To Do List

5조

조장 : 장 태 연
조원 : 김 진 규
이 용 희
이 현 민
최 호 준

CONTENTS

01 | 프로젝트 개요

01

프로젝트 개요

프로젝트 명 : **개인 일정 관리 프로그램 By C**

프로젝트 **목적**

- C프로그래밍 문법을 활용하여 바쁜 일상 속에서 잊어버리기 쉬운 개인 일정 기록 프로그램 제작

- 간단한 조작 방법으로 여러가지 기능을 사용하며 한 눈에 보기 편한 프로그래밍

프로젝트 **기간**

- 2022. 07. 29 ~ 2022. 08. 16

- 세부 일정은 프로젝트 수행 절차(p7) 참고

02 | 팀 구성원 역할 소개

02

팀 구성원 역할 소개


장 태 연
발표, 기능 구현
김 진 규
데이터 수집, 발표 자료 정리
이 용 희
기능 구현, PPT 제작
이 현 민
데이터 수집, 발표 자료 정리
최 호 준
데이터 수집, 발표 자료 정리

03 프로젝트 수행 절차

01. 프로젝트 세부 일정

02. 프로그램 알고리즘

03

프로젝트 수행 절차

01. 프로젝트 세부 일정

02. 프로그램 알고리즘

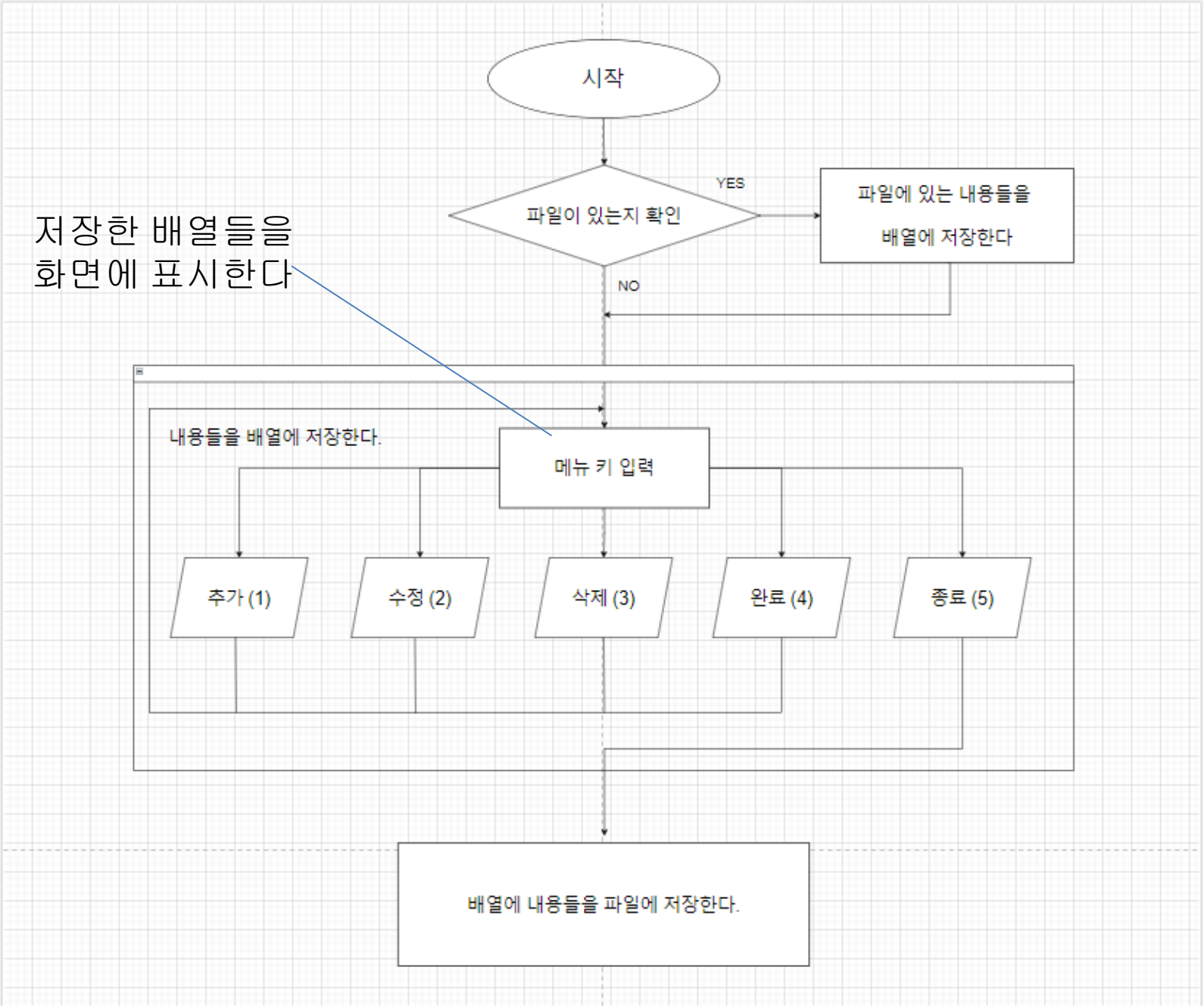
03

01. 프로젝트 세부 일정

02 . 프로그램 알고리즘

프로그램 알고리즘

순서도



03

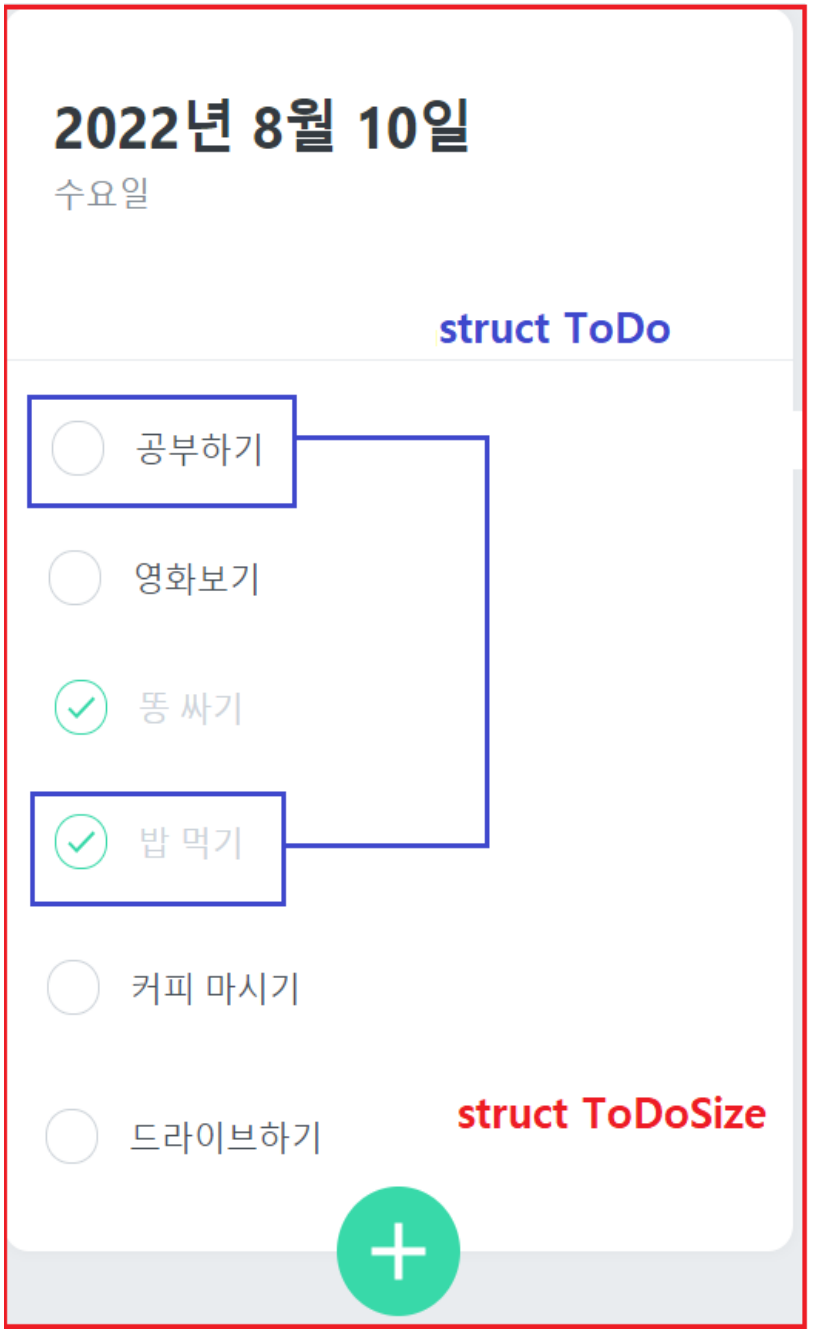
01. 프로젝트 세부 일정

02 . 프로그램 알고리즘

프로그램 알고리즘

구조체 선언

```
struct ToDo {
    char* str;      // 할일을 적는 문자열
    bool completed; // 참,거짓
};

typedef struct ToDo* ToDoRef;

struct ToDoSize {
    int number;     // 할일 순번
    int maximum;    // 할일 최대 수용량
