

2018 시스템 프로그래밍
- Lab 04 -

제출일자	2018.10.16
분 반	02
이 름	진승언
학 번	201404377

실습 1 - 프로그램 작성

조건 1 student 라는 이름의 struct 정의

student.c (~/week04) - VIM

```
1 #include <stdio.h>
2
3 struct student{
4     char* myname;
5     int mynumber;
6 };
```

student라는 구조체를 생성하고 char 포인터 형태로 이니셜을 담을 변수와 학번을 담을 mynumber라는 int형 변수를 선언하였다.


조건 2 조교와 학생의 student 구조체를 생성 후 정보 입력

student.c (~/week04) - VIM

```
12 }
13
14 int main()
15 {
16     struct student me;
17     me.myname = "JSE";
18     me.mynumber = 201404377;
19     struct student teacher;
20     teacher.myname = "JBK";
21     teacher.mynumber = 0;
```

main에서 내 정보를 담을 me라는 이름으로 student구조체를 생성하고 그 밑에는 조교님의 정보를 담을 teacher라는 이름으로 student구조체를 각각 1개씩 생성하였다. 그리고 생성한 구조체에 .(점)으로 구조체의 변수에 접근하여 값을 저장해주었다.

조건 3	swap 함수
------	---------

 student.c (~/week04) - VIM

```
7 void swap(struct student *a, struct student *b) {
8     struct student tmp;
9     tmp = *a;
10    *a = *b;
11    *b = tmp;
12 }
```

```
30    printf("swap Function call !!\n");
31    swap(&me, &teacher);
```

call by reference를 하기위해서 swpa 매개변수를 구조체 포인터로 한다. 그리고 값을 임시로 담아놓 student구조체로 tmp라는 이름으로 하나 생성하고 swap해주는 코드를 작성해주면 된다.

swap함수를 호출하는 쪽에서는 call by reference니까 &를 붙여 주소를 전달해준다.

결과화면	셸창에서 결과 화면에 대한 출력
------	-------------------

```
c201404377@2018-sp: ~/week04
c201404377@2018-sp:~/week04$ ./a.out
myname : JSE
mynumber : 201404377
tname : JBK
class : 0
[before] myname : JSE, tname: JBK
[before] mynum : 201404377, tanum : 0
swap Function call !!
[after] myname : JBK, tname: JSE
[after] mynum : 0, tanum : 201404377
c201404377@2018-sp:~/week04$
```

실습 2

1 학생, 조교 저장 데이터 확인

```
c201404377@2018-sp: ~/week04
c201404377@2018-sp:~/week04$ gcc -g -o gdb_student student.c
c201404377@2018-sp:~/week04$ ls
a.out  gdb_student  student
c201404377@2018-sp:~/week04$

c201404377@2018-sp: ~/week04
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law. Type "show copying"
and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.
For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from gdb_student...done.
(gdb) l
1      #include <stdio.h>
2
3      struct student{
4          char* myname;
5          int mynumber;
6      };
7      void swap(struct student *a, struct student *b){
8          struct student tmp;
9          tmp = *a;
10         *a = *b;
(gdb)

c201404377@2018-sp: ~/week04
13
14     int main()
15     {
16         struct student me;
17         me.myname = "JSE";
18         me.mynumber = 201404377;
19         struct student teacher;
20         teacher.myname = "JBK";
(gdb) l
21         teacher.mynumber = 0;
22         printf("myname : %s\n", me.myname);
23         printf("mynumber : %d\n", me.mynumber);
24         printf("tname : %s\n", teacher.myname);
25         printf("class : %d\n", teacher.mynumber);
26
27         printf("[before] myname : %s, tname: %s\n", me.myname, teacher.m
yname);
28         printf("[before] mynum : %d, tanum : %d\n", me.mynumber, teacher
.mynumber);
29
30         printf("swap Function call !!\n");
(gdb) l
31         swap(&me, &teacher);
32         printf("[after] myname : %s, tname: %s\n", me.myname, teacher.my
```

```
c201404377@2018-sp: ~/week04
34      }
(gdb) b 30
Breakpoint 1 at 0x4006db: file student.c, line 30.
(gdb) b swap
Breakpoint 2 at 0x4005e2: file student.c, line 9.
(gdb) b 32
Breakpoint 3 at 0x4006f8: file student.c, line 32.
(gdb) r
Starting program: /home/sys02/c201404377/week04/gdb_student
myname : JSE
mynumber : 201404377
tname : JBK
class : 0
[before] myname : JSE, tname: JBK
[before] mynum : 201404377, tanum : 0

Breakpoint 1, main () at student.c:30
30      printf("swap Function call !!\n");
(gdb) local infos
Undefined command: "local". Try "help".
(gdb) info local
me = {myname = 0x4007d8 "JSE", mynumber = 201404377}
teacher = {myname = 0x4007dