1. 서론

- 프로젝트 목적 및 배경 : 7주차까지 배운 내용에 대한 실습을 위해 진행

- - 목표 : TODO 리스트 만들기

2. 요구사항

- 사용자가 할 일을 입력, 삭제, 출력할 수 있는 프로그램

- 기능 요구사항

1) 기능 요구 사항

① 사용자에게 작업 요청 받기

• 1. 할 일 추가, 2. 할 일 삭제, 3. 목록 보기, 4. 종료, 5. 할 일 수정

② 요청 받은 작업에 따라 아래 기능 수행

① 할 일 추가를 입력했을 경우, 사용자에게 할 일을 입력 받고 저장

② 할 일 삭제를 입력했을 경우, 인덱스를 입력 받고 해당 할 일 삭제

③ 목록 보기를 입력했을 경우, 전체 할 일 목록을 보여주기

④ 종료를 입력했을 경우, 프로그램 종료

⑤ 할 일 수정을 입력했을 경우, 인덱스와 할 일 (문자열)을 입력 받고, 해당 인덱스의 할 일 변경

• 주의: 입력 받는 인덱스에 -1 한 것이 실제 배열의 인덱스가 됨

③ 할 일이 10개로 다 찬 경우는 할 일이 다 찼다고 출력하고 프로그램 종료

2) 제약 조건

• 할 일 목록은 2차원 배열 (10 x 100) = 100개의 문자를 저장할 수 있는 문자열 10개 저장

3. 설계 및 구현

- 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/ 함수 스크린샷 : 할 일 추가하는 코드 블록

2) 입력

taskCount = 현재 작업 수

tasks = 할 일 목록 저장 2차원 배열

3) 결과

입력받은 할 일이 추가된 tasks

4) 설명

사용자에게 할 일을 입력받는다.

할 일을 2차원 배열에 저장한다.

```
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]); //2번째 할일 삭제
        }
        taskCount -= 1; //해야 할 일 하나 삭제했으니까 할일 수 -1
    }
    break;
```

1) 코드블록/함수 스크린샷 : 할 일 삭제하는 코드 블록

2) 입력

taskCount = 현재 작업 수

tasks = 할 일 목록을 저장하는 2차원 배열

delIndex = 할 일 삭제할 때 필요한 index를 저장하는 변수

3) 결과

할 일 목록이 삭제된 tasks

4) 설명

사용자에게 삭제할 일의 번호를 입력 받음

번호-1이 인덱스이기 때문에 배열에서 번호-1의 값을 삭제한다

삭제한 후에 번호-1 뒤의 일들을 앞으로 끌어온다

```
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/함수 스크린샷: 할 일 목록을 출력하는 코드 블록

2) 입력

taskCount = 할 일의 수 저장

tasks = 할 일의 목록을 저장하는 2차원 배열

3) 결과

Tasks 목록을 출력한다.

4) 설명

현재 할 일의 수만큼 반복을 실행한다. 하지만 인덱스는 0부터 시작하여 taskCount-1까지 진행되어야 하기 때문에 0~taskCount-1까지 반복을 실행하여 tasks 목록을 출력한다.

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

1) 코드블록/함수 스크린샷: 프로그램 종료 코드

2) 입력

terminate = 종료를 위한 변수

3) 결과

프로그램을 종료한다.

4) 설명

사용자가 4번을 입력할 경우 변수 terminate가 1이 되면서 프로그램이 종료됨.

```
case 5:
    printf("할 일의 번호를 입력하세요.");
    scanf_s("%d", &todo_number);
    ch = getchar();
    printf("할 일의 내용을 입력하세요.");
    scanf_s("%s", todo_name, sizeof(todo_name));
    strcpy_s(before_todo_name, sizeof(before_todo_name), tasks[todo_number - 1]);
    strcpy_s(tasks[todo_number-1], sizeof(tasks[todo_number-1]), todo_name);
    printf("%d. %s가 %d. %s로 할일이 변경되었습니다.\n", todo_number, before_todo_name, todo_number, todo_name);
    break;
```

1) 코드블록/함수 스크린샷: 할 일 수정하는 코드 블록

2) 입력

todo_number = 수정할 일의 번호를 저장하는 변수

ch = 버퍼를 저장하는 변수

todo_name = 수정할 일의 내용을 저장하는 변수

before_todo_name = 수정하기 전의 일의 내용을 저장하는 변수

3) 결과

할 일이 수정된 tasks

4) 설명

사용자에게 수정할 일의 번호를 입력 받고 변수 todo_number에 저장한다.

함수 getchar를 이용하여 버퍼를 받아 ch에 저장하고 사용자에게 할 일의 내용을 입력받아 todo_name에 저장한다.

함수 strcpy_s를 통해 할 일의 내용을 수정한다. 변수 before_todo_name에 전에 저장되어 있던 할 일의 내용을 저장한다. printf문을 통해 전에 할 일이 바꼈음을 알려줌

4. 테스트

- 기능 별 테스트 결과

1) 할 일 추가하기