    ToDoRef* list;  // struct Todo의 값을 참조
};
typedef struct ToDoSize* ToDoSizeRef;
```



03

01. 프로젝트 세부 일정

02 . 프로그램 알고리즘

프로그램 알고리즘

```
void start()
{
    ToDoSizeRef list = ToDoSizeInit();
    FILE* open_file;
    fopen_s(&open_file, FILENAME,"rb");
    if (open_file != NULL)
    {
        ToDoListLoadFromFile(list, open_file);
        fclose(open_file);
    }
    else
    {
        welcome_Todo();
    }
```

start 함수

```
    while (finish)
    {
        system("cls");
        textColor(15);
        printf("\n                    메 뉴  \n\n");
        printf(" ① 추 가 ");
        printf(" ② 삭 제 ");
        printf(" ③ 수 정 ");
        printf(" ④ 완 료 ");
        printf(" ⑤ 종 료 \n\n");
        printf("                To Do List\n\n");
        show_File(list);
        printf("\n\n▶▶▶▶▶▶ ");
        char n;
        scanf_s("%c", &n);
        switch (n)
        {
        case '1':
            add_List(list);
            system("cls");
            break;
        case '2':
            del_List(list);
            system("cls");
            break;
        case '3':
            rename_List(list);
            system("cls");
            break;
        case '4':
            done_List(list);
            system("cls");
            break;
        case '5':
            system("cls");
            exit_Program();
            finish = false;
            break;
        default: break;
        }
    }
    fopen_s(&open_file, FILENAME, "wb");
    ToDoListSave(list,open_file);
    fclose(open_file);
}
```

03

01. 프로젝트 세부 일정

02 . 프로그램 알고리즘

프로그램 알고리즘

```
struct ToDo {
    char* str;      // 할일을 적는 문자열
    bool completed;  // 참,거짓
};

typedef struct ToDo* ToDoRef;

struct ToDoSize {
    int number;     // 할일 순번
    int maximum;    // 할일 최대 수용량
    ToDoRef* list;   // struct Todo의 값을 참조
};
typedef struct ToDoSize* ToDoSizeRef;
```

