



EDDI

Electronic Design
Development Institute

에디로봇아카데미

임베디드 마스터 Lv1 과정

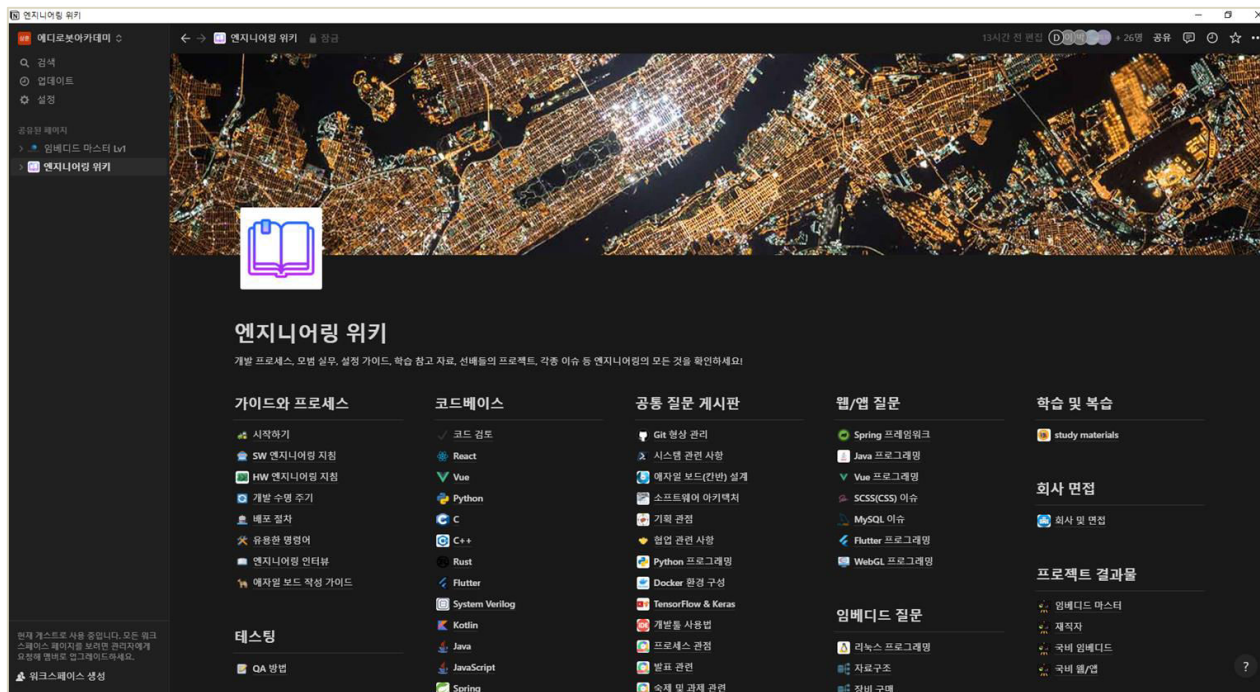
제 4기

2022. 12. 09

진동민

참고

- 질문은 날짜 순서대로 기입
- 노선은 질문&답변_2 부터 웹/앱과 임베디드가 통합된 워크스페이스 '에디로봇아카데미'에서 확인할 수 있다.



- 1) (복습) 매크로 함수에 데이터 타입이 없는 이유는 무엇인가? (C 프로그래밍, 2022/11/19)
- 2) (참고) 과제 답안에 존재하는 오류 발견 (C 프로그래밍, 2022/11/20)
- 3) 함수 정의 파일(.c)에 함수 선언을 작성한 헤더파일(.h)을 include 하는 이유 (C 프로그래밍, 2022/11/20)
- 4) (복습) 헤더파일과 소스파일의 관계성 질문 (C 프로그래밍, 2022/11/22)
- 5) 파이썬 리스트 유사하게 만들기 과제에 관한 질문과 오류 (C 프로그래밍, 2022/11/25)
- 6) 클래스의 self가 있는 것과 없는 것 (Python 프로그래밍, 2022/11/25)
- 7) (새로운 지식) 기존 코드 컴파일 오류 (C 프로그래밍, 2022/11/28)
- 8) C 표준은 무엇인가 (C 프로그래밍, 2022/11/30)
- 9) 무료 대학원 전략에 관한 의견 여쭙보기 (카카오톡, 2022/12/04)
- 10) 납땜할 때 왜 모양을 봉긋하게 해야하나요 (전기/전자 회로 + 카카오톡, 2022/12/05)

(복습) 매크로 함수에 데이터 타입이 없는 이유는 무엇인가?