```
1
할 일을 입력하세요 (공백 없이 입력하세요): C언어공부하기
할 일 C언어공부하기가 저장되었습니다
```

2) 할 일 삭제하기

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

3) 할 일 목록 보기

```
3
할 일 목록
1. C언어공부하기
2. 뭐쓰지
```

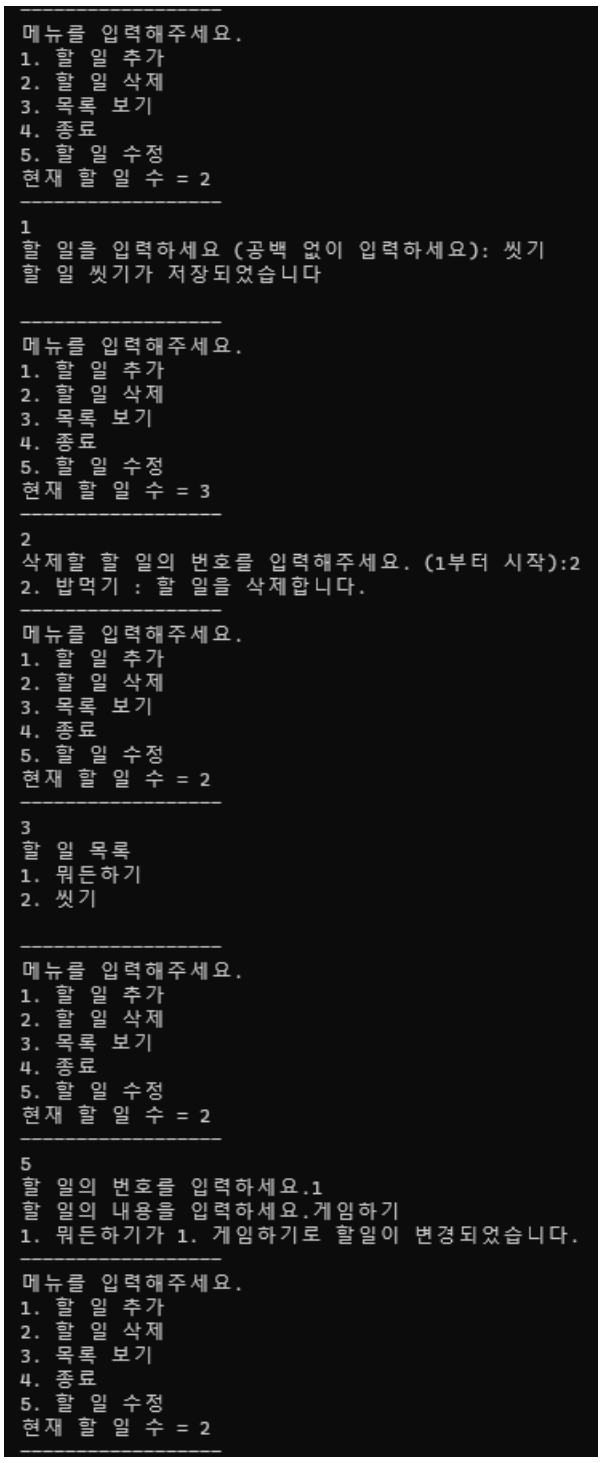
4) 프로그램 종료하기

```
4
종료를 선택하셨습니다. 프로그램을 종료합니다.
```

5) 할 일 수정하기

```
5
할 일의 번호를 입력하세요.2
할 일의 내용을 입력하세요.씻고나가기
2. 밥먹기가 2. 씻고나가기로 할일이 변경되었습니다.
```

- 최종 테스트 스크린샷



5. 결과 및 결론

- 프로젝트 결과: To_do 관리 프로그램을 만들었다.

- 느낀 점: 그래도 알던 내용이라서 할 일 수정하기를 만들 수 있었지만, 한번씩 머리가 하얘지는 것을 느꼈음. 코딩 공부를 열심히 해서 이런 것에서 막히지 않는 실력을 가져야겠다고 생각함.