

C programming project _ To do List

인공지능학부

233131

홍소현

1. 서론

프로젝트 목적은 7주차까지 배운 내용에 대한 실습을 통해 학습한 개념과 기술을 실제로 적용하고 확장하는 것입니다. 이 프로젝트를 통해 학습한 이론을 실제상황에서는 어떻게 활용할 수 있을지를 깨닫고자 진행하였습니다. 다차원 배열을 활용하여 할 일 추가,삭제,수정 기능을 포함한 "TODO 리스트 만들기"를 목표로 하였습니다.

2. 요구사항

(1) 사용자 요구사항

할 일을 입력, 삭제, 출력 및 수정할 수 있는 프로그램을 만드는 것입니다.

(2) 기능 요구사항

사용자에게 먼저 To Do List 의 전체 기능을 보여주는 화면을 작성합니다. 할 일 추가, 삭제, 출력, 프로그램 종료, 할 일 수정의 총 5가지 기능을 지닙니다. 추가로, 사용자가 입력 범위에서 벗어났을 경우 프로그램에 에러가 나지 않고 다시 입력할 수 있도록 코딩합니다.

3. 설계 및 구현

- 기능 별 구현 사항

(1) 기본 콘솔

```
int main() {
    int taskCount = 0; // 할 일 수 저장 변수
    char tasks[MAX_TASKS][CHAR_NUM] = { "" }; // 할 일 목록 저장 2차원 배열

    printf("TODO 리스트 시작! \n");

    // while 함수를 사용하여 반복적인 기능을 하는 프로그램 작성
    while (1) {
        int choice = -1; // 사용자 입력 메뉴를 저장하기 위한 변수

        // 메뉴 출력 및 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);

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

[변수 설정]

int terminate : 종료를 위한 flag

int delIndex : 할 일 삭제를 위한 index 저장 변수

int changeIndex : 할 일 수정을 위한 index 저장 변수

char ch : 할 일 수정 시 버퍼를 받기 위한 문자 변수

[설명]

사용자에게 메뉴 목록 보여주고 난 후, 사용자가 원하는 목록 번호 입력 시 아래 기능들로 넘어갑니다. 잘못된 번호 입력 시 잘못된 번호 입력이라고 다시 원하는 기능의 번호 입력받도록 기본값을 설정합니다.

(2) 할 일 추가 기능

```
// 1~5번 기능에 대한 각 case 작성
switch (choice) {
// 1번. 할 일 추가 기능 작성
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 일이 10개가 되면 프로그램 종료하도록 코드 작성
    if (taskCount == 10) {
        printf("할 일이 다 찹습니다.\n<현재 할 일 수: %d>\n", taskCount);
        terminate = 1;
    }
    break;
```

[입력]

taskCount = 현재 할 일의 개수

[결과]

할 일을 list에 추가하고, 할 일의 개수를 입력합니다.

[설명]

사용자가 1번 기능을 선택했을 시 '할 일을 입력하세요'라는 문구가 나오면 사용자가 할 일을 입력할 수 있도록 합니다. 사용자가 주어진 조건인 공백 없이 할 일을 입력했다면 list에 할 일을 추가합니다. 추가로, taskCount가 10이 되어 할 일의 개수가 다 찼다면 '할 일이 다 찹습니다. <현재 할 일 수>' 문구를 출력하며 프로그램을 종료합니다.

(3) 할 일 삭제 기능

```
break;
// 2번. 할 일 삭제 기능 작성
case 2:
    printf("삭제할 할 일의 번호를 입력해주세요. (1부터 시작):");
    scanf_s("%d", &delIndex);
    if (delIndex > taskCount || delIndex <= 0) {
        printf("삭제 범위를 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 = 삭제할 할 일의 번호

[결과]

할 일을 tasks 배열에서 삭제 후 삭제 번호 할 일 이후의 할 일들을 앞으로 가져옵니다.

