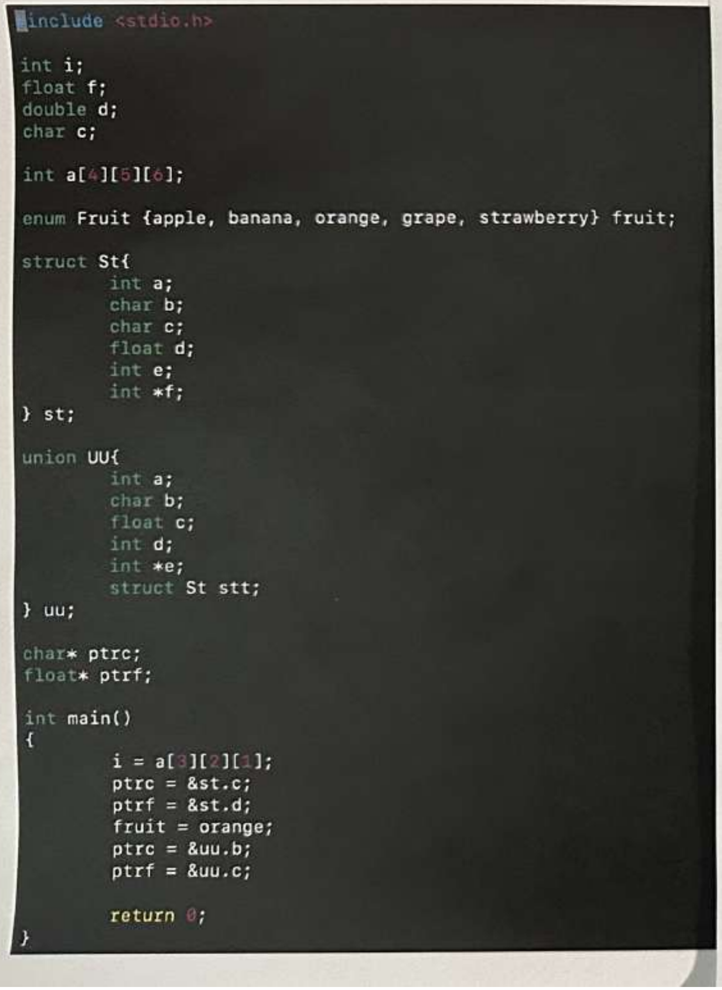
PL_HW3

20191583 김태곤



```c
#include <stdio.h>

int i;
float f;
double d;
char c;

int a[4][5][6];

enum Fruit {apple, banana, orange, grape, strawberry} fruit;

struct St{
        int a;
        char b;
        char c;
        float d;
        int e;
        int *f;
} st;

union UU{
        int a;
        char b;
        float c;
        int d;
        int *e;
        struct St stt;
} uu;

char* ptrc;
float* ptrf;

int main()
{
        i = a[3][2][1];
        ptrc = &st.c;
        ptrf = &st.d;
        fruit = orange;
        ptrc = &uu.b;
        ptrf = &uu.c;

        return 0;
}
```

.comm 변수이름, 크기, 정렬값

.comm i,4,4 → i는 4바이트만큼 메모리 할당 ← int i
.comm f,4,4 → f는 4바이트만큼 메모리 할당 ← float f
.comm d,8,8 → d는 8바이트만큼 메모리 할당 ← double d
.comm c,1,1 → c는 1바이트만큼 메모리 할당 ← char c
.comm a,480,32 → a는 480바이트만큼 메모리 할당 ← int a[4][5][6]
int의 크기 4 × 4×5×6 = 480 enum Fruit{…}
.comm fruit,4,4 → fruit은 4바이트만큼 할당. int형의 크기와 같다 ↵ struct St{…}
.comm st,24,16 → st는 24바이트만큼 할당 (int a:4)(char b:1) ↵
(char c:1)(padding:2)(float d:4)(int e:4)(int *f:8) ⇒ 총 24
.comm uu,24,16 → uu는 24바이트만큼 할당. union은 union 내 선언된 것중 가장 많이 할당된 것의 메모리 크기와 같다. (St stt ; 24) ← union UU{…}
.comm ptrc,8,8 → ptrc는 8바이트만큼 메모리 할당. 포인터는 8바이트 ← char* ptrc
.comm ptrf,8,8 → ptrf는 8바이트만큼 메모리 할당. 포인터는 8바이트 ← float* ptrf

// 여기까지 전역변수로 선언된 변수들의 메모리 할당
// main 시작. 지역변수이므로 stack에 할당되고, 함수가 종료되면 해제된다
// (외부에서 접근 불가)

pushq %rbp → 현재 rbp 레지스터 값을 stack에 push

movq %rsp, %rbp → rsp의 값을 rbp로 옮겨 스택 프레임 설정

movl a+412(%rip), %eax → a+412 주소의 값을 eax레지스터로 로드
a의 시작주소 + (3×5×6 + 2×6 + 1) × (int형 4) = 412 ← a[3][2][1]

movl %eax, i(%rip) → eax의 값을 i로 로드. ← i = a[3][2][1]

movq $st+5, ptrc(%rip) → st의 시작주소+5를 ptrc로 로드
struct St를 보면 &st.c 앞에 int a(4바이트), char b(1바이트)가 있으므로 +5를 해주었다. ← ptrc = &st.c

movq $st+8, ptrf(%rip) → st의 시작주소 +8을 ptrf로 로드
struct St를 보면 &st.d 앞에 int a(4바이트), char b(1바이트) char c(1바이트) padding(2바이트) 있으므로 +8 해주었다 ← ptrf = &st.d