구조체 선언
크기 할당

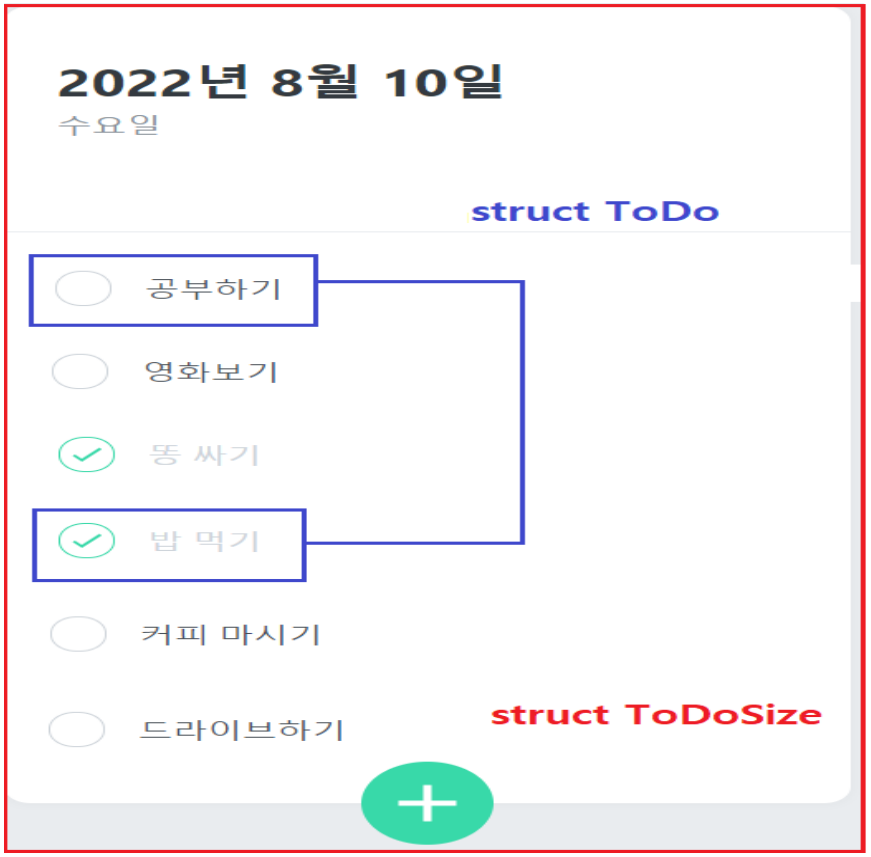
```
void start()
{
    ToDoSizeRef list = ToDoSizeInit();
```

```
ToDoSizeRef ToDoSizeInit()
{
    ToDoSizeRef list = (ToDoSizeRef)malloc(sizeof(struct ToDoSize));
    list->number = 0;
    list->maximum = 100;
    list->list = (ToDoRef*)malloc(sizeof(ToDoRef) * list->maximum);
    return list;
}
```

list.number

list.maximum

list.list



03

01. 프로젝트 세부 일정

02 . 프로그램 알고리즘

프로그램 알고리즘

While 문

```
while (finish)
{
    system("cls");
    textColor(15);
    printf("\n                    메 뉴  \n\n");
    printf(" ① 추 가 ");
    printf(" ② 삭 제 ");
    printf(" ③ 수 정 ");
    printf(" ④ 완 료 ");
    printf(" ⑤ 종 료 \n\n");
    printf("                  To Do List\n\n");
    show_File(list);
    printf("\n\n▶▷▶▷▶▷▶ ");
    char n;
    scanf_s("%c", &n);
    switch (n)
    {
    case '1':
        add_List(list);
        system("cls");
        break;
    case '2':
        del_List(list);
        system("cls");
        break;
    case '3':
        rename_List(list);
        system("cls");
        break;
    case '4':
        done_List(list);
        system("cls");
        break;
    case '5':
        system("cls");
        exit_Program();
        finish = false;
        break;
    default: break;
    }
```

03

01. 프로젝트 세부 일정

02 . 프로그램 알고리즘

프로그램 알고리즘

```
struct ToDo {
    char* str;      // 할일을 적는 문자열
    bool completed;  // 참,거짓
};

typedef struct ToDo* ToDoRef;

struct ToDoSize {
    int number;     // 할일 순번
    int maximum;    // 할일 최대 수용량
    ToDoRef* list;   // struct Todo의 값을 참조
};
typedef struct ToDoSize* ToDoSizeRef;
```

```
case '1':
    add_List(list);
    system("cls");
    break;
```

```
ToDoRef ToDoListAdd(ToDoSizeRef list, char* name)
{
    if (list->number == list->maximum)
        return NULL;
    ToDoRef newTodo = ToDoCreate(name);
    (list->list)[list->number] = newTodo;
    list->number += 1;
    return newTodo;
}
```

