

텍스트 파일에 내용이 없는 경우 시나리오 사진

```
C:\Users\JeesooPark\source\repos\Assignment2\Debug\Assigni
최대 저장가능한 책의 개수를 입력해주세요
5
1. 도서입력
2. 저자별검색
3. 제목검색
4. 가격 순으로 정렬
5. 전체 도서 정보 출력
6. 끝
```

도서 입력

```
1.
1
저자 :박지수
책 제목 :초급C언어
가 격 :1000
1. 도서입력
2. 저자별검색
3. 제목검색
4. 가격 순으로 정렬
5. 전체 도서 정보 출력
6. 끝
1
저자 :우현제
책 제목 :고급C언어
가 격 :3000
1. 도서입력
2. 저자별검색
3. 제목검색
4. 가격 순으로 정렬
5. 전체 도서 정보 출력
6. 끝
1
저자 :정민
책 제목 :중급C언어
가 격 :2000
1. 도서입력
2. 저자별검색
3. 제목검색
4. 가격 순으로 정렬
5. 전체 도서 정보 출력
6. 끝
2
최대 저장가능한 책의 개수를 입력해주세요
```

저자검색 후 책 내용 수정하기

```
5. =
2.
저자명을 입력해주세요
우현제
우현제가 작성한 책은 고급C언어이며 3000의 가격을 가지고 있습니다.
1.도서 정보 수정
2.도서 정보 삭제
1
도서 정보 수정 메뉴입니다.
변경할 정보를 순서대로 입력해주세요
저자 :박준현
책 제목 :고급C언어
가 격 :3500
1. 도서입력
2. 저자별검색
3. 제목검색
4. 가격 순으로 정렬
5. 전체 도서 정보 출력
6. 끝
```

제목검색후 도서 삭제

```
해당 도서가 삭제 되었습니다
1. 도서입력
2. 저자별검색
3. 제목검색
4. 가격 순으로 정렬
5. 전체 도서 정보 출력
6. 끝
5
=== 전체 책 목록 ===
제목: 중급C언어
저자: 정민
가격: 2000원

제목: 초급C언어
저자: 박지수
가격: 1000원

1. 도서입력
2. 저자별검색
3. 제목검색
4. 가격 순으로 정렬
5. 전체 도서 정보 출력
6. 끝
1
```

가격순 정렬

```
4
정렬 후 금액
3500
2000
1. 도서입력
2. 저자별검색
3. 제목검색
4. 가격 순으로 정렬
5. 전체 도서 정보 출력
6. 끝
```

전체 도서 정보 출력

```
5. ==
5
=== 전체 책 목록 ===
제목: 중급C언어
저자: 박준현
가격: 3500원

제목: 중급C언어
저자: 정민
가격: 2000원

1. 도서입력
2. 저자별검색
3. 제목검색
4. 가격 순으로 정렬
5. 전체 도서 정보 출력
6. 끝
```

종료

6번을 누르면 아래의 문구와 함께 프로그램이 종료됩니다.

```
C:\Users\JeesooPark\source\repos\Assignment2\Debug\Assignment2.exe(프로세스 23824개)이(가) 종료되었습니다(코드: 0개).
디버깅이 중지될 때 콘솔을 자동으로 닫으려면 [도구] -> [옵션] -> [디버깅] > [디버깅이 중지되면 자동으로 콘솔 닫기]를 사용하도록 설정합니다.
이 창을 닫으려면 아무 키나 누르세요...
```

개선사항 1 헤더파일과 소스 파일을 기능 별로 적절하게 나누기

```

└─ 소스 파일
    ├── book.c
    ├── Filelo.c
    ├── main.c
    └── menu.c
└─ 헤더 파일
    ├── book.h
    ├── Filelo.h
    └── menu.h
```

book.c

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4 #include "book.h"
5
6
7 BOOK* AllocateBooks(int count)
8 {
9     BOOK* result = NULL;
10    result = (BOOK*)malloc(sizeof(BOOK) * count);
11    if (result == NULL)
12        printf("중적 메모리 할당 실패\n");
13    else
14        memset(result, 0, sizeof(BOOK) * count);
15    return result;
16
17 void InputBookInfo(BOOK* book, int count)
18 {
19     printf("저자 :");
20     gets(book[count].writer);
21     printf("책 제목 :");
22     gets(book[count].title);
23     printf("가 격 :");
24     scanf("%d", &book[count].price);
25
26
27
28
```

Allocate함수와 책의 정보를 입력하는 InputBookInfo메소드를 Book.c에 배치하였습니다.

book.h