[설명]

사용자가 2번 기능을 선택했을 시 '삭제할 할 일의 번호를 입력해주세요'라는 문구가 나오면 사용자가 삭제를 원하는 할 일의 번호를 입력합니다. 사용자가 주어진 조건인 1부터 시작하는 번호를 입력했을 시 성공적으로 할 일의 번호가 tasks 배열에서 삭제되고, 삭제된 번호 이후에 있던 할 일 목록들이 앞으로 당겨져 옵니다.

(4) 할 일 목록 출력 기능

```
// 3번. 할 일 목록 출력 기능 작성
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;
```

[결과]

사용자가 추가, 삭제, 수정한 할 일들의 목록을 출력하여 사용자에게 보여줍니다.

[설명]

사용자가 3번 기능을 선택했을 시 '할 일 목록'이라는 문구와 함께 리스트화 된 할 일 목록들이 1번부터 보여집니다.

(5) 프로그램 종료 기능

```
// 4번, 프로그램 종료 기능 작성
case 4:
    terminate = 1;
    break;
```

[결과]

프로그램을 종료합니다.

[설명]

사용자가 4번 기능을 선택했을 시 '프로그램을 종료합니다'라는 문구와 함께 프로그램이 종료됩니다.

(6) 할 일 목록 수정 기능

```
// 5번, 할 일 목록 수정 기능 작성
case 5:
    printf("수정할 할 일의 번호를 입력해주세요(1~) : ");
    scanf_s("%d", &changeIndex);
    ch = getchar();
    printf("새로운 할 일을 입력해주세요 : ");
    scanf_s("%s", tasks[changeIndex - 1], (int)sizeof(tasks[changeIndex - 1]));
    printf("새로운 할 일이 추가되었습니다 : %d, %s\n", changeIndex, tasks[changeIndex - 1]);
    break;
```

[입력]

changeIndex = 수정할 할 일의 번호

[결과]

Task 배열에서 사용자가 수정을 원하는 번호의 index값을 꺼내와 새로운 값을 입력하도록 요구합니다. 사용자가 새로운 할 일을 입력하면 배열의 해당 index에 할 일을 대체시킵니다.

[설명]

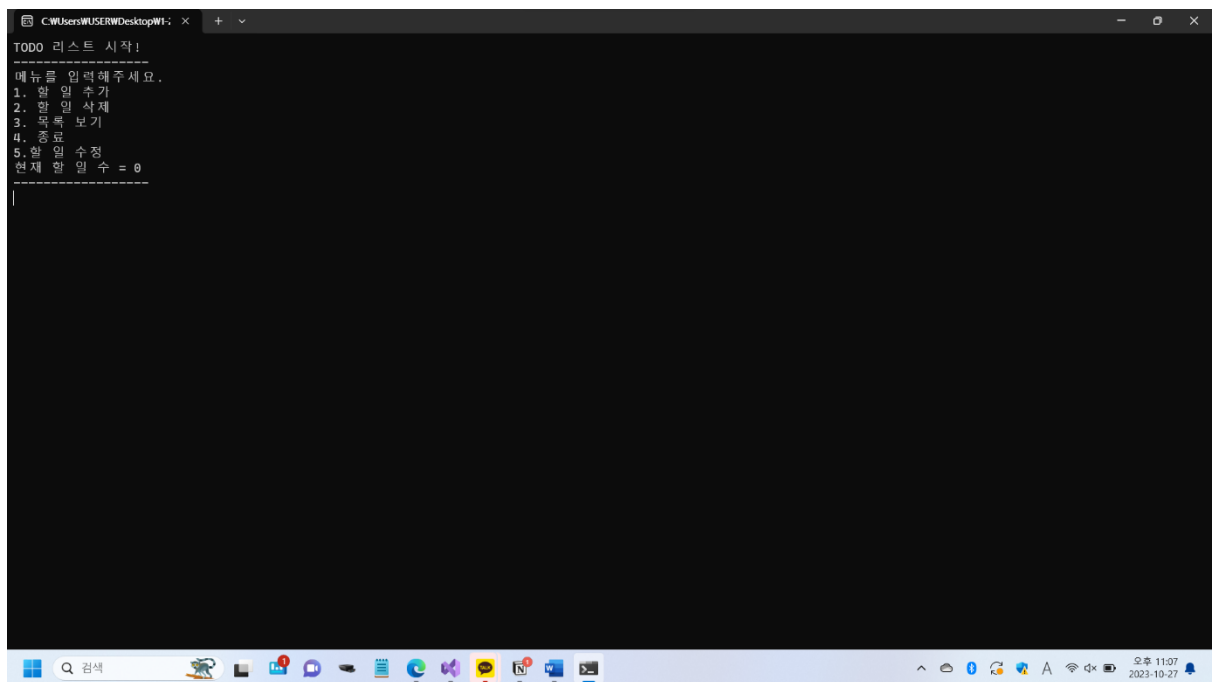
사용자가 5번 기능을 선택했을 시 수정할 할 일의 번호를 입력받은 후 새로운 할 일을 입력