Add_list

```
void add_List(ToDoSizeRef list)
{
    rewind(stdin);
    char name[MAX];
    memset(name, 0, MAX);
    textColor(15);
    printf("\n추 가 내 용\n▶ ");
    gets(name);
    ToDoListAdd(list, name);
}
```

```
ToDoRef ToDoCreate(char* name)
{
    ToDoRef newTodo = (ToDoRef)malloc(sizeof(struct ToDo));
    newTodo->str = (char*)malloc(sizeof(char) * strlen(name) + 1);
    strcpy(newTodo->str, name);
    newTodo->completed = false;
    return newTodo;
}
```

03

01. 프로젝트 세부 일정

02 . 프로그램 알고리즘

프로그램 알고리즘

```
case '2':
    del_List(list);
    system("cls");
    break;
```

Del_list

```
void del_List(ToDoSizeRef list)
{
    int i;
    bool selected = false;
    while (!selected)
    {
        textColor(15);
        printf("\n삭제할 번호 ▶ ");
        scanf_s("%d", &i);
        if (i <= list->number)
            selected = true;
        ToDoListRemoveIndex(list, i-1);
    }
}
```

```
void ToDoListRemoveIndex(ToDoSizeRef list, int index)
{
    if (index >= list->number || list->number == 0)
        return;
    ToDoRef toDel = (list->list)[index];
    int i = index;
    while (i < (list->number) - 1)
    {
        (list->list)[i] = (list->list)[i + 1];
        i++;
    }
    list->number -= 1;
}
```

03

01. 프로젝트 세부 일정

02 . 프로그램 알고리즘

프로그램 알고리즘

Rename_list

```
case '3':
    rename_List(list);
    system("cls");
    break;
```

```
void rename_List(ToDoSizeRef list)
{
    int i;
    ToDoRef toRename;
    bool selected = false;
    char str[MAX];
    memset(str, 0, MAX);
    while (!selected)
    {
        textColor(15);
        printf("\n수정할 번호 ▶ ");
        scanf_s("%d", &i);
        if ((list->list)[i - 1]->completed == true)
        {
            printf("\n\t  번호 다시 입력하세요.\n");
            continue;
        }
        if (i <= list->number)
            selected = true;
        toRename = (list->list)[i-1];
        rewind(stdin);
        printf("\n새로운 내용 입력\n▶▷");
        gets(str);
        ToDoRename(toRename, str);
    }
}
```

```
void ToDoRename(ToDoRef toRename, char* str)
{
    if (strlen(toRename->str) < strlen(str))
        toRename->str = (char*)realloc(toRename->str, sizeof(char) * (strlen(str) + 1));
    memset(toRename->str, 0, strlen(toRename->str) + 1);
    strcpy(toRename->str, str);

}
```

03

01. 프로젝트 세부 일정

02 . 프로그램 알고리즘

프로그램 알고리즘

Done_list

```
case '4':
    done_List(list);
    system("cls");
    break;
```

```
void done_List(ToDoSizeRef list)
{
    ToDoRef toDone;
    int i;
    bool selected = false;
    while (!selected)
    {
        textColor(15);
        printf("\n완료한 번호 선택 ▶ ");
        scanf_s("%d", &i);
        if (i <= list->number)
            selected = true;
        toDone = (list->list)[i-1];
        toDone->completed = true;
    }
}
```

03

01. 프로젝트 세부 일정

02 . 프로그램 알고리즘

프로그램 알고리즘

fopen()

fprintf()

fclose()

```
FILE* open_file;
fopen_s(&open_file, FILENAME, "wb");
fwrite(&num_List, sizeof(int), 1, open_file);

fwrite(&length_List, sizeof(int), 1, open_file);
fwrite(current->str, sizeof(char), length_List, open_file);
fwrite(&(current->completed), sizeof(bool), 1, open_file);


fclose(open_file);
```

