewNode]
  /  \
 11  13
```

(NewNode를 부모와 비교하여 교환)

heapifyDown:

```
    [RemoveRoot]
   /  \
  9    5
 /  \
11  13
```

(루트를 제거하고 자식 노드와 비교하여 교환)

Insert:

- 새 노드를 완전 이진 트리의 마지막 위치에 삽입
- heapifyUp을 호출

findInsertPosition:

- BFS를 사용하여 삽입 위치를 찾음

## - Result Screen

```

dsds@ubuntu:~/Downloads/77$ make
g++ -std=c++11 -g -o run SelectionTree.cpp LoanBookHeap.cpp BpTree.cpp Manager.cpp main.cpp BpTree.h Manager.h BpTree
Node.h SelectionTree.h BpTreeIndexNode.h LoanBookHeap.h SelectionTreeNode.h BpTreeDataNode.h LoanBookHeapNode.h LoanB
ookData.h
Manager.h:1:9: warning: #pragma once in main file
#pragma once
^
BpTreeNode.h:1:9: warning: #pragma once in main file
#pragma once
^
SelectionTree.h:1:9: warning: #pragma once in main file
#pragma once
^
LoanBookHeap.h:1:9: warning: #pragma once in main file
#pragma once
^
SelectionTreeNode.h:1:9: warning: #pragma once in main file
#pragma once
^
LoanBookHeapNode.h:1:9: warning: #pragma once in main file
#pragma once
^
LoanBookData.h:1:9: warning: #pragma once in main file
#pragma once
^
dsds@ubuntu:~/Downloads/77$

```

make를 수행했을 때, pragma once 경고가 발생했지만, 실행파일 수행에 있어서는 문제가 발생하지 않았다.

## LOAD

```

#pragma once
^
LoanBookHeapNode.h:1:9: warning: #pragma once in main file
#pragma once
^
LoanBookData.h:1:9: warning: #pragma once in main file
#pragma once
^
dsds@ubuntu:~/Downloads/77$ ./run
dsds@ubuntu:~/Downloads/77$ cat log.txt
=====LOAD=====
Success
=====
dsds@ubuntu:~/Downloads/77$
if (strcmp(cmd, "LOAD") == 0)
    LOAD();

```

loan\_book.txt (~/Downloads/77) - gedit

1984	100	George Orwell	1949	0	
Korean air	400	Harper	1920	0	
Educated	300	Tare wei	1960	0	
my Booksieeiwo	100	mai	1921	0	

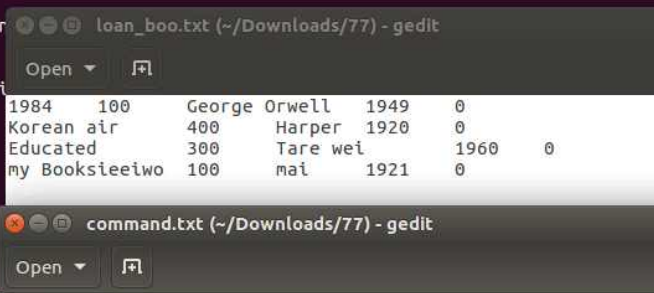
command.txt (~/Downloads/77) - gedit

LOAD

loan\_book.txt 파일에 데이터가 존재하기에 LOAD 명령어를 수행하여 결과를 확인한다. 이후 B+ tree에 LOAD 데이터를 추가한다.

## 예외 처리

```
SelectionTreeNode.h:1:9: warning: #pragma once in main file
#pragma once
^
LoanBookHeapNode.h:1:9: warning: #pragma once in main file
#pragma once
^
LoanBookData.h:1:9: warning: #pragma once in main file
#pragma once
^
dsds@ubuntu:~/Downloads/77$ ./run
dsds@ubuntu:~/Downloads/77$ cat log.txt
=====ERROR=====
100
=====
dsds@ubuntu:~/Downloads/77$ 
if (strcmp(cmd, "LOAD") == 0)
    LOAD();
```




Year	Count	Author	Year	Count
1984	100	George Orwell	1949	0
Korean air	400	Harper	1920	0
Educated	300	Tare wei	1960	0
my Booksieeiwo	100	mai	1921	0

```
command.txt (~/Downloads/77) - gedit
LOAD
```

텍스트 파일의 이름을 loan\_book.txt 파일에서 loan\_boo.txt로 바꾼 후 ERROR 코드를 출력하도록 하였다.