Reason

아래 python like list에서 obj, data에 데이터 타입이 없는 이유는 무엇인가에 대한 질문이 있었음.

```
#define array_list_add(obj, data) \
{\
    int type = typecheck(data);\
    printf("type: %d\n", type);\
    array_list_add_impl(obj, data, type);\
}
```

Try

Solution

#define을 사용한 매크로 내부에는 데이터 타입을 명시하지 않습니다.

이와 같은 이유로 여러가지 데이터 타입을 수용하고자 하는 경우에도

#define을 활용한 매크로 함수 방식을 사용하여 처리할 수 있습니다.

(참고) 과제 답안에 존재하는 오류 발견

Description

아래와 같이 두 부분에 오류가 존재합니다.

```
array_lists.h  array_lists.c x
data_structure > list > array_lists.c > init_array_list(array_list_obj **, int)
1
8  #include "array_lists.h"
9  #include "../memory/memory_alloc.h"
10
11 int init_array_list (array_list_obj **obj, int size)
12 {
13     int i;
14     *obj = (array_list_obj *)malloc(sizeof(array_list_obj));
15
16     if (obj == NULL) { return MEMORY_ALLOC_FAILURE; }
--
```

obj == NULL 이 아니라 *obj == NULL 아닌가요?

오후 5:44

오후 5:45

오 ㅋㅋㅋㅋㅋㅋㅋㅋ

맞습니다.



오후 5:46

멋지네요

(참고) 과제 답안에 존재하는 오류 발견

또 다른 오류라면 아래에 있습니다.

```
(*obj)->array_list_obj_items = (array_list_metadata **)malloc(sizeof(array_list_metadata) * size);
```

sizeof연산자 안에 array_list_metadata가 아닌 array_list_metadata * 넣는게 맞나요?

오후 5:58

네 맞습니다.

오후 5:59

진



오후 5:59

온근히 실수들이 있네요 ㄷㄷㄷ

오후 6:00

진

근데 실행하면 segment fault 안뜨네요 ㅋㅋ

오후 6:01

테스트를 안 만들고 해서 그런거 같긴합니다.

오후 6:02

진

오후 6:03

오후 6:11

이런 부분 때문에 실제로 프로젝트 코드 리뷰할 때 크로스 체크가 필요하긴 합니다.

(참고) 과제 답안에 존재하는 오류 발견

Solution

테스트 작성을 안 하면 이런 실수들이 발생하는 경향이 존재한다.

다음부터는 테스트 작성 하는 방법을 Lv1에도 도입 하는 것이 좋아 보인다.

Flutter 혹은 Flask 중 하나를 빼고 GoogleTest를 추가하여

C 레벨에서 테스트 작업을 포함 시키는 것이 더 좋아 보인다.

본인은 답안 코드 분석하면서 대부분의 버그를 해결했는데, 테스트를 이용하는 방법도 있다는 것을 알게 되었다.

함수 정의 파일(.c)에 함수 선언을 작성한 헤더파일(.h)을 include 하는 이유



Description

C device.c ×

C device.c > ...

```
1  #include <stdio.h>
2
3  #include "device.h"
4
5  void proc_camera(void)
6  {
7      printf("Camera Processing\n");
8  }
9
```

C device.h ×

C device.h > proc_i2c(void)

```
1  #ifndef __DEVICE_H__
2  #define __DEVICE_H__
3
4  void proc_camera(void);
5  int proc_dc_motor(void);
6  float proc_bldc(void);
7  void proc_pmsm(void);
8  void proc_acim(void);
9  void proc_led(void *);
```

헤더파일을 include 하지 않아도 컴파일할 때 지장이 없는데, 사용하는 이유는 무엇인가요?

일단 제가 추측해본 바로는 함수 프로토타입이 담긴 헤더파일을 include 함으로써 함수의 매개변수 형과 반환형이 올바른지 확인하고자 include 한 것이 아닌가 싶습니다.

함수 정의 파일(.c)에 함수 선언을 작성한 헤더파일(.h)을 include 하는 이유



Solution

넵 그러한 의도도 있고 어쨌든 컴파일된 객체 차원에서 두 정보가 연결 되어야 하기 때문에 위와 같이 배치 했습니다

참고로 이 부분은 컴파일 방식에 따라서도 달라질 수 있는 내용입니다!