```
1 #ifndef book_h
2 #define book_h
3 typedef struct book {
4     char title[40];
5     char writer[20];
6     int price;
7 }BOOK;
8 BOOK* AllocateBooks(int count);
9 void InputBookInfo(BOOK* contacts, int count);
10 #endif
```

헤더파일은 ifndef를 이용하여 book.h가 중복선언되지 않게 막았으며 book구조체와 book.c에서 사용되는 메소드들을 선언하였습니다.

menu.h

```
1 #ifndef menu_h
2 #define menu_h
3 int compareByPrice(const void* b1, const void* b2);
4 int compareByWriter(const void* b1, const void* b2);
5 int compareByTitle(const void* b1, const void* b2);
6 void changeBookInfo(BOOK* book, int location);
7 void printBook(const BOOK* book, int count);
8 void deleteBookInfo(BOOK* book, int location, int size);
9 #endif // 11
10
```

menu.h는 book.h와 동일하며 주로 메뉴에서 사용되는 메소드들을 선언하였습니다.

menu.c

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4 #include "book.h"
5 #include "menu.h"
6
7 void changeBookInfo(BOOK* book, int location)
8 {
9     printf("변경할 정보를 순서대로 입력해주세요\n");
10    printf("저자 :");
11    gets(book[location].writer);
12    printf("책 제목 :");
13    gets(book[location].title);
14    printf("가 격 :");
15    scanf("%d", &book[location].price);
16
17
18 void deleteBookInfo(BOOK* book, int location, int count) {
19
20     for (int i = location; i < count; i++) {
21         book[i] = book[i + 1];
22     }
23
24
25
```

책의 정보를 변경하기 위한 method로 bsearch를 통해 책의 존재여부를 확인 한 후 bsearch의 결과물을 담은 research와 book의 값을 뺌으로써 책의 위치를 찾은 후 해당 책의 정보를 수정할 수 있도록 메소드를 만들었습니다.

```

22 void printBook(const BOOK* book, int count) {
23     printf(" == 전체 책 목록 == \n");
24     for (int i = 0; i < count; i++) {
25         printf("제목: %s\n", book[i].title);
26         printf("저자: %s\n", book[i].writer);
27         printf("가격: %d원\n", book[i].price);
28         printf("\n");
29     }
30 }
31 int compareByWriter(const void* b1, const void* b2) {
32     const BOOK* B1 = (const BOOK*)b1;
33     const BOOK* B2 = (const BOOK*)b2;
34     return strcmp(B1->writer, B2->writer);
35 }
36 int compareByTitle(const void* b1, const void* b2) {
37     const BOOK* B1 = (const BOOK*)b1;
38     const BOOK* B2 = (const BOOK*)b2;
39     return strcmp(B1->title, B2->title);
40 }

```

책의 전체 정보를 출력하는 Printbook메소드와 bsearch와 qsort를 진행하기 위한 compare 메소드입니다.

```

40 }
41 int compareByPrice(const void* b1, const void* b2) {
42     const BOOK* B1 = (const BOOK*)b1;
43     const BOOK* B2 = (const BOOK*)b2;
44     if (((B1->price == B2->price))) {
45         return 0;
46     }
47     else if ((B1->price < B2->price)) {
48         return 1;
49     }
50     else if ((B1->price > B2->price)) {
51         return -1;
52     }
53 }
54 }

```

Qsort와 Bsearch를 오름차순으로 가격을 비교하기 위한 메소드입니다.

