

タスク管理

VERC

C taskkannri.c ×

```
C taskkannri.c > ...
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <string.h>
4
5  #define MAX_TASKS 100
6  #define TASK_NAME_LENGTH 100
7  #define INPUT_BUFFER_SIZE 256
8
9  // タスク情報を格納する構造体
10 typedef struct {
11     char name[TASK_NAME_LENGTH];
12     int priority; // 1:高, 2:中, 3:低
13     char is_completed; // 'Y':完了, 'N':未完了
14 } Task;
15
16 Task tasks[MAX_TASKS];
17 int task_count = 0;
18
19 // --- 安定した入力処理のためのヘルパー関数 ---
20
21 // 1. 1行全体を安全に読み込み関数 (バッファリアも行う)
22 char* read_line(char *buffer, int size) {
23     char *result = fgets(buffer, size, stdin);
24     char *newline_pos;
25
26     if (result == NULL) return NULL;
27
28     // 改行文字 '\n' を探す
29     newline_pos = strchr(buffer, '\n');
30     if (newline_pos) {
31         *newline_pos = '\0'; // 見つかったらNULL終端に置き換える
32     } else {
33         // バッファ長を超えた入力を読み捨てる (タイムアウト防止の鍵)
34         int c;
35         while ((c = getchar()) != '\n' && c != EOF);
36     }
37     return buffer;
38 }
39
40 // 2. read_lineを使ってメニュー選択の入力を受け取り、数値に変換する関数
41 int get_menu_choice() {
42     char buffer[INPUT_BUFFER_SIZE];
43
44     if (read_line(buffer, sizeof(buffer)) == NULL) {
45         return -1; // 終了シグナル
46     }
47
48     // 文字列を整数に変換して返す
```

```
        return atoi(buffer);
    }
    // --- メインロジック ---
}
void add_task() {
    char buffer[INPUT_BUFFER_SIZE];
    int priority_choice;
    if (task_count >= MAX_TASKS) {
        printf("登録件数が上限に達しました。\\n");
        return;
    }
    // 1. タスク名 (文字列) の入力 - fgetsで安全
    printf("タスク名を入力してください: ");
    if (read_line(tasks[task_count].name, TASK_NAME_LENGTH) == NULL) return;
    if (tasks[task_count].name[0] == '\\0') {
        printf("タスク名が入力されなかったため、登録を中止します。\\n");
        return;
    }
    // 2. 優先度 (数値) の入力 - fgetsで読み込み、atoiで変換
    printf("優先度を入力してください (1:高, 2:中, 3:低): ");
    if (read_line(buffer, sizeof(buffer)) == NULL) return;
    priority_choice = atoi(buffer);
    if (priority_choice < 1 || priority_choice > 3) {
        printf("優先度が無効です。デフォルトで「中」に設定します。\\n");
        priority_choice = 2;
    }
    // 構造体に格納
    tasks[task_count].priority = priority_choice;
    tasks[task_count].is_completed = 'N'; // 初期値は未完了
    task_count++;
    printf("タスクを登録しました。\\n");
}
void display_tasks() {
    if (task_count == 0) {
        printf("登録されているタスクはありません。\\n");
        return;
    }
    printf("\\n--- タスク一覧 ---\\n");
    printf("| No | 優先度 | 状態 | タスク名\\n");
```

```
printf("-----\\n");
for (int i = 0; i < task_count; i++) {
    char *priority_str;
    switch(tasks[i].priority) {
        case 1: priority_str = "高"; break;
        case 2: priority_str = "中"; break;
        case 3: priority_str = "低"; break;
        default: priority_str = "?"; break;
    }
    printf("| %2d | %-4s | %-4s | %s\\n",
        i + 1,
        priority_str,
        tasks[i].is_completed == 'Y' ? "完了" : "未了",
        tasks[i].name);
}
printf("-----\\n");
}
int main() {
    int choice;
    printf("---- 簡易タスク管理プログラム ----\\n");
    while (1) {
        printf("\\n--- メニュー ---\\n");
        printf("1. タスクの新規登録\\n");
        printf("2. タスク一覧表示\\n");
        printf("3. 終了\\n");
        printf("-----\\n");
        printf("選択肢を入力してください: ");
        // ★★★ すべての入力を安全な get_menu_choice() で取得 ★★★
        choice = get_menu_choice();
        if (choice == -1) {
            printf("\\n\\n入力の終端またはエラーによりプログラムを終了します。\\n");
            break;
        }
        switch (choice) {
            case 1:
                add_task();
                break;
            case 2:
```

タスク管理プログラム verC

タスクを管理できるアプリです。

タスク名と優先度を入力する事が必要になります。きちっと管理できる代物になります。

登録されているかどうか確認できます。

登録されたタスクが空っぽになれば完遂出来た照明にもなります。

```
        display_tasks();  
        break;  
    case 3:  
        printf("プログラムを終了します。\\n");  
        return 0;  
  
    default:  
        printf("無効な選択です。1から3の番号を入力してください。\\n");  
        break;  
    }  
}  
return 0;  
}
```