Q: 컴파일된 객체 차원에서 두 정보가 연결 되어야한다는 것은 이 소스코드가 특정한 헤더파일과 연관되어있음을 참고할 수 있도록 include 한 뜻인가요?

넵 맞습니다.

(복습) 헤더파일과 소스파일의 관계성 질문

Description

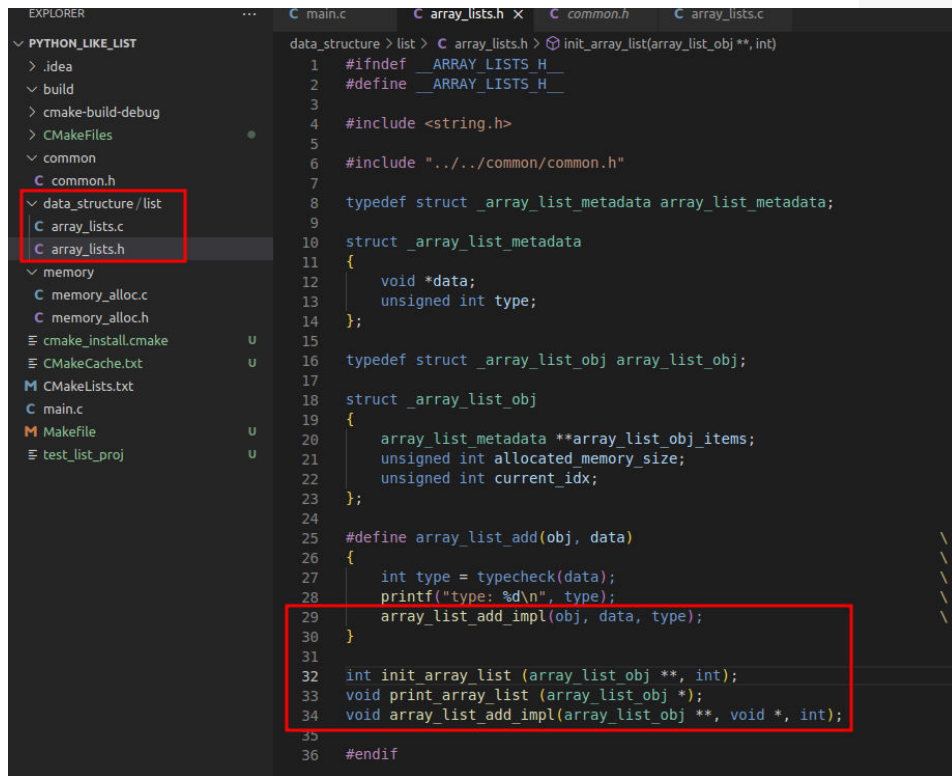
array_lists.h 파일에는 array_lists.c 파일을 참조하고 있지 않습니다.

하지만 array_lists.h 헤더파일은 array_lists.c 소스 파일의 함수를 참조하여 사용하고 있습니다.

프로젝트를 생성하고 동일 경로 상에 소스 파일과 헤더파일을 만들면

자동으로 참조되는건지 궁금합니다.

나아가 소스 .c파일과 헤더 .h파일을 구성하는 전체적인 과정이 어떻게 되는지 궁금합니다.



```
EXPLORER
└─ PYTHON_LIKE_LIST
  └─ .idea
  └─ build
  └─ cmake-build-debug
  └─ CMakeFiles
  └─ common
    └─ common.h
  └─ data_structure / list
    └─ array_lists.c
    └─ array_lists.h
  └─ memory
    └─ memory_alloc.c
    └─ memory_alloc.h
  └─ cmake_install.cmake
  └─ CMakeCache.txt
  └─ CMakeLists.txt
  └─ main.c
  └─ Makefile
  └─ test_list_proj

main.c
array_lists.h
common.h
array_lists.c

data_structure > list > C array_lists.h > init_array_list(array_list_obj **,int)
1  #ifndef  _ARRAY_LISTS_H_
2  #define  _ARRAY_LISTS_H_
3
4  #include <string.h>
5
6  #include "../common/common.h"
7
8  typedef struct _array_list_metadata array_list_metadata;
9
10 struct _array_list_metadata
11 {
12     void *data;
13     unsigned int type;
14 };
15
16 typedef struct _array_list_obj array_list_obj;
17
18 struct _array_list_obj
19 {
20     array_list_metadata **array_list_obj_items;
21     unsigned int allocated_memory_size;
22     unsigned int current_idx;
23 };
24
25 #define array_list_add(obj, data)
26 {
27     int type = typecheck(data);
28     printf("type: %d\n", type);
29     array_list_add_impl(obj, data, type);
30 }
31
32 int init_array_list (array_list_obj **, int);
33 void print_array_list (array_list_obj *);
34 void array_list_add_impl(array_list_obj **, void *, int);
35
36 #endif
```