```
LoanBookHeapNode.h:1:9: warning: #pragma once in main file
#pragma once
^
LoanBookData.h:1:9: warning: #pragma once in main file
#pragma once
^
dsds@ubuntu:~/Downloads/77$ ./run
dsds@ubuntu:~/Downloads/77$ cat log.txt
=====ERROR=====
100
=====
dsds@ubuntu:~/Downloads/77$ 
if (strcmp(cmd, "LOAD") == 0)
    LOAD();
else if (strcmp(cmd, "ADD") == 0)
```

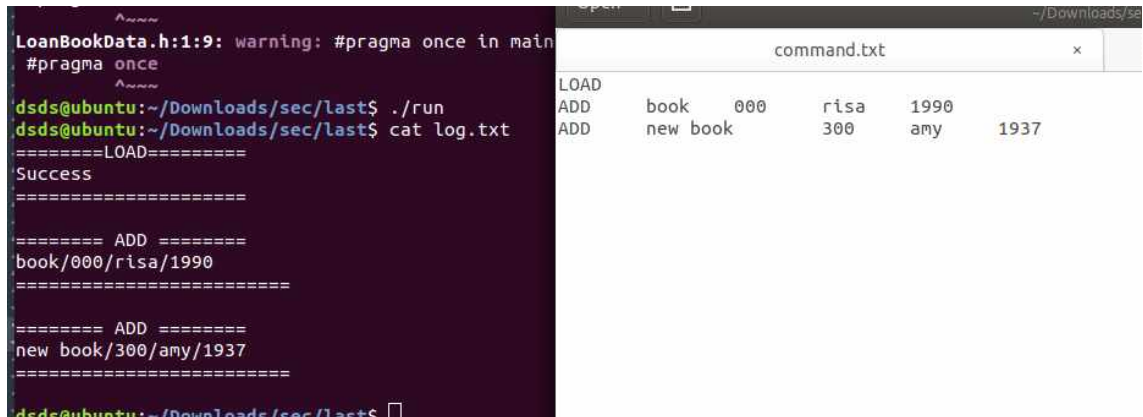


Year	Count	Author	Year	Count
1984	100	George Orwell	1949	0
Korean air	400	Harper	1920	0
Educated	300	Tare wei	1960	0
my Booksieeiwo	100	mai	1921	0

```
command.txt (~/Downloads/77) - gedit
LOAD
```

loan\_book.txt에 데이터가 없기에 LOAD 명령어 수행시 에러코드를 출력하도록 하였다.

## ADD

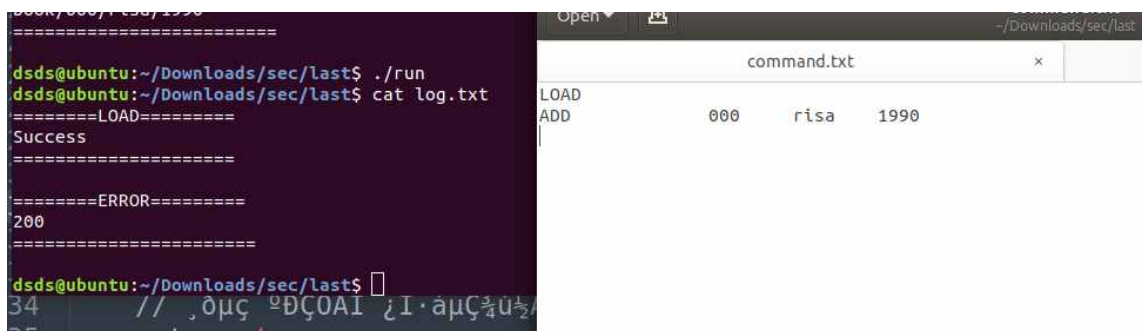


The terminal window shows the execution of a program that successfully adds two records to a B+-tree. The command.txt window shows the following content:

