1. 서론
    A. 프로젝트 목적 및 배경:7주차까지 배운 내용에 대한 실습을 위해 진행
    B. 목표:TODO리스트 만들기
2. 본론
    A. 요구사항
        i. 사용자 요구사항: 사용자가 할 일을 입력, 삭제, 출력할 수 있는 프로그램
        ii. 기능 요구사항: 할 일 입력, 삭제, 출력
    B. 설계 및 구현
        i. 기능 별 구현 사항
    C. 테스트
        i. 기능 별 테스트 결과
        ii. 최종 테스트 스크린샷
3. 결과 및 결론
    A. 프로젝트 결과: 할 일 관리 프로그램을 만듦
    B. 느낀 점: 앞으로 있을 프로젝트에 가장 많은 도움이 될 프로그램 작성이었다.

2_B_i 기능 별 구현 사항

```
case 1:
    printf("할 일을 입력하세요 (공백 없이 입력하세요): ");
    scanf_s("%s", tasks[taskCount], (int)sizeof(tasks[taskCount]));
    printf("할 일 ""%s""가 저장되었습니다₩n₩n", tasks[taskCount]);
    taskCount++;
    break;
```

1.코드블록 스크린샷

2.입력

-tasks: 할 일 목록 저장 2차원 배열

-taskCount: 현재 작업 수

3.결과(블록 종료된 결과)

-tasks배열 taskCount 순서에 할 일 입력됨.

4.설명

-tasks[taskCount]에 할 일을 입력시키고

-taskCount를 1 증가시킴

```
case 2:
    // 할 일 삭제하는 코드 블록
    printf("삭제할 할 일의 번호를 입력해주세요. (1부터 시작):");
    scanf_s("%d", &delIndex);
    if (delIndex > taskCount || delIndex <= 0) {
        printf("삭제 범위가 벗어났습니다.₩n");
    }
    else {
        printf("%d. %s : 할 일을 삭제합니다.₩n", delIndex, tasks[delIndex - 1]);

        // 배열간 대입 (=배열에 문자 배열인 문자열의 대입) 이 불가능하기 때문에
        // 문자열 복사 함수로 삭제
        strcpy_s(tasks[delIndex - 1], sizeof(tasks[delIndex - 1]), "");

        // 특정 인덱스의 할 일 삭제 후 뒤에 있는 할 일 앞으로 옮기기
        for (int i = delIndex; i < taskCount + 1; i++) {
            strcpy_s(tasks[i - 1], sizeof(tasks[i]), tasks[i]);
        }
        taskCount -= 1;
    }
    break;
```

1. 코드블록 스크린샷

2.입력

-delIndex:삭제할 일의 번호

3.결과

-tasks[delIndex-1]의 할 일 삭제

4.설명

-delIndex에 삭제할 일의 번호를 입력받고,

-delIndex가 taskCount보다 크거나 0보다 작거나 같으면 범위가 벗어났음을 출력,

-그게 아니라면 어떤 일을 삭제할지 출력하고,

-문자열 복사 함수로 해당 배열의 문자열을 삭제

-i가 taskCount-1보다 작은 값까지 1늘리며 반복

-문자열 복사 함수로 뒤의 배열을 한 칸씩 앞으로 당김

-마지막에 taskCount를 1씩 줄임

```
case 3:
    printf("할 일 목록₩n");
    for (int i = 0; i < taskCount; i++) {
        printf("%d. %s ₩n", i + 1, tasks[i]);
    }
    printf("₩n");
    break;
```

1.코드블록 스크린샷

2.입력

3.결과

-tasks[i]의 문자열 출력

4.설명

-i가 taskCount보다 작은값까지 1늘리며 반복

-i+1,tasks[i]배열을 출력

```
case 4:
    terminate = 1;
    break;
```

```
if (terminate == 1) {//case4에서 설정한 terminate=1로 적용되어 프로그램break
        printf("종료를 선택하셨습니다. 프로그램을 종료합니다.₩n");
        break;
```

1.코드블록 스크린샷

2.입력

-terminate:프로그램 종료를 위한 값 저장

3.결과

-프로그램 종료

4.설명

-terminate에 1을 입력하고

-프로그램 마지막에 조건문 terminate=1일 때 프로그램 종료되도록 작성하여 프로그램 종료

```
case 5:
    printf("수정할 일의 번호를 입력하세요.(1부터 시작): ");
    scanf_s("%d", &changeIndex);//수정할 일의 번호 입력받음
    printf("수정후의 일을 입력하세요.: ");
    ch = getchar();
    scanf_s("%s", tasks[changeIndex - 1], (int)sizeof(tasks[changeIndex - 1]));/
    printf("%d 번째의 할 일이 수정되었습니다.", changeIndex);
    break;
```

1.코드블록 스크린샷

2.입력

-changeIndex: 수정할 일의 순서 입력

3.결과

-tasks[changeIndex-1]의 할 일 수정

4.설명

-changeIndex에 수정할 배열의 번호 입력

-tasks[changeIndex-1]의 배열에 수정후 일을 입력

```
if (taskCount >= 10) {
    printf("할 일이 다 찼습니다. 프로그램을 종료합니다.₩n");
    break;
```

1.코드블록 스크린샷

2.입력

3.결과

-taskCount가 10이상이면 프로그램 종료

4.설명

-taskCount가 10이상일 시 종료되는 조건문

2_C_i 기능 별 테스트 결과

```
1
할 일을 입력하세요 (공백 없이 입력하세요): study
할 일study가 저장되었습니다
```

할 일 추가하기

```
3
할 일 목록
1. study
```

할 일 목록 보여주기

```
2
삭제할 할 일의 번호를 입력해주세요. (1부터 시작):1
1. study : 할 일을 삭제합니다.
```

할 일 삭제하기

2_C_ii 최종 테스트 스크린샷

```
TODO 리스트 시작!
------------------
메뉴를 입력해주세요.
1. 할 일 추가
2. 할 일 삭제
3. 목록 보기
4. 종료
5. 할 일 수정
현재 할 일 수 = 0
------------------
1
할 일을 입력하세요 (공백 없이 입력하세요): study
할 일study가 저장되었습니다

------------------
메뉴를 입력해주세요.
1. 할 일 추가
2. 할 일 삭제
3. 목록 보기
4. 종료
5. 할 일 수정
현재 할 일 수 = 1
------------------
3
할 일 목록
1. study

------------------
메뉴를 입력해주세요.
1. 할 일 추가
2. 할 일 삭제
3. 목록 보기
4. 종료
5. 할 일 수정
현재 할 일 수 = 1
------------------
2
삭제할 할 일의 번호를 입력해주세요. (1부터 시작):1
1. study : 할 일을 삭제합니다.
------------------
메뉴를 입력해주세요.
1. 할 일 추가
2. 할 일 삭제
3. 목록 보기
4. 종료
5. 할 일 수정
현재 할 일 수 = 0
------------------
4
종료를 선택하셨습니다. 프로그램을 종료합니다.
```