(복습) 헤더파일과 소스파일의 관계성 질문

Solution

넵 동일 경로상에 만들면 자동으로 참조 됩니다.

실제로 `array_lists.c`에는 `array_lists.h`를 `include` 하는 것이 더 좋을것 같군요.

파이썬 리스트 유사하게 만들기 과제에 관한 질문과 오류



Description

1. 파일 이름을 array_list가 아닌 array_lists로 지은 이유가 알고 싶습니다.
2. array_lists.h 파일에서 구조체의 array_list_obj_items 변수는 왜 왜 단일 포인터가 아닌 이중 포인터로 하신건가요?
3. array_lists.h 파일에서 array_list_add 매크로 함수에서 printf 함수를 사용하므로 헛업하는 관점이나 유지보수하는 관점에서 개발자가 참고할 수 있도록 #include <stdio.h> 해야한다고 생각합니다. 그리고 또한 string.h는 왜 include 되었는가요?
4. 중복 할당 되는 심각한 오류 발견

- array_lists.c 파일
 - init 함수의 빨간 박스를 보면 array_list_metadata를 동적할당하고 있습니다.

```
data_structure > list > C array_lists.c > print_array_list(array_list_obj *)
11 int init_array_list (array_list_obj **obj, int size)
12 {
13     int i;
14     *obj = (array_list_obj *)malloc(sizeof(array_list_obj));
15
16     if (obj == NULL) { return MEMORY_ALLOC_FAILURE; }
17
18     (*obj)->array_list_obj_items = (array_list_metadata **)malloc(sizeof(array_list_metadata) * size);
19
20     if ((*obj)->array_list_obj_items == NULL) { return MEMORY_ALLOC_FAILURE; }
21
22     for (i = 0; i < size; i++) {
23         (*obj)->array_list_obj_items[i] = (array_list_metadata *)malloc(sizeof(array_list_metadata));
24         (*obj)->array_list_obj_items[i]->type = TYPENAME_OTHER;
25     }
26
27     (*obj)->allocated_memory_size = size;
28     (*obj)->current_idx = 0;
29
30     return MEMORY_ALLOC_SUCCESS;
31 }
```

- array_lists.c파일

- 리스트가 만들어지고 난 후 array_list_add_impl 함수로 요소를 저장하는데, memory_type_alloc_table 배열에서 함수를 호출하고 있습니다.

```
57 void array_list_add_impl(array_list_obj **obj, void *data, int type)
58 {
59     int current_idx = (*obj)->current_idx++;
60
61     (*obj)->array_list_obj_items[current_idx] = memory_type_alloc_call_table[type]();
62     (*obj)->array_list_obj_items[current_idx]->data = data;
63 }
```

- memory_alloc.c 파일

- 위 사진에서 memory_type_alloc_call_table 배열 안에 저장된 함수를 보면 array_list_metadata 를 동적할당 하고 있습니다. 그러므로 init 함수에서 이미 초기화 진행했으니 중복 할당입니다.

```
C memory_alloc.c x
memory > C memory_alloc.c > double_alloc(void)
12 array_list_metadata * int_alloc(void)
13 {
14     printf("int_alloc() called\n");
15
16     array_list_metadata *tmp = (array_list_metadata *)malloc(sizeof(array_list_metadata));
17     tmp->type = TYPENAME_INT;
18
19     return tmp;
20 }
```

중복할당을 해결하는 방법: table을 활용하려면 단순히 array_lists.c 파일내에 있는 for문을 주석처리하면됩니다.