받습니다. 사용자가 새로운 할 일을 입력하면 새로운 할 일을 꺼내온 배열의 자리에 대치시킵니다.

4. 테스트

(1) 기능 별 테스트 결과

- 기본 콘솔



```
C:\Users\WUSER\Desktop\WT-1>
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
```

- 할 일 추가 기능

```
1
할 일을 입력하세요 (공백 없이 입력하세요): 잠자기
할 일 잠자기가 저장되었습니다
```

```
할 일이 다 찼습니다.
<현재 할 일 수: 10>
프로그램을 종료합니다.
```

- 할 일 삭제 기능

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

- 할 일 목록 출력 기능

```
3
할 일 목록
1. 잠자기
2. 집가기
3. 밥먹기
4. 공부그만하기
5. 여행가기
6. 여행계획세우기
7. 씻기
8. 영화보기
9. 드라마재방보기
```

- 프로그램 종료 기능

```
4
프로그램을 종료합니다.
```

- 할 일 목록 수정 기능

```
5
수정할 할 일의 번호를 입력해주세요(1~) : 4
새로운 할 일을 입력해주세요 : 공부하기
새로운 할 일이 추가되었습니다 : 4. 공부하기
-----
메뉴를 입력해주세요.
1. 할 일 추가
2. 할 일 삭제
3. 목록 보기
4. 종료
5. 할 일 수정
현재 할 일 수 = 9
-----
3
할 일 목록
1. 잠자기
2. 집가기
3. 밥먹기
4. 공부하기
5. 여행가기
6. 여행계획세우기
7. 씻기
8. 영화보기
9. 드라마재방보기
```

(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): 공부하기
할 일 공부하기가 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

2

삭제할 할 일의 번호를 입력해주세요. (1부터 시작):1

1. 공부하기 : 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

프로그램을 종료합니다.

```

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): 공부하기
할 일 공부하기가 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

5
수정할 할 일의 번호를 입력해주세요(1~) : 공부그만하기
새로운 할 일을 입력해주세요 : 새로운 할 일이 추가되었습니다 : -1. 編續城曼廠?

-----
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. 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

4
프로그램을 종료합니다.

```

5. 결과 및 결론

프로젝트 결과 주어진 To Do List 기능을 완벽히 수행해내는 결과를 얻었습니다. To Do List 기능에는 할 일 추가, 삭제, 수정, 할 일 목록 수정 및 프로그램 종료의 기능이 있었습니다. 처음에는 이론과 간단한 개념만으로 간단한 실습을 진행 했었기에 과연 지금까지 배운 개념들을 복합적으로 사용하는 프로젝트를 진행할 수 있을까에 대한 의문이 들었습니다. 그러나 기본적인 코드들은 주어졌고, 일부 코드만 작성하고 함수화하면 되는 수준이었기에 주어진 코드들을 차근차근 읽어 내려가며 이해하고 부족한 부분을 채울 수 있었습니다. 이 후 올라온 프로젝트 정답을 비교하며 수정한 부분도 있고, 제 코드를 유지한 부분도 있습니다. 제 코드가 맞다는, 간결하다는 100%확신은 없지만 프로그램이 잘 작동한다는 것에 의의를 두고 앞으로 C언어를 더 배워나가면서 코드를 간략화하고 효율성있게 짜는 방법에 대해 더 공부해보고자하는 목표를 가지고 있습니다.