1. 서론

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

2. 목표 : TODO 리스트 만들기

2. 요구사항

1. 사용자 요구사항 : 사용자가 할 일을 입력, 삭제, 출력, 수정할 수 있는 프로그램

2. 기능 요구사항 : 사용자에게 작업 요청 받기, 요청 받은 작업에 따라 기능 수행, 할 일이 10개로 다 찬 경우는 할 일이 다 찼다고 출력하고 프로그램 종료

3. 설계 및 구현

1. 기능 별 구현 사항 :

```
case 1:  // 1. 할 일 추가
    printf("할 일을 입력하세요 (공백 없이 입력하세요): ");
    scanf_s("%s", tasks[taskCount], (int)sizeof(tasks[taskCount]));  // 사용자로부터 할 일을 입력받기
    printf("할 일 ""%s""가 저장되었습니다₩n₩n", tasks[taskCount]);
    taskCount++;  // 할 일의 수 1 증가
    break;
```

(1) 입력 : taskCount = 현재 작업 수

tasks = 할 일 목록 저장 2차원 배열

(2) 결과 : 할 일이 추가된 tasks

(3) 설명 :

- 사용자에게 추가할 할 일을 입력받는다
- 입력 받은 내용을 tasks[taskCount] 에 저장한다.
- taskCount 값을 1 증가시킨다.

```
case 2:  // 2. 할 일 삭제
    printf("삭제할 할 일의 번호를 입력해주세요. (1부터 시작):");
    scanf_s("%d", &delIndex);  // 사용자로부터 삭제할 할 일의 번호 입력받기
    if (delIndex > taskCount || delIndex <= 0) {
        printf("삭제 범위가 벗어났습니다.₩n");
    }  // 삭제 할 번호가 저장되어있는 할 일의 수보다 많거나 0,음수 일 경우 오류 메세지 출력
    else {
        printf("%d. %s : 할 일을 삭제합니다.₩n", delIndex, tasks[delIndex - 1]);
        strcpy_s(tasks[delIndex - 1], sizeof(tasks[delIndex - 1]), "");  // 삭제 한 할 일에 ""문자열 복사
        for (int i = delIndex; i < taskCount + 1; i++) {
            strcpy_s(tasks[i - 1], sizeof(tasks[i]), tasks[i]);  // 삭제 후 뒤에 있는 할 일 앞으로 옮기기
        }
        taskCount -= 1;  // 할 일의 수 1 감소
    }
    break;
```

(1) 입력 : taskCount = 현재 작업 수

tasks = 할 일 목록 저장 2차원 배열

delIndex = 삭제할 할 일의 인덱스 번호

(2) 결과 : 할 일이 삭제된 tasks

(3) 설명 :

- 사용자에게 삭제할 할 일의 인덱스를 입력받는다.
- 입력 받은 인덱스 -1 에 있는 할 일을 배열에서 제거한다.
- 제거한 후 뒤에 있던 할 일들을 앞으로 당겨온다.

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

(1) 입력: taskCount = 현재 작업 수

tasks = 할 일 목록 저장 2차원 배열

(2) 결과 : 할 일 목록을 나열

(3) 설명 :

- 인덱스 0번부터 taskCount 까지 저장 되어있는 할 일을 출력한다.

```
case 5:  // 5. 할 일 수정
    printf("수정할 할 일의 번호를 입력해주세요. (1부터 시작): ");
    scanf_s("%d", &changeIndex);  // 사용자로부터 수정 할 할 일의 번호 입력받기
    if (changeIndex > taskCount || changeIndex <= 0) {
        printf("수정 범위가 벗어났습니다.₩n");
    }  // 삭제 할 번호가 저장되어있는 할 일의 수보다 많거나 0, 음수 일 경우 오류 메세지 출력
    else {
        printf("%d. %s : 할 일을 수정합니다.₩n", changeIndex, tasks[changeIndex - 1]);
        printf("수정할 할 일의 내용을 입력해주세요.₩n");
        scanf_s("%s", tasks[changeIndex - 1], (int)sizeof(tasks[changeIndex-1]));  // 사용자로부터 수정할 할 일의 내용 입력받아 내용 수정하기
        printf("%d. %s : 새롭게 수정되었습니다. ₩n", changeIndex, tasks[changeIndex - 1]);
    }
    break;
```

(1) 입력 : taskCount = 현재 작업 수

Tasks = 할 일 목록 저장 2차원 배열

changeIndex = 수정할 할 일의 인덱스 번호

(2) 결과 : 할 일이 수정된 tasks

(3) 설명:

- 사용자에게 수정할 할 일의 인덱스를 입력 받는다.
- 사용자에게 수정할 할 일의 내용을 입력 받는다.
- 입력 받은 인덱스 -1 에 있는 할 일을 입력 받은 내용으로 수정한다.

4. 테스트

1. 기능 별 테스트 결과 :

(1) 할 일 추가하기

```
1
할 일을 입력하세요 (공백 없이 입력하세요): 공부
할 일 공부가 저장되었습니다
```

(2) 할 일 삭제하기

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

(3)할 일 목록 나열하기

```
3
할 일 목록
1. 알바
2. 시험
```

(4) 할 일 수정하기

```
5
수정할 할 일의 번호를 입력해주세요. (1부터 시작): 1
1. 보고서쓰기 : 할 일을 수정합니다.
수정할 할 일의 내용을 입력해주세요.
밥먹기
1. 밥먹기 : 새롭게 수정되었습니다.
```

2. 최종 테스트 스크린샷 :

```
할 일 목록
1. 알바
2. 시험

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

5. 결과 및 결론

1. 프로젝트 결과 : 투두 관리 프로그램을 만들었다.

2. 느낀 점 : 투두 프로그램의 코딩을 해석하며 프로그램 실행의 원리와 과정을 이해하였고 프로그램을 만들 때 구조화 시켜 그에 맞게 작업이 들어가야 한다는 것을 알았다. 또한, 수정하는 기능을 스스로 만들며 scanf_s 함수가 내용을 그대로 덧붙일 수 있다는 사실을 새로 알았고 다른 기능들을 함수화 시키며 그의 편리함을 깨달았다.