파이썬 리스트 유사하게 만들기 과제에 관한 질문과 오류

5. 잠재적인 오류 발견

```
void print_array_list (array_list_obj *obj)
{
    int i;
    int type;
    int size = obj->allocated_memory_size;

    for (i = 0; i < size; i++)
    {
        type = obj->array_list_obj_items[i]->type;

        switch (type) {
            case TYPENAME_INT:
                printf("data: %d\n", obj->array_list_obj_items[i]->data);
                break;
            case TYPENAME_POINTER_TO_CHAR:
                printf("data: %s\n", obj->array_list_obj_items[i]->data);
                break;
            case TYPENAME_POINTER_TO_DOUBLE:
                printf("data: %lf\n", *(double *)obj->array_list_obj_items[i]->data);
                break;
        }
    }
}
```

강사님이 업로드한 파일을 기준으로 설명하겠습니다.

main 함수 내에서는 요소의 개수가 6개인 리스트를 생성하였고, 그 리스트에 총 3개의 요소를 넣었습니다.

그리고 나서 리스트를 출력하는데요.

리스트를 출력하는 함수인 print_array_list 내부를 보면 size 변수가 리스트의 allocated_memory_size의 (여기서는 6) 값을 가져오므로 실제 가지고 있는 요소의 개수보다 더 많이 접근할 수 있습니다.

여기까지만 하겠습니다...

파이썬 리스트 유사하게 만들기 과제에 관한 질문과 오류



Solution

결론적으로 가변 포인터 배열을 만든 것입니다.

string.h는 아마도 작업하면서 남아 있던 레거시 같습니다.

중복 할당 되는 오류는 어떤 내용인지 조금 더 상세하게 이슈 기록 부탁드립니다!

아 맞네요!

for 문에 해당하는 작업을 없애면 됩니다.

Q. 질문 1번부터 다시 답변 부탁드립니다. (TryTwoTop)

→ 답변입니다.

1번: 여러개를 포함할 수 있어서 그렇습니다.

→ 이 부분은 실제로 향후 팀 프로젝트를 진행한다면

복수형 표기를 할 것인지 단수형으로 표기할 것인지 상호간 합을 맞춰야 하는 부분이기도 합니다.

2번: 포인터 배열을 관리할 목적이라 그렇습니다.

→ 좀 더 길게 봐서 자료구조의 확장이 가능하기 때문이죠.

3번: 작업하는 도중에 들어가 있는 레거시입니다.

4번: 나머지 전부 맞고 5번째 케이스는 초기화 할 때 6개를 만들었기 때문에 전체 요소가 출력됩니다.

이 부분을 개선하기 위해 만든 문제가 2번째 개선 문제라 보면 되겠습니다.

실질적으로 3개 있다고 3개를 뿌리는 시스템을 구성하려면

반드시 pre-allocate 방식을 사용해야 합니다.

5 → for문을 주석처리하였기 때문에 array_list_metadata는 array_list_add_impl 함수에서 동적할당 되므로 오류가 발생합니다.. (TryTwoTop)

```
22 // for (i = 0; i < size; i++) {
23 //     (*obj)->array_list_obj_items[i] = (array_list_metadata *)malloc(sizeof(array_list_metadata));
24 //     (*obj)->array_list_obj_items[i]->type = TYPENAME_OTHER;
25 // }
26
27 (*obj)->allocated_memory_size = size;
28 (*obj)->current_idx = 0;
```

```
문제 풀이 디버그 콘솔 터미널 JUPYTER
(base) try@try-desktop:~/eddi/EmbeddedMasterLv1/47/LeeSangHoon/python/python_like_list/build$ ./test_list_proj
type: 5
int_alloc() called
type: 14
char_pointer_alloc() called
type: 17
double_pointer_alloc() called
data: 10
data: purple
data: 37.700000
세그멘테이션 오류 (코어 덤프됨)
(base) try@try-desktop:~/eddi/EmbeddedMasterLv1/47/LeeSangHoon/python/python_like_list/build$
```

→ 해당 파트는 제가 디버깅 해봐야겠네요.

Draft 부탁드립니다!

5 → 강사님의 코드에서 중복할당 오류가 있는 for문(4번)만 주석처리하시면 됩니다! (TryTwoTop)

```
eddi@eddi-C621-SU8:~/proj/EmbeddedMasterLv1/47/JinDongMin/homework/9/answers$ ./proj
type: 5
int_alloc() called
type: 14
char_pointer_alloc() called
type: 16
double_pointer_alloc() called
data: 10
data: purple
data: 37.700000
eddi@eddi-C621-SU8:~/proj/EmbeddedMasterLv1/47/JinDongMin/homework/9/answers$
```

음 동민씨 코드에서 for문 주석된 상태 컴파일하고 실행한 결과입니다.