```
command.txt
LOAD
ADD    book    000    risa    1990
ADD    new book  300    any    1937
```

B+-tree에 command.txt로부터 데이터를 읽어와서 추가하는 명령어인 ADD로 4개의 인자를 입력받게 된다.

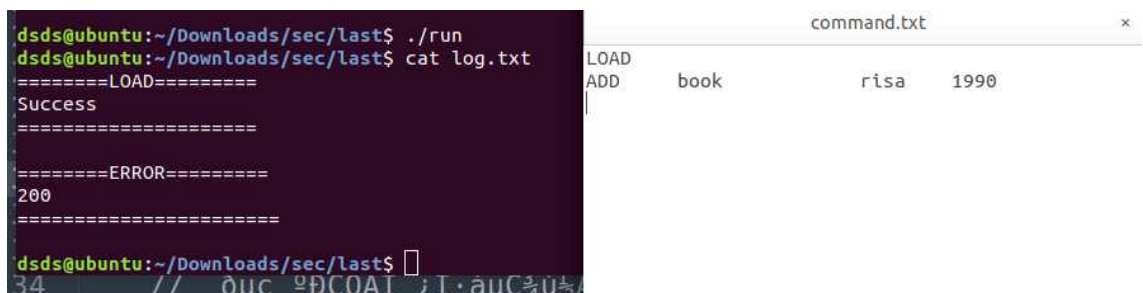
## 예외 처리



The terminal window shows an error message: "200". The command.txt window shows the following content:

```
command.txt
LOAD
ADD    000    risa    1990
```

도서명이 없는 경우



The terminal window shows an error message: "200". The command.txt window shows the following content:

```
command.txt
LOAD
ADD    book    risa    1990
```

분류코드가 없는 경우

```
dsds@ubuntu:~/Downloads/sec/last$ ./run
dsds@ubuntu:~/Downloads/sec/last$ cat log.txt
=====LOAD=====
Success
=====
=====ERROR=====
200
=====

dsds@ubuntu:~/Downloads/sec/last$
```

command.txt			
LOAD			
ADD	book	000	1990

저자명이 없는 경우

```
dsds@ubuntu:~/Downloads/sec/last$ ./run
dsds@ubuntu:~/Downloads/sec/last$ cat log.txt
=====LOAD=====
Success
=====
=====ERROR=====
200
=====

dsds@ubuntu:~/Downloads/sec/last$
```

command.txt			
LOAD			
ADD	book	000	risa

발행 연도가 없는 경우

위는 예외처리를 진행한 화면이며, 도서명이 없는 경우, 분류코드가 없는 경우, 저자명이 없는 경우, 발행 연도가 없는 경우를 나누어 하나라도 없으면 에러코드를 출력하도록 한다.

## SEARCH\_BP

```
dsds@ubuntu:~/Downloads/sec/last$ ./run
dsds@ubuntu:~/Downloads/sec/last$ cat log.txt
=====LOAD=====
Success
=====
===== ADD =====
book/000/risa/1990
=====
===== SEARCH_BP =====
book/000/risa/1990/0
=====
dsds@ubuntu:~/Downloads/sec/last$
```

```
command.txt
LOAD
ADD    book    000    risa    1990
SEARCH_BP    book
```

```
dsds@ubuntu:~/Downloads/sec/last$ ./run
dsds@ubuntu:~/Downloads/sec/last$ cat log.txt
=====LOAD=====
Success
=====
===== ADD =====
book/000/risa/1990
=====
===== SEARCH_BP =====
book/000/risa/1990/0
=====
===== SEARCH_BP =====
tway/600/hong/2012/2
=====
dsds@ubuntu:~/Downloads/sec/last$
```

```
loan_book.txt
korean air    100    kim    2028    1
asiana 200    lee    2020    2
jeju air    300    norh    2021    0
jin air    500    cha    1929    3
tway    600    hong    2012    2
delta    400    dustin    1978    1
finair    000    aoir    1092    3
esta air    700    masion    1999    0

command.txt
LOAD
ADD    book    000    risa    1990
SEARCH_BP    book
SEARCH_BP    tway
```

SEARCH\_BP의 인자로 1개가 입력되는 경우는 해당 도서의 검색 결과를 출력하도록 수행하였다.

