1. 서론

1. 프로젝트 목적 및 배경: 지금까지 C프로그래밍 수업에서 배웠던 내용들(배열, 내장함수 등등등)을 기반으로 프로그램을 만들어보는 실습을 하기 위해

2. 목표: 할 일 관리 프로그램 만들기

2. 요구사항

1. 사용자 요구사항: 번호 입력, 할 일 입력하기, 할 일 삭제하기, 수정하기와 사용자가 기록했던 할 일을 볼 수 있는 프로그램

2. 기능 요구사항:

1) 사용자에게 작업 요청 받기 (1. 할 일 추가 2. 할 일 삭제 3. 목록 보기 4. 종료 5. 할 일 수정)

2) 요청 받는 작업에 따라 기능 수행

- 할 일 추가를 입력했을 경우에 사용자에게 할 일을 입력받고 저장

- 할 일 삭제를 입력했을 경우에 인덱스를 입력 받고 해당 할 일 삭제

- 목록 보기를 입력했을 경우에 전체 할 일 목록을 보여주기

- 종료를 입력했을 경우에 프로그램 종료

- 할 일 수정을 입력했을 경우는 인덱스와 할 일(문자열)을 입력받고 해당 인덱스 할 일 변경 (주의사항: 입력 받는 인덱스에 -1 한 것이 실제 배열의 인덱스이다.)

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

3. 설계 및 구현

1. 기능 별 구현 사항:

```
printf("----------------₩n");
printf("메뉴를 입력해주세요.₩n");;
printf("1. 할 일 추가₩n2. 할 일 삭제₩n3. 목록 보기₩n4. 종료₩n5. 할 일 수정₩n");
printf("현재 할 일 수 = %d₩n", taskCount);
printf("----------------₩n");
scanf_s("%d", &choice);
```

1) 코드블록 스크린샷(위)

2) 입력

• taskCount = 현재 할 일 수(처음에는 0으로 초기화 되어있음)

• choice = 사용자가 입력한 번호가 저장되는 변수

3) 결과

• 메뉴 출력과 choice에 사용자가 입력한 번호값 저장

4) 설명

실행하면 사용자에게 메뉴가 좌르륵 출력된다. 그리고 입력을 받아서 choice에 값을 저장한다.

```
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++;  // 할 일 개수가 늘어남.
    break;
```

1) 코드블록 스크린샷(위)

2) 입력

• tasks = 할 일 목록 저장 2차원 배열

• taskCount = 현재 할 일 수

3) 결과

• 할 일이 추가된 tasks

• taskCount에 1이 더해짐

4) 설명

사용자에게 추가할 할 일을 입력받는다. 그 문자열을 tasks 배열에 저장한다. 처음에는 인덱스가 0이기 때문에 그 곳에 할 일을 저장한 후, taskCount에 1을 더한다.

```
case 2:
    // 할 일 삭제하는 코드 블록
    printf("삭제할 할 일의 번호를 입력해주세요. (1부터 시작):");
    scanf_s("%d", &delIndex);  // 사용자가 입력한 값을 delIndex에 저장
    if (delIndex > taskCount || delIndex <= 0) {  // 만약 지울 인덱스번호가 할 일
        printf("삭제 범위가 벗어났습니다.₩n");
    }
    else {
        printf("%d. %s : 할 일을 삭제합니다.₩n", delIndex, tasks[delIndex - 1]);
        // 문자열 복사 함수로 삭제(조사한 결과 형식은 배열이름, 배열크기, "새로운
        strcpy_s(tasks[delIndex - 1], sizeof(tasks[delIndex - 1]), "");  // 마찬가
        // 입력한 번호의 할 일 삭제 후 뒤에 있는 할 일을 앞으로 옮김.
        for (int i = delIndex; i < taskCount + 1; i++) {
            strcpy_s(tasks[i - 1], sizeof(tasks[i]), tasks[i]);  // 문자열을 정의
        }
        taskCount -= 1;  // 할 일 개수 빼줌
    }
    break;
```

1) 코드블록 스크린샷(위)

2) 입력

• delIndex = 할 일 삭제 인덱스 번호 저장 변수

• tasks = 할 일 목록 저장 2차원 배열

• taskCount = 현재 할 일 수

3) 결과

• 할 일이 삭제된 tasks

• taskCount에 1이 빼짐

4) 설명

사용자에게 삭제할 할 일의 번호(인덱스)를 입력받고 if문으로 범위를 체크한다. 만약 지울 인덱스 번호가 할 일 수보다 많거나 0이랑 같거나 작으면(즉, 벗어난 경우)에는 범위가 벗어났다고 출력한다. 그게 아니라면 입력 받은 인덱스-1에 있는 할 일을 배열에서 제거한다. 그 후, 뒤에 있던 할 일들을 앞으로 옮긴다. 그리고 taskCount에서 1을 뺀다.

```
case 3:  // 목록 보여주기
    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(위)

2) 입력

• tasks = 할 일 목록 저장 2차원 배열

• taskCount = 현재 할 일 수

3) 결과

• 할 일 목록 출력됨

4) 설명

for 반복문을 이용해 배열에 저장되어 있는 할 일의 목록이 출력된다. i<taskCount 일때까지 반복된다. 출력할 때는 i+1을 해서 0. ~ 이 아닌 1. ~ 이런 형식으로 출력될 수 있게 하였다.