Fileo.c

```

1 #include "Fileo.h"
2 #include<stdio.h>
3 void saveBook(BOOK* book,int counter,int count) {
4     FILE* fp;
5     fp = fopen("data.txt", "w+");
6     if (fp == NULL) {
7         printf("해당파일이 없습니다.");
8     }
9     fprintf(fp, "%d\n", count);
10    fprintf(fp, "%d\n", counter);
11    for (int i = 0; i < counter; i++) {
12        fprintf(fp, "%s %s %d\n", book[i].title, book[i].writer, book[i].price);
13    }
14    fclose(fp);
15 }
16 void loadBOOK(int* counter, int* count) {
17     FILE* fp;
18     fp = fopen("data.txt", "r");
19     if (fp == NULL) {
20         printf("해당파일이 없습니다.");
21     }
22     else {
23         fscanf(fp, "%d\n", count);
24         fscanf(fp, "%d\n", counter);
25     }
26     fclose(fp);
27 }
28 }
29 }

```

구조체 배열에 작성된 책의 정보를 txt파일에 저장하는 savebook메소드와 txt파일에 작성된 구조체 배열의 전체 크기와 현재 저장된 도서의 개수를 가지고 오는 loadBook입니다.

```

void loadBOOKInfo(BOOK* book,int counter,int count) {
    FILE* fp;
    fp = fopen("data.txt", "a+");

    int saveCount,saveCounter;
    fscanf(fp, "%d\n", &saveCount);
    fscanf(fp, "%d\n", &saveCounter);

    for (int i = 0;i < count;i++) {
        fscanf(fp, "%s %s %d\n", book[i].title, book[i].writer, &book[i].price);
    }
    fclose(fp);
}

```

책의 정보를 구조체 배열에 써주는 loadBOOKInfo메소드입니다.

Filelo.h

```

1 #ifndef Filelo_h
2 #define Filelo_h
3 #include "book.h"
4 void saveBook(BOOK* book, int counter,int count);
5 void loadBOOK(int* counter, int* count);
6 void loadBOOKInfo(BOOK* book,int counter,int count);
7 #endif

```

main.c

```

#include <stdio.h>
#include<stdlib.h>
#include<string.h>

```

```

#include "book.h"
#include "Filelo.h"
#include "menu.h"

```

```

int main(void) {
    BOOK* book = NULL;
    BOOK* temp = NULL;
    BOOK* result, writer, title;
    int count = 0;
    int counter = 0;
    int *cPoint = &count;
    int *ctPoint = &counter;

    loadBOOK(ctPoint, cPoint);
    if (count == 0) {
        printf("최대 저장가능한 책의 개수를 입력해주세요\n");
        scanf("%d", &count);
    }
}

```

```

        while (getchar() != '\n');
        book = AllocateBooks(count);
    }else {
        book = AllocateBooks(count);
        loadBOOKInfo(book,counter,count);
    }
    int select = 1;
    char key[40] = { 0 };
    while (1) {
        printf("1. 도서입력\n");
        printf("2. 저자별검색\n");
        printf("3. 제목검색\n");
        printf("4. 가격 순으로 정렬\n");
        printf("5. 전체 도서 정보 출력\n");
        printf("6. 끝\n");
        scanf("%d", &select);
        switch (select)
        {
            case 1:
                if (counter == count) {
                    printf("더이상의 책 정보를 입력할 수 없습니다.\n");
                    int select2 = 0;
                    printf("공간 수정을 원하시면 1번을 눌러주세요\n");
                    scanf("%d", &select2);
                    switch (select2)
                    {
                        case 1:
                            printf("용량이 늘어났습니다.\n");
                            temp = book;
                            count += count;
                            int* resize = &count;
                            book = (BOOK*)realloc(book, (sizeof(BOOK)*
(*resize)));

                            if (book == NULL) {
                                book = temp;
                            }
                            break;
                        default:
                            printf("메뉴로 돌아갑니다.\n");
                            break;
                    }
                }

```

```

    }
    else {
        if (book == NULL) {
            return -1;
        }
        while (getchar() != '\n');
        InputBookInfo(book, counter);
        counter = counter + 1;
    }
    break;
case 2:
    printf("저자명을 입력해주세요\n");
    getchar();
    gets(writer.writer);
    qsort(book, count, sizeof(*book), compareByWriter);
    result = bsearch(&writer, book, count, sizeof(BOOK),
compareByWriter);
    if (result != NULL) {
        printf("%s가 작성한 책은 %s이며 %d의 가격을 가지고 있
습니다.\n", result->writer, result->title, result->price);
        int select4 = 0;
        int location = result - book;
        printf("1.도서 정보 수정\n");
        printf("2.도서 정보 삭제\n");
        scanf("%d", &select4);
        switch (select4)
        {
            case 1:
                printf("도서 정보 수정 메뉴입니다.\n");

                while (getchar() != '\n');
                changeBookInfo(book, location);
                break;
            case 2:
                printf("해당 도서가 삭제 되었습니다\n");
                while (getchar() != '\n');
                deleteBookInfo(book, location, count);
                counter = counter - 1;
                break;
            default:
                printf("메뉴로 돌아갑니다.");