```
dsds@ubuntu:~/Downloads/sec/last$ ./run
dsds@ubuntu:~/Downloads/sec/last$ cat log.txt
=====LOAD=====
Success
=====
===== ADD =====
book/000/risa/1990
=====
===== SEARCH_BP =====
asiana/200/lee/2020/2
book/000/risa/1990/0
=====
dsds@ubuntu:~/Downloads/sec/last$
```

```
loan_book.txt
korean air    100    kim    2028    1
asiana 200    lee    2020    2
jeju air    300    norh    2021    0
jin air    500    cha    1929    3
tway    600    hong    2012    2
delta    400    dustin    1978    1
finair    000    aoir    1092    3
esta air    700    masion    1999    0

command.txt
LOAD
ADD    book    000    risa    1990
SEARCH_BP    a    b
```

인자가 (시작 단어, 끝 단어)로 2개가 입력되는 경우 해당 되는 범위의 결과를 출력하도록 하였다.

## 예외 처리

```
dsds@ubuntu:~/Downloads/sec/last$ ./run
dsds@ubuntu:~/Downloads/sec/last$ cat log.txt
=====LOAD=====
Success
=====
===== ADD =====
book/000/risa/1990
=====
=====ERROR=====
300
=====
dsds@ubuntu:~/Downloads/sec/last$
```

air	id	name	year	count
korean	100	kim	2028	1
asiana	200	lee	2020	2
jeju	300	norh	2021	0
jin	500	cha	1929	3
tway	600	hong	2012	2
delta	400	dustin	1978	1
finair	000	aoir	1092	3
esta	700	masion	1999	0

```
LOAD
ADD book 000 risa 1990
SEARCH_BP okok
```

없는 도서명인 okok를 입력했을 때, 에러코드 300이 출력되도록 하였다.

```
dsds@ubuntu:~/Downloads/sec/last$ ./run
dsds@ubuntu:~/Downloads/sec/last$ cat log.txt
=====LOAD=====
Success
=====
===== ADD =====
book/000/risa/1990
=====
=====ERROR=====
300
=====
dsds@ubuntu:~/Downloads/sec/last$
```

air	id	name	year	count
korean	100	kim	2028	1
asiana	200	lee	2020	2
jeju	300	norh	2021	0
jin	500	cha	1929	3
tway	600	hong	2012	2
delta	400	dustin	1978	1
finair	000	aoir	1092	3
esta	700	masion	1999	0

```
LOAD
ADD book 000 risa 1990
SEARCH_BP u y
```

도서명의 범위를 u와 y로 입력하여 도서명이 없기에 에러코드를 출력하도록 하였다.



## PRINT\_BP

```
dsds@ubuntu:~/Downloads/sec/last$ ./run
dsds@ubuntu:~/Downloads/sec/last$ cat log.txt
=====LOAD=====
Success
=====
===== ADD =====
book/000/risa/1990
=====
===== SEARCH_BP =====
asiana/200/lee/2020/2
book/000/risa/1990/0
delta/400/dustin/1978/1
esta air/700/masion/1999/0
=====
=====PRINT_BP=====
asiana/200/lee/2020/2
book/000/risa/1990/0
delta/400/dustin/1978/1
esta air/700/masion/1999/0
finair/000/aoir/1092/3
jeju air/300/norh/2021/0
jin air/500/cha/1929/3
korean air/100/kin/2028/1
tway/600/hong/2012/2
=====
```

air	id	name	year	count
korean air	100	kin	2028	1
asiana	200	lee	2020	2
jeju air	300	norh	2021	0
jin air	500	cha	1929	3
tway	600	hong	2012	2
delta	400	dustin	1978	1
finair	000	aoir	1092	3
esta air	700	masion	1999	0

command	id	name	year
LOAD			
ADD	book	000	risa 1990
SEARCH_BP	a	e	
PRINT_BP			

B+-tree에 저장된 데이터를 도서명을 기준으로 오름차순으로 출력하도록 하였다.

## 예외 처리

```
dsds@ubuntu:~/Downloads/sec/last$ ./run
dsds@ubuntu:~/Downloads/sec/last$ cat log.txt
=====ERROR=====
100
=====
=====ERROR=====
300
=====
=====ERROR=====
400
=====
dsds@ubuntu:~/Downloads/sec/last$
```

air	id	name	year	count
-----	----	------	------	-------

command	id	name	year
LOAD			
SEARCH_BP	a	e	
PRINT_BP			

B+-tree에 저장된 데이터가 없는 경우 예외 코드를 출력하도록 수행하였다.  
loan\_book.txt에 저장되어 있는 데이터가 없기에 예외코드가 출력된다.




## PRINT\_ST