저장한 배열을

File에 저장하기

파일모드	기능	설명
"w"	쓰기 전용	새 파일을 생성합니다. 만약 파일이 있으면 내용을 덮어씁니다.

b는 바이너리모드 (구조체 쓸 때)

03

01. 프로젝트 세부 일정

02 . 프로그램 알고리즘

프로그램 알고리즘

저장한 배열을

File에 저장하기

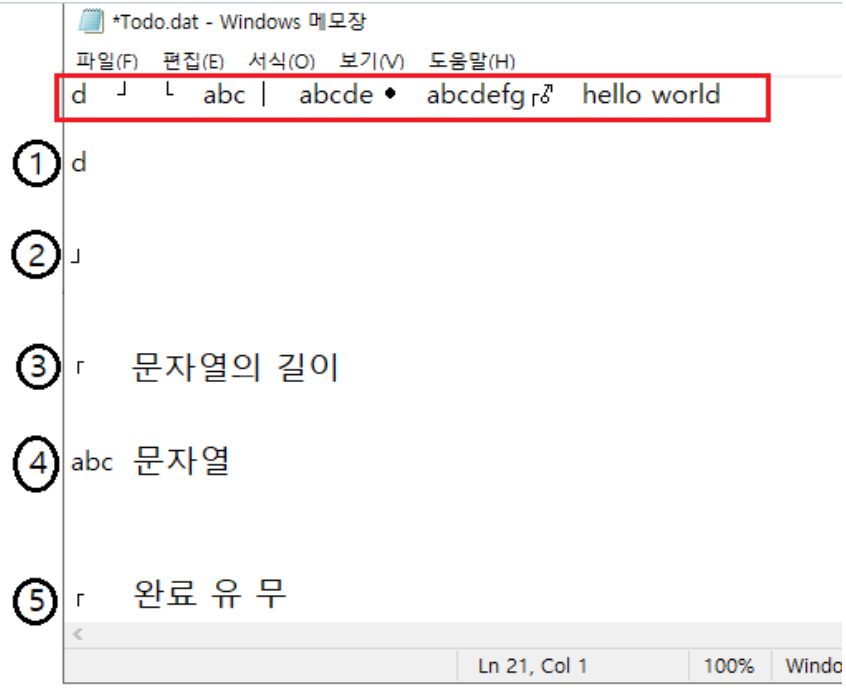
```
FILE* open_file;
```

```
fopen_s(&open_file, FILENAME, "wb");
ToDoListSave(list,open_file);
fclose(open_file);
```

```
To Do List

1 : abc
2 : abcde
3 : abcdefg
4 : hello world
```

```
void ToDoListSave(ToDoSizeRef list, FILE* open_file)
{
    int num_List = list->number;
    int length_List;
    int maximum_List = list->maximum;
    ToDoRef current;
    fwrite(&maximum_List, sizeof(int), 1, open_file);


    fwrite(&num_List, sizeof(int), 1, open_file);


    int i = 0;
    while (i < num_List)
    {
        current = (list->list)[i];
        length_List = strlen(current->str);


        fwrite(&length_List, sizeof(int), 1, open_file);


        fwrite(current->str, sizeof(char), length_List, open_file);


        fwrite(&(current->completed), sizeof(bool), 1, open_file);
        i++;
```