파이썬 리스트 유사하게 만들기 과제에 관한 질문과 오류



제작성 (TryTwoTop)

```
50 void print_array_list(array_list_obj *obj)
51 {
52     int i;
53     int type;
54     // 잠재적인 오류를 가진 코드를 주석처리 하였음
55     // int size = obj->allocated_memory_size;
56     int size = obj->current_idx;
57
58     for (i = 0; i < size; i++)
59     {
60         type = obj->array_list_obj_items[i]->type;
61
62         switch (type)
63         {
64             case TYPENAME_INT:
65                 printf("data: %d\n", obj->array_list_obj_items[i]->data);
66                 break;
67
68             case TYPENAME_POINTER_TO_CHAR:
69                 printf("data: %s\n", obj->array_list_obj_items[i]->data);
70                 break;
71
72             case TYPENAME_POINTER_TO_DOUBLE:
73                 printf("data: %lf\n", *(double *)obj->array_list_obj_items[i]->data);
74                 break;
75         }
76     }
77 }
```

segmentation fault를 해결하는 방법으로 변수 size를 allocated_memory_size가 아닌 current_idx 값으로 변경했습니다.

위 방법으로 해야 하는 이유

- 일단 init_array_list 함수와 array_list_add_impl 함수에서 array_list_meta 를 동적 할당함
 - (참고)
 - init_array_list 함수: 예를 들어 6개의 요소를 가진 리스트를 초기화한다고 하면 6개를 한 번에 동적 할당함
 - array_list_add_impl 함수: 이 함수는 추가할 요소 하나만 동적 할당함
- 위의 이유로 중복할당을 방지하기 위해 init_array_list 함수내에서 array_list_meta를 동적할당하는 for문 코드를 날림(주석처리) → 이제 array_list_meta를 동적할당하는 함수는 array_list_add_impl 만 남음
- 그러나 print_array_list의 반복문에 사용되는 size는 allocated_memory_size 값으로 했기 때문에 할당되지 않은 메모리를 참조할 수 있음 → segmentation fault 발생
- 그러므로 allocated_memory_size가 아닌 지금까지 할당된 인덱스 또는 개수 정보가 필요하므로 current_idx로 변경함

만약 segmentation fault를 발생시키고 싶다면

- 강사님의 코드에서 중복할당되는 for문만을 주석처리하고 컴파일하시면 됩니다.
- 또는
- 제가 업로드한 코드에서 int size = obj->current_idx; 를 주석처리하고 int size = obj->allocated_memory_size;를 주석 해제하시면 됩니다.

→ 존재하는 대부분의 버그를 다 찾아서 제거한 것 같습니다.

아주 훌륭하네요!

파이썬 리스트 유사하게 만들기 과제에 관한 질문과 오류

코드를 분석하다가..



ㅋㅋㅋ 상당히 잘 하셨네요!



이 문제의 답안 코드 분석하면서 버그 잡느라 시간을 많이 썼는데 이때 나는 인간 디버거였다...

클래스의 self가 있는 것과 없는 것

Reason

```
class Person:
    name = '이름'

    def __init__(self, age):
        self.age = age
```

클래스 Person 내에 있는 name과 age는 어떤 차이를 가지고 있나요?

Try

Solution

Person 내에서 name 으로 구성 했기 때문에 결론적으로 동일하게 self.name 처리해도 무방합니다.

(새로운 지식) 기존 코드 컴파일 오류

동민씨의 경우엔 이것을 스크립트화 해서 ./compile.sh 하면 실행되도록 만들었다 보면 됩니다.

EmbeddedMasterLv1/compile.sh at main · EDDI-RobotAcademy/Em...

This file contains bidirectional Unicode text that may be interpreted or compiled differently than what appears below. To review, open the file in an editor that

<https://github.com/EDDI-RobotAcademy/EmbeddedMasterLv1/blob/main/...>

EDDI-RobotAcademy/
EmbeddedMasterLv1

임베디드 마스터 레벨1

11

Contributors

0

Issues

6

Stars

13

Forks



참고로 이것을 git pull 이나 fetch 로 받아올 경우엔 권한 설정이 안되어 있을 것입니다.