```
case 4:  // 프로그램 종료
    terminate = 1;
    break;
```

```
if (terminate == 1) {  // case4에서 terminate가 1이 되는데 이때 종료하게 만드는 if문이다.
    printf("종료를 선택하셨습니다. 프로그램을 종료합니다.₩n");
    break;
```

1) 코드블록 스크린샷(위)

2) 입력

• terminate = 프로그램 종료를 위한 변수

3) 결과

• 프로그램이 종료됨

4) 설명

terminate가 1이면 저기 if문에 의해 프로그램이 종료된다.

```
case 5:
    printf("수정할 할 일의 번호를 입력하세요.₩n");
    scanf_s("%d", &changeIndex);  // 수정할 일 번호 입력받기
    if (changeIndex > taskCount || changeIndex <= 0) {  // changeIndex가 taskCou
        printf("수정 범위가 벗어났습니다.₩n");
        break;
    }
    else {
        strcpy_s(tasks[changeIndex - 1], sizeof(tasks[changeIndex - 1]), "");  /


    }
    // 그 자리에 수정할 일 입력받기
    printf("할 일을 입력하세요.₩n");
    ch = getchar();  // 버퍼 제거
    scanf_s("%s", tasks[changeIndex -1], (int)sizeof(tasks[changeIndex -1]));
    break;
```

1) 코드블록 스크린샷(위)

2) 입력

• changeIndex = 할 일 수정 인덱스 번호 저장 변수

• tasks = 할 일 목록 저장 2차원 배열

• taskCount = 현재 할 일 수

3) 결과

• 할 일이 수정된 tasks

4) 설명

사용자로부터 수정할 일의 번호를 입력받아서 changeIndex에 저장한다. if문으로 수정범위를 체크하고 changeIndex-1에 있는 할 일을 배열에서 제거한다. 그 후 수정할 할 일을 입력받아서 그 자리(제거된 자리)에 할 일을 추가한다. 입력 받는 인덱스에 -1 한 것이 실제 배열의 인덱스가 된다.

왜냐하면 할 일 숫자는 1부터 시작인데 실제 배열은 0부터 인덱스가 시작했기 때문이다!

```
if (taskCount == 10) {  // 할 일 개수가 10이 되면 프로그램을 종료하게 만드는 if문
    printf("할 일이 10개로 다 찼습니다. 프로그램을 종료합니다.₩n");
    break;
```

1) 코드블록 스크린샷(위)

2) 입력

• taskCount = 현재 할 일 수

3) 결과

• 프로그램 종료

4) 설명

if문을 이용해서 taskCount가 10이 되면 다찼다는 문구가 출력되면서 프로그램이 종료된다.

4. 테스트

1. 기능 별 테스트 결과:

① 사용자에게 작업 요청 받기

```
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
------------------
```

② 할 일 추가하기

```
1
할 일을 입력하세요 (공백 없이 입력하세요): 게임하기
할 일 게임하기가 저장되었습니다
```

③ 할 일 목록 보여주기

```
3
할 일 목록
1. 게임하기
```

④ 할 일 수정하기

```
5
수정할 할 일의 번호를 입력하세요.
1
할 일을 입력하세요.
공부하기
```

⑤ 할 일 삭제하기

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

⑥ 프로그램 종료하기

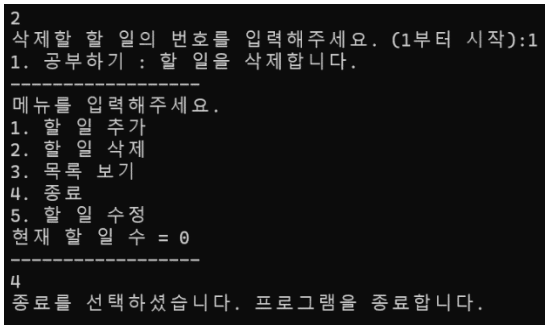
```
4
종료를 선택하셨습니다. 프로그램을 종료합니다.
```

2. 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): 게임하기
할 일 게임하기가 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. 게임하기

------------------
메뉴를 입력해주세요.
1. 할 일 추가
2. 할 일 삭제
3. 목록 보기
4. 종료
5. 할 일 수정
현재 할 일 수 = 1
------------------
5
수정할 할 일의 번호를 입력하세요.
1
할 일을 입력하세요.
공부하기
------------------
메뉴를 입력해주세요.
1. 할 일 추가
2. 할 일 삭제
3. 목록 보기
4. 종료
5. 할 일 수정
현재 할 일 수 = 1
------------------
```



5. 결과 및 결론

1. 프로젝트 결과: 할 일을 관리하는 프로그램을 만들어서 사용자가 편리하게 할 일을 추가하고 삭제하고 수정할 수 있고 한눈에 할 일들을 확인할 수 있게 하였다.

2. 느낀 점: 다음주부터 시작할 자신이 처음부터 계획해서 직접 프로그램을 만드는 기말 프로젝트를 하기 전에 진행한 실습이었는데 아직까지는 C언어에 익숙하지 않아서 코드를 짜는 것이 어려운 것 같다. 그래도 이렇게 한번 보고서랑 프로그램을 완성시켜봤으니까 방식은 잘 알 것 같다. 나의 프로젝트를 성공적으로 해내기 위해 열심히 노력해야겠다는 생각이 든다 :)