

Todo_manager 프로젝트

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

목표: TODO 리스트 만들기

요구사항

사용자 요구사항:

출력 받은 문을 읽고 해당 12345 중 원하는 것을 선택

1을 선택 시 추가할 할 일을 씀

2를 선택 시 삭제하고 싶은 index입력

3을 선택 시 현재 목록이 나옴

4를 선택 시 종료된다.

5를 선택 시 수정하고 싶은 인덱스를 적고 할 일을 수정한다.

주의: 입력 받는 인덱스에 -1 한 것이 실제 배열의 인덱스가 됨

기능 요구사항

1. 할 일 추가 기능:
2. 할 일 삭제 기능:
3. 현재 할 일 목록 출력 기능:
4. 종료 기능
5. 할 일 수정 기능:

제약조건: 할 일 목록은 2차원 배열 (10×100) = 100개의 문자를 저장할 수 있는 문자열 10개 저장

설계 및 구현

```
#include <stdio.h>
#define MAX_TASKS 10
#define CHAR_NUM 100
#include <string.h>

int main() {
    char tasks[MAX_TASKS][CHAR_NUM] = { "" }; // 할 일 목록을 저장하기 위한 2차원 배열
    int taskCount = 0; // 할 일의 수를 저장하기 위한 변수
```

입력: tasks[MAX_TASKS][CHAR_NUM]은 할 일 목록을 저장하기 위한 2차원 배열이다.

TASK는 10으로 제한을 두고 CHAR_NUM은 100자리까지 문자열을 제한을 둔다.

taskCount는 할 일의 수를 저장하기 위한 변수이다.

```
printf("TODO 리스트 시작! \n");

while (1) {
    int choice = -1; // 사용자 입력 메뉴를 저장하기 위한 변수

    // 사용자에게 메뉴를 출력하고, 메뉴를 입력받기
    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);

    int terminate = 0; // 종료를 위한 flag
    int delIndex = -1; // 할 일 삭제를 위한 index 저장 변수
    int changeIndex = -1; // 할 일 수정을 위한 index 저장 변수
    char ch; // 할 일 수정시 버퍼를 받기 위한 문자 변수
```

while문을 통해 루프를 반복적으로 실행하며 사용자와 상호작용한다.

입력:

choice: 사용자 입력 메뉴를 저장하기 위한 변수이다.

Terminate: 종료에 관한 변수이다.

delIndex: 할 일 삭제를 위한 인덱스 변수이다.

modifyIndex: 할 일 수정을 위한 인덱스 변수이다.

Ch: 할 일 수정 시 버퍼를 받기 위한 문자 변수이다.

결과: 사용자에게 메뉴를 출력하고 입력받는다.

1,2,3,4,5 중 하나를 입력 받아 choice에 저장된다

```
// 입력에 따른 기능 수행
switch (choice) {
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;
```

1번 선택 시 사용자의 할 일을 입력 받아 저장한다.

입력:

Task[]: 사용자가 입력한 할 일을 저장한다.

taskCount: 사용자가 입력한 할 일의 수를 저장하는 변수

결과: Printf를 활용해 위 문장이 출력된다.

설명: %s는 문자열 서식지정자이다. Sizeof는 수에 제한을 두는 것이다.

```

    break;
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;

```

입력: delindex: 특정할 일을 삭제할 위한 변수

결과:

2번 선택 시 현재 할 일을 목록에 삭제하기 위한 기능을 한다.

만약 입력 받은 삭제 범위가 벗어나면 삭제 범위가 벗어났다고 뜨며 switch문을 나간 후 다시 메뉴가 출력된다.

for문을 통해 특정 인덱스의 할일 삭제 후 뒤에 있는 할 일을 앞으로 옮긴다.

예시) 할일 1을 삭제하고 싶다고 입력받을 때 배열 task[i-1]로 자리가 옮겨지고 task[0]이 된다.

```

    break;
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;

```

3번을 선택시 현재 할일 목록을 출력된다.

결과 : 예시) "1. 밥먹기" 사용자에게 입력받은 값이 출력된다.

```
case 4:
    terminate = 1;
    break;
default:
    printf("잘못된 선택입니다. 다시 선택하세요.\n");
}

if (terminate == 1) {
    printf("종료를 선택하셨습니다. 프로그램을 종료합니다.\n");
    break;
}
```

입력: terminate: 종료를 위한 변수

4를 선택 시 terminate의 값이 1로 초기화되고

만약 이외의 값이 입력되면 다시 선택하세요가 뜨면서 메뉴가 다시 출력된다.

4를 선택하면 switch문을 나가서 terminate가 1이나와 종료가 출력되고 실행이 종료된다.

```
// TODO: 할 일이 다 찼는지 체크하는 코드 작성
if (taskCount == 10) {
    printf("할 일이 %d개로 다 찼습니다.", taskCount);
    break; // taskCount가 10이면 위문장 출력후 멈춤
}
```

그 외 taskCount가 10이면 할 일이 다 차서 멈추고 종료된다.

테스트

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

처음에는 할일 메뉴를 보여준다.

6

잘못된 선택입니다. 다시 선택하세요.

메뉴를 입력해주세요.

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

현재 할 일 수 = 0

|

1,2,3,4,5 메뉴 이외의 값을 입력하면 에러메시지 뜨며 다시 메뉴를 보여주며 메뉴 1,2,3,4,5 중에 하나를 입력한다.

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): 집가기
할 일 집가기가 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

1을 입력하면 할일을 입력하세요가 뜨고 만약 집가기를 입력하면 할일 "집가기"가 저장되었다고 뜬다.

그리고 다시 메뉴를 출력해준다.

```

-----
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. 집가기 : 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
-----

```

만약 2를 누르면 삭제할 할 일의 번호를 입력하고 그에 해당하는 할 일이 삭제된다고 뜬다. 그리고 다시 메뉴를 입력 받을 수 있다.

```

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

3
할 일 목록
1. 밥먹기
2. 공부
3. 세수

```

3을 입력하면 입력했던 할 일들을 1부터 다 볼 수 있다.

메뉴를 입력해주세요.

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

현재 할 일 수 = 3

5

수정할 할 일의 번호를 입력해주세요. (1부터 시작): 2
새로운 할 일을 입력해주세요 마라탕먹기
새로운 할 일이 추가되었습니다: 2. 마라탕먹기

만약 5를 누르면 그동안 입력했던 인덱스 중에서 수정하고 싶은 할 일을 선택하여 입력한다음

새로운 할 일을 추가하고, 그에 따라 수정한 할 일이 출력된다.

할 일이 10개로 다 차면 10개로 다 찼다고 뜨고 종료된다.

1

할 일을 입력하세요 (공백 없이 입력하세요): ㅇ
할 일 ㅇ가 저장되었습니다

메뉴를 입력해주세요.

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

현재 할 일 수 = 9

1

할 일을 입력하세요 (공백 없이 입력하세요): ≡
할 일 ≡가 저장되었습니다

할 일이 10개로 다 찼습니다.

```
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
```

```
-----
```

```
4
```

```
종료를 선택하셨습니다. 프로그램을 종료합니다.
```

4를 누르면 이 프로그램이 종료된다.

결과 및 결론

결과: 메인 함수에 for문 switch문을 활용해서

입력을 받고 출력시키는 간단한 To do list를 만들었다.

결론: main함수에 포함되는 코드 각각을 함수로 코드화 할 수 있다는 게 신기했다.