```
===== PRINT_BP =====
asiana/200/lee/2020/2
book/100/risa/1990/0
delta/400/dustin/1978/1
esta air/700/masion/1999/0
finair/000/aoir/1092/3
jin air/500/cha/1929/3
korean air/100/kim/2028/1
new book/300/risa/1990/0
tway/600/hong/2012/2
=====

===== PRINT_ST =====
new book/300/risa/1990/0
=====


dsds@ubuntu:~/Downloads/sec/last$
```

```
Open ▾ 
LOAD
ADD    book    100    risa    1990
ADD    new book    300    risa    1990
SEARCH_BP    a    b
PRINT_BP
PRINT_ST    300
```

```
delta/400/dustin/1978/1
esta air/700/masion/1999/0
finair/000/aoir/1092/3
jin air/500/cha/1929/3
korean air/100/kim/2028/1
new book/400/risa/1990/0
tway/600/hong/2012/2
=====

===== PRINT_ST =====
delta/400/dustin/1978/1
new book/400/risa/1990/0
=====

dsds@ubuntu:~/Downloads/sec/last$
```

```
Open ▾  command.txt
~/Downloads/sec/last
LOAD
ADD    book    100    risa    1990
ADD    new book    400    risa    1990
SEARCH_BP    a    b
PRINT_BP
PRINT_ST    400
```


Selection tree에서 인자로 받은 도서분류코드에 해당하는 데이터를 출력한다.  
도서명을 기준으로 오름차순 출력을 진행한다.

```
jin air/500/cha/1929/3
korean air/100/kim/2028/1
new book/400/risa/1990/0
tway/600/hong/2012/2
=====

===== DELETE =====
Success
=====

ook/100/risa/1990/0
=====

dsds@ubuntu:~/Downloads/sec/last$
```

```
Open ▾  ~/D
LOAD
ADD    book    100    risa    1990
ADD    new book    400    risa    1990
SEARCH_BP    a    b
PRINT_BP
PRINT_ST    100
DELETE
```

구현하던 도중에 코드명 다음에 아무런 명령어가 없으면 정상적인 출력이 되었으나,  
명령어가 있으면 에러가 발생했다.

## 예외 처리

```
===== PRINT_BP =====
asiana/200/lee/2020/2
book/100/risa/1990/0
delta/400/dustin/1978/1
esta air/700/masion/1999/0
finair/000/aoir/1092/3
jeju air/300/norh/2021/0
jin air/500/cha/1929/3
korean air/100/kim/2028/1
new book/300/risa/1990/0
tway/600/hong/2012/2
=====

===== ERROR =====
500
=====

dsds@ubuntu:~/Downloads/sec/last$
```

```
command
~/Downloads/s
Open
LOAD
ADD    book    100    risa    1990
ADD    new book    300    risa    1990
SEARCH_BP    a    b
PRINT_BP
PRINT_ST    111
```

해당되는 코드가 없기에 에러코드 500을 출력하도록 하였다.

## DELETE

```
===== SEARCH_BP =====
asiana/200/lee/2020/2
book/100/risa/1990/0
=====

===== PRINT_BP =====
asiana/200/lee/2020/2
book/100/risa/1990/0
delta/400/dustin/1978/1
esta air/700/masion/1999/0
finair/000/aoir/1092/3
jin air/500/cha/1929/3
korean air/100/kim/2028/1
new book/400/risa/1990/0
tway/600/hong/2012/2
=====