그러므로 아래 명령을 통해 compile.sh 파일에 실행 권한을 줘야 합니다.

755 도 그렇고 a+x도 실행 권한을 주는 작업입니다.

```
chmod 755 compile.sh
```

혹은

```
chmod a+x compile.sh
```

이후 터미널 창에서 ./compile.sh 라고 입력하면 위의 gcc 명령이 자동화 됩니다.

상황 설명

python like list 답안을 참고하여 거의 클론코딩과 비슷하게 만들었는데, 컴파일 방법으로 CMake 대신 gcc 명령어를 대신 실행시켜주는 쉘 스크립트(.sh)를 만들어서 PR 했다.

새로운 지식

다른 사람에겐 git pull로 받아올 때 권한 설정을 해줄 필요가 있다는 점을 알게 되었다.

C 표준은 무엇인가

Description

제 몇몇속에서 C 표준으로 생각나는 것이 '헤더파일'과 '컴파일러의 지원'이라 2가지 예시를 들겠습니다.

Bool

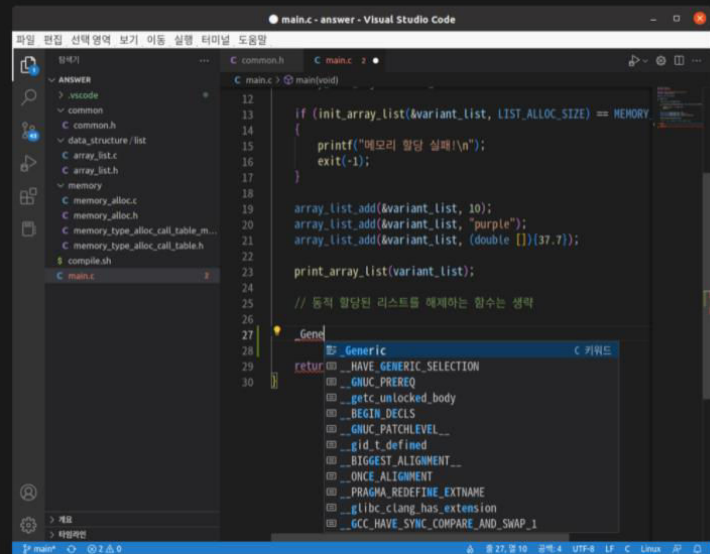
- C99 표준부터 bool 타입을 사용할 수 있습니다. 다만 사용하려면 stdbool.h 헤더 파일을 추가해야 합니다.
- 이는 파일로 존재하기 때문에, 구현이 어떻게 되어있는지 직접 볼 수 있는데요.

```
C common.h C stdbool.h x
usr > lib > gcc > x86_64-linux-gnu > 9 > include > C stdbool.h > ...
25  /* ISO C Standard: 7.16 Boolean type and values <stdbool.h>
26  */
27
28  #ifndef _STDBOOL_H
29  #define _STDBOOL_H
30
31  #ifndef __cplusplus
32
33  #define bool      _Bool
34  #define true      1
35  #define false     0
36
37  #else /* __cplusplus */
38
39  /* Supporting _Bool in C++ is a GCC extension. */
40  #define _Bool      bool
41
42  #if __cplusplus < 201103L
43  /* Defining these macros in C++98 is a GCC extension. */
44  #define bool      bool
45  #define false     false
46  #define true      true
47  #endif
```

하지만! 위와 같이 구현이 어떻게 되어있는지조차 볼 수 없는 것이 있습니다.
바로 키워드입니다.

_Generic Keyword

- C11 표준부터는 _Generic 키워드를 응용하여 타입을 확인할 수 있었습니다.



- 보시는 것과 같이 _Generic은 int나 void 같은 키워드로써 컴파일러(여기서는 gcc)가 지원해줄을 추측할 수 있습니다.

C 표준이 개정된다면 여러 기능이 추가될텐데요.

그 기능을 지원하는 방법 중에 위에서 언급한 헤더파일과 키워드가 맞나요?

만약 맞다면 C 표준의 기능을 지원하는 방법으로 헤더파일과 키워드 뿐만 아니라 다른 방법이 있는 지 궁금합니다.

C 표준은 무엇인가

Solution

말 그대로 키워드입니다.