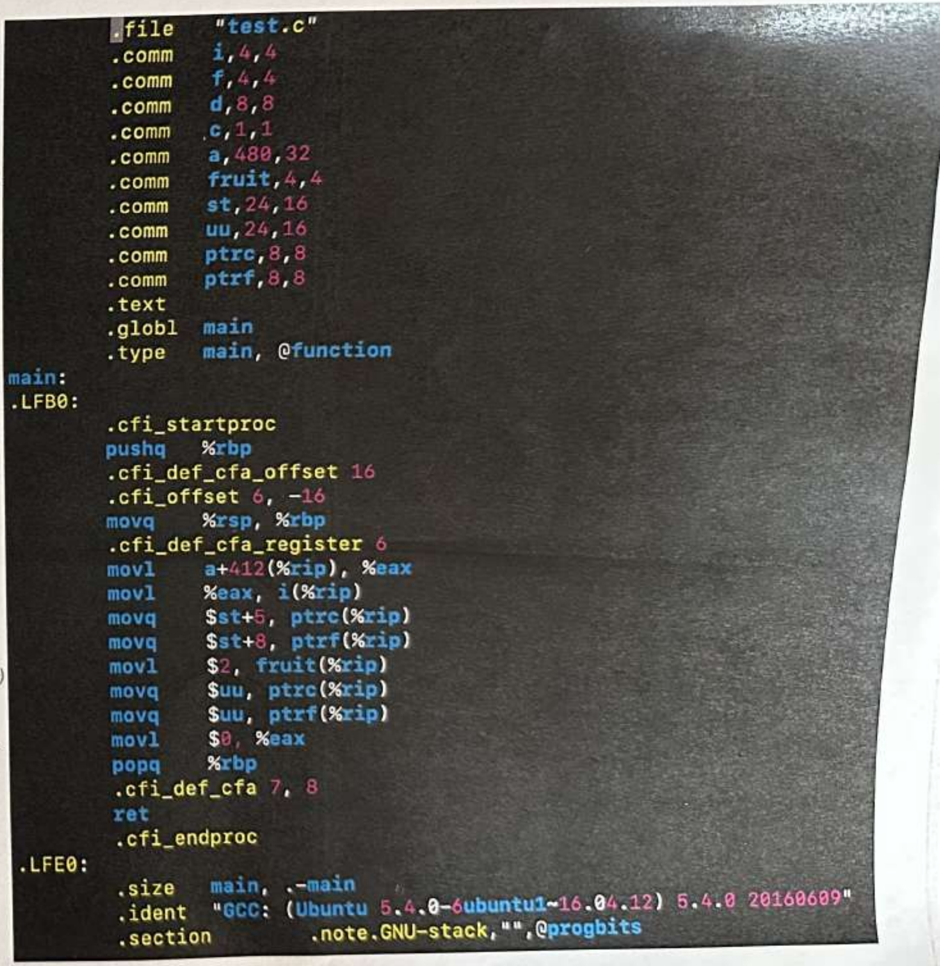
movl $2, fruit(%rip) → 2를 fruit로 로드. enum에 선언된 값들은 0부터 시작하는 정수에 대응된다 orange는 2에 대응된다 ← fruit = orange

movq $uu, ptrc(%rip) → uu의 시작주소를 ptrc로 옮긴다 union에 선언된 변수들은 모두 시작주소에 있다 ← ptrc = &uu.b

movq $uu, ptrf(%rip) → uu의 시작주소를 ptrf로 옮긴다 ← ptrf = &uu.c

movl $0, %eax → eax레지스터에 0을 로드한다 ← return 0;

popq %rbp → 처음 stack에 push했던 rbp를 제거한다



```
        .file   "test.c"
        .comm   i,4,4
        .comm   f,4,4
        .comm   d,8,8
        .comm   c,1,1
        .comm   a,480,32
        .comm   fruit,4,4
        .comm   st,24,16
        .comm   uu,24,16
        .comm   ptrc,8,8
        .comm   ptrf,8,8
        .text
        .globl  main
        .type   main, @function
main:
.LFB0:
        .cfi_startproc
        pushq   %rbp
        .cfi_def_cfa_offset 16
        .cfi_offset 6, -16
        movq    %rsp, %rbp
        .cfi_def_cfa_register 6
        movl    a+412(%rip), %eax
        movl    %eax, i(%rip)
        movq    $st+5, ptrc(%rip)
        movq    $st+8, ptrf(%rip)
        movl    $2, fruit(%rip)
        movq    $uu, ptrc(%rip)
        movq    $uu, ptrf(%rip)
        movl    $0, %eax
        popq    %rbp
        .cfi_def_cfa 7, 8
        ret
        .cfi_endproc
.LFE0:
        .size   main, .-main
        .ident  "GCC: (Ubuntu 5.4.0-6ubuntu1~16.04.12) 5.4.0 20160609"
        .section        .note.GNU-stack,"",@progbits
```