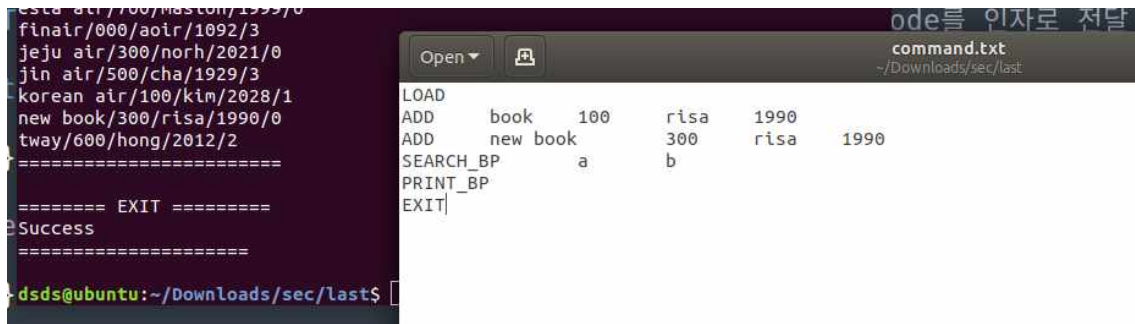
===== DELETE =====
Success
=====

dsds@ubuntu:~/Downloads/sec/last$
```

```
LOAD
ADD    book    100    risa    1990
ADD    new book    400    risa    1990
SEARCH_BP    a    b
PRINT_BP
DELETE
```

DELETE 명령어는 Selection Tree에서 지정된 도서를 제거하도록 수행하도록 하였다. 예외처리를 수행하였으나, 결과의 출력은 어려웠다.

## EXIT



```
esta air/700/maston/1999/0
finalair/000/aoir/1092/3
jeju air/300/norh/2021/0
jin air/500/cha/1929/3
korean air/100/kim/2028/1
new book/300/risa/1990/0
tway/600/hong/2012/2
=====
===== EXIT =====
Success
=====
dsds@ubuntu:~/Downloads/sec/last$
```

```
command.txt
~/Downloads/sec/last
LOAD
ADD    book    100    risa    1990
ADD    new book    300    risa    1990
SEARCH_BP    a    b
PRINT_BP
EXIT
```

EXIT 명령어를 이용하여 프로그램의 메모리를 해제하고, 종료하도록 한다.

### - Consideration

본 프로젝트를 수행하며, tree에 대한 이해를 높일 수 있었다. 도서관에서나 볼 수 있었던 대출 관리 프로그램을 직접 구현해보며, 다양한 예외 처리의 존재를 확인하고, 자료 구조의 구축 방안과 조건에 대하여 다시 한 번 생각해 볼 수 있었던 기회가 될 수 있었다. loan\_book.txt으로부터 data를 읽어와서 command.txt에 존재하는 명령어를 확인하여 원하는 결과를 출력하는 방식의 형식을 구현하며, flog와 heap, selection, b+tree의 구조를 생각할 수 있었다. LOAD는 파일에서 도서 데이터를 읽어와 B+ 트리에 삽입하는 것으로 대량의 데이터를 초기화하거나 업데이트하는 데 사용될 수 있다. ADD 명령어는 새로운 도서 정보를 시스템에 추가한다. 도서관이나 서점에서 새로운 책이 들어올 때 마다 정보를 시스템에 넣을 수 있을 것이라고 생각된다. SEARCH\_BP는 도서 이름이나 범위를 기준으로 검색하는 것으로 사용자가 특정 도서를 찾거나, 특정 범위의 도서를 조회하는 경우 사용되는 명령어일 것이다. PRINT\_BP, PRINT\_ST 명령어는 저장된 도서 정보를 출력하는 것으로 데이터의 전체적인 상황을 파악하거나 특정 분류에 대한 정보를 보여줄 때 사용된다. 프로그램은 다양한 상황에서 에러 메시지를 출력하고, 작업의 성공을 알려주게 된다. 사용자가 프로그램을 더 쉽게 이해하고 사용할 수 있게 한다. 데이터 관리와 검색 기능을 중점으로 두는 복잡한 시스템으로 생각되고, 효율적인 데이터 저장 및 빠른 검색 성능에 중점을 둔다. 파일 입출력을 통한 데이터 로딩 및 로그 기록, 범위 검색, 데이터 추가 및 삭제 기능 등을 포함하고 있어, 사용자의 다양한 요구를 충족시킬 수 있을 정도로 실무에 적합한 프로젝트 난이도였던 것 같다. 전반적으로 데이터 중심의 응용 프로그램 개발에 있어서 중요한 구성 요소들을 포함하고 있으며, 실제 환경에서의 효율적인 데이터 처리에도 필수적인 예외 처리가 있는 것을 이해하였다. 에러 처리와 성공 메시지의 로깅은 사용자 경험과 디버깅에 있어 유용하게 작용할 것으로 생각된다. 데이터 구조와 알고리즘에 대한 깊은 이해를 필요로 하였다.