그렇기 때문에 if, for, while, int, char, float, double 같은 컴파일러 내장 키워드에 해당합니다.

이 구현을 보려면 gcc나 llvm과 같은 컴파일러의 소스 코드를 살펴봐야 합니다.

결론적으로 헤더 파일이 아니라 컴파일러 소스 코드 자체에 기능이 구현되어 있다는 뜻입니다.

C11Status - GCC Wiki

Support for the standard ISO/IEC 9899:2011 (C11) in GCC (draft n1570, PDF) GCC 4.9 Changes: "ISO C11 support is now at a similar level of completeness to ISO C99 support: substantially complete modulo bugs, extended identifiers (supported except



<https://gcc.gnu.org/wiki/C11Status>

main.lv

http://main.lv/writeup/c_c11_standard_generic_keyword.md

C 표준은 무엇인가

→ 질문: 그러면 bool 타입을 사용하려 한다면 C 컴파일러가 지원하는 C 표준 버전하고 관계없이 stdbool.h 헤더 파일을 추가하기만 해도 된다는 것인가요?

넵 현재 bool 타입의 경우엔 stdbool.h에서 해당 타입을 정의해서 사용하기 때문에 그렇습니다.
이 부분은 stdbool.h가 아닌 아래와 같은 방식으로 대응이 가능합니다.

```
#include <stdio.h>

typedef int bool;

int main (void)
{
    bool test = 1;
    printf("%d\n", test);
    return 0;
}
```

무료 대학원 전략에 관한 의견 여쭙보기

오후 12:59

오후 12:59

오후 12:59

오후 12:59

오후 12:59

오후 1:00

오후 1:01

오후 1:02

오후 1:03

오후 1:08

오후 1:09

오후 1:12

오후 1:15

오후 1:40

오후 1:41

오후 1:46

납땀할 때 왜 모양을 bongut하게 해야하나요

주제

납땀할 때 왜 모양을 bongut하게 작업해야하는건가요?

이번 질문은 거의 물리학 질문이라 봐도 될 정도의 질문이군요.

namu.wiki

<https://namu.wiki/w/%EB%82%A9%EB%95%9C>

실제 납의 녹는점은 대략 327도에 해당합니다.

적혀 있듯이 보편적으로 450도 이하인 연납땀을 많이 사용합니다.

그리고 온도 조절 인두기를 사용할 경우 300 ~ 350도 정도를 추천합니다.

납땀할 때 왜 모양을 bonggut하게 해야하나요

납이라는 고체가 녹기 시작해서 계속 열을 받으면 에너지를 누적하게 됩니다.

계속해서 누적된 에너지는 점점 뜨거워지게 되고

열이 주변으로 확산하며 전달되기 때문에

장시간의 납땀은 보편적으로 TR 고장의 주 원인이 됩니다.

낮은 온도는 납이 어설픈게 녹아서 납땀이 뽕죽하게 되는 편이고요.

너무 높은 온도는 위에 적혔듯이 TR이 고장나거나

혹은 들러 붙은 느낌으로 땀질이 되기 때문에

보편적으로 bonggut한 형태를 납땀이 잘 되었다고 이야기 하는 것으로 알고 있습니다.


적정 온도를 벗어나면 물질을 구성하는 녀석들의 에너지가 높아져서

거기서 지들끼리 쿵쿵쿵 하면서 부딪히기 때문에

모양을 잡기가 더 어려워지는 측면이라 해석해도 됩니다.

(이건 물리 질문이라 봐도 무방할 것 같습니다)


납땜할 때 왜 모양을 bongut하게 해야하나요

 병짚 튜브

이슈 답변 드립니다.

이건 거의 물리 문제 느낌이긴 하네요! 오후 8:53

오후 9:04 확인했습니다! 재밌네요 ㅋㅋ

 병짚 튜브

부가적으로 모양은 표면장력 때문이라고 보면 됩니다.
물이 72 정도 되는데 납이 48 정도 되거든요.

복합적인 이유죠 오후 9:05



오후 9:09 표면장력까지.. 아직 갈 길이 머네요

 병짚 튜브

이렇게까지 알 필요는 없다고 봅니다 ;;;
;

질문이 나와서 답변하긴 했지만 말이죠
ㅋㅋ 오후 9:09

오후 9:15 그래도 저에겐 새로운 지식이라 하나 더 가져갑니다