① ② ③ ④ ⑤



03

01. 프로젝트 세부 일정

02 . 프로그램 알고리즘

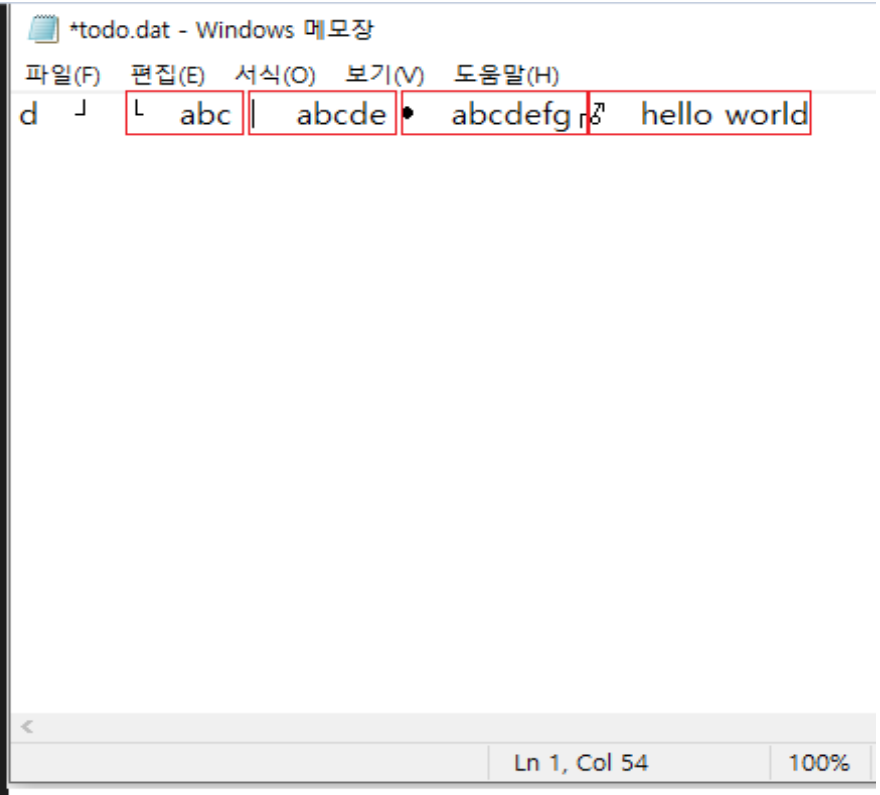
프로그램 알고리즘

저장된 파일을
배열에 저장하기

```
FILE* open_file;
fopen_s(&open_file, FILENAME,"rb");
if (open_file != NULL)
{
    ToDoListLoadFromFile(list, open_file);
    fclose(open_file);
}
```

```
To Do List

1 : abc
2 : abcde
3 : abcdefg
4 : hello world
```

```
void ToDoListLoadFromFile(ToDoSizeRef list, FILE* open_file)
{
    list->number = 0;
    int num_List;
    int i = 0;
    ToDoRef newToDo;
    int length_Str;
    bool completed;
    char* str1;
    fread(&(list->maximum), sizeof(int), 1, open_file);
    fread(&num_List, sizeof(int), 1, open_file);
    while (i < num_List)
    {
        fread(&length_Str, sizeof(int), 1, open_file);
        str1 = (char*)calloc(sizeof(char), length_Str + 1);
        fread(str1, sizeof(char), length_Str, open_file);
        fread(&completed, sizeof(bool), 1, open_file);
        newToDo = ToDoCreate(str1);
        newToDo->completed = completed;
        free(str1);
        if (list->number != list->maximum)
        {
            (list->list)[list->number] = newToDo;
            list->number += 1;
        }
        i++;
    }
}
```