```



```

        break;
    }
}
else {
    printf("해당 도서가 존재하지 않습니다.\n");
}
break;
case 3:
    printf("도서명을 입력해주세요\n");
    getchar();
    gets(title.title);
    qsort(book, count, sizeof(*book), compareByTitle);
    result = bsearch(&title, book, count, sizeof(BOOK),
compareByTitle);

    if (result != NULL) {
        printf("%s의 책은 %s가 작성하였고 %d의 가격입니
다.\n", result->title, result->writer, result->price);
        int select3 = 0;
        int location = result - book;
        printf("1.도서 정보 수정\n");
        printf("2.도서 정보 삭제\n");
        scanf("%d", &select3);
        switch (select3)
        {
            case 1:
                printf("도서 정보 수정 메뉴입니다.\n");

                while (getchar() != '\n');
                changeBookInfo(book, location);

                break;
            case 2:
                printf("해당 도서가 삭제 되었습니다\n");
                while (getchar() != '\n');
                deleteBookInfo(book, location, count);
                counter = counter - 1;
                break;
            default:
                printf("메뉴로 돌아갑니다.");
                break;
        }
    }
}

```

```

    }
    else {
        printf("해당 도서가 존재하지 않습니다.\n");
    }
    break;
case 4:
    qsort(book, count, sizeof(*book), compareByPrice);
    printf("정렬 후 금액\n");
    for (int i = 0; i < counter; i++) {
        printf("%d\n", book[i].price);
    }
    break;
case 5:
    qsort(book, count, sizeof(*book), compareByPrice);
    printBook(book, counter);
    break;
case 6:
    saveBook(book, counter, count);
    free(book);
    book = NULL;
    exit(0);
    break;
default:
    printf("잘못된 입력입니다.\n");
    break;
}
}
return 0;
}

```

Txt에 파일 정보가 기록되어져 있는 경우

```
C:\Users\JeesooPark\source\repos\Assign1>
1. 도서입력
2. 저자별검색
3. 제목검색
4. 가격 순으로 정렬
5. 전체 도서 정보 출력
6. 끝
5
=== 전체 책 목록 ===
제목: 중급C언어
저자: 정민
가격: 2000원

제목: 초급C언어
저자: 박지수
가격: 1000원

1. 도서입력
2. 저자별검색
3. 제목검색
4. 가격 순으로 정렬
5. 전체 도서 정보 출력
6. 끝
```

데이터가 꽉찬 경우 realloc을 사용하여 count의 크기 * 2만큼 공간이 늘어납니다.

```
1
저자 :김정호
책 제목 :분산프로그래밍
가 격 :4000
1. 도서입력
2. 저자별검색
3. 제목검색
4. 가격 순으로 정렬
5. 전체 도서 정보 출력
6. 끝
1
더이상의 책 정보를 입력할 수 없습니다.
공간 수정을 원하시면 1번을 눌러주세요
```

그림 2처럼 용량을 늘린후 추가 데이터를 입력할 수 있습니다.

```
더이상의 책 정보를 입력할 수 없습니다.  
공간 수정을 원하시면 1번을 눌러주세요  
1  
용량이 늘어났습니다.  
1. 도서입력  
2. 저자별검색  
3. 제목검색  
4. 가격 순으로 정렬  
5. 전체 도서 정보 출력  
6. 끝  
1  
저자 : 김호남  
책 제목 : 팀프로젝트I  
가 격 : 2000  
1. 도서입력  
2. 저자별검색  
3. 제목검색  
4. 가격 순으로 정렬  
5. 전체 도서 정보 출력  
6. 끝
```

프로그램 종료시 textfile에 정보가 저장됩니다.



10

6

중급C언어 정민 2000

초급C언어 박지수 1000

고급C언어 박준현 3500

초고급C언어 우현제 4000

분산프로그래밍 김정호 4000

팀프로젝트I 김호남 2000