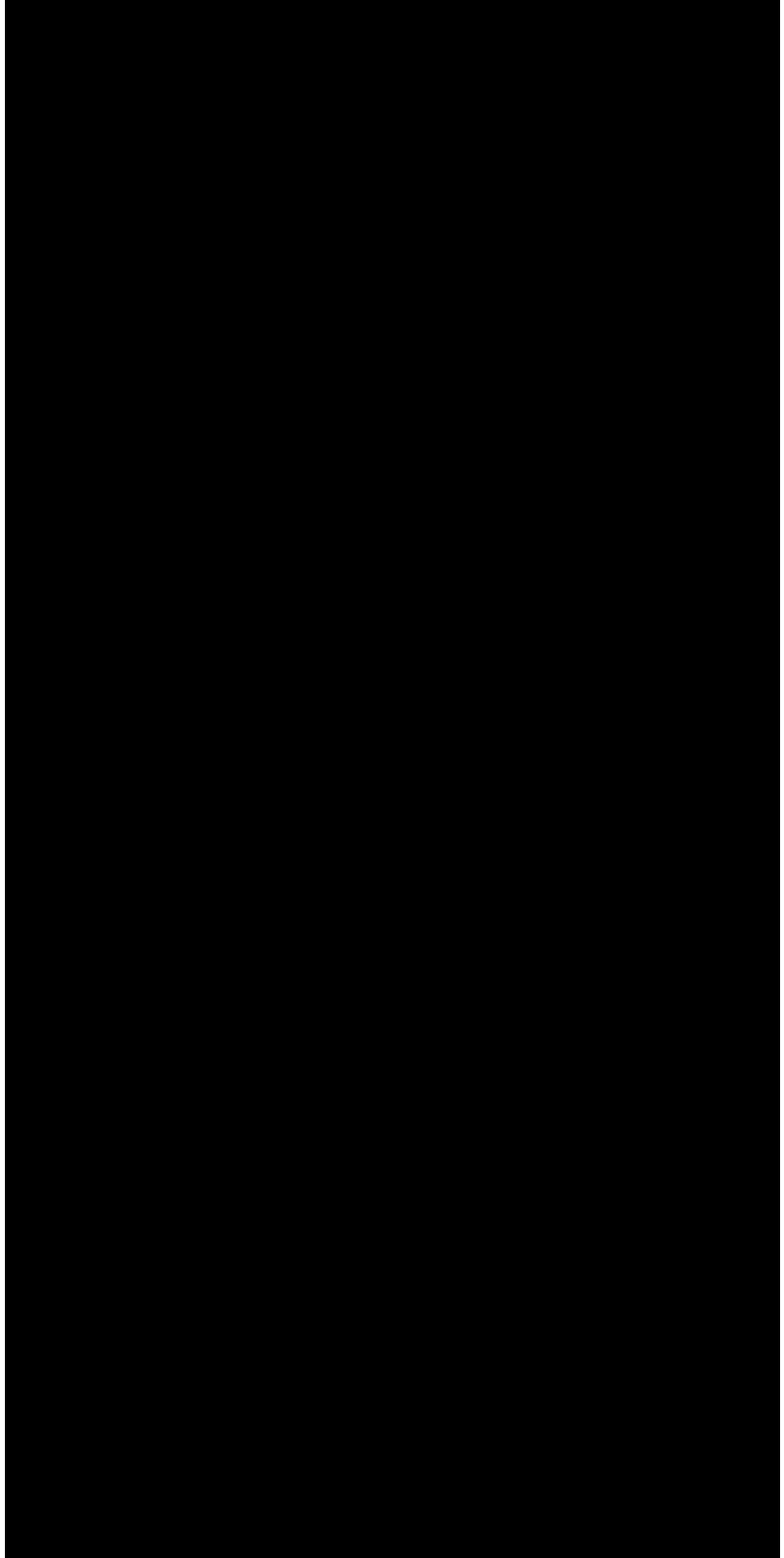


04 | 프로젝트 수행 결과

04

프로젝트 수행 결과

-영상시연



05 | 자체 평가 의견

05

자체 평가 의견

장 태 연	김 진 규	이 용 희	이 현 민	최 호 준
배열에 관련하여서는 큰 어려움이 없었지만 구조체와 파일 스트림에 대해서 더 공부해야겠다 생각되었습니다.	프로젝트를 하면서 구조체와 배열에 대해서 공부할 수 있어서 도움이 되었습니다. 어려웠지만 코딩 실력이 향상되었습니다.	프로젝트를 진행하면서 구조체, 포인터에 관한 부족한 개념을 공부할 수 있어서 좋았습니다.	이해가 안되는 부분이 많았지만 팀원들과 함께하며 이해할 수 있었고 많은 도움이 되었습니다.	수업내용을 완벽하게 이해하지 못한 상태에서 프로젝트에 참여하게 되어 어려운 점이 있었고, 코드를 이해하는데 시간이 좀 걸렸지만 공부하는데 도움이 되었습니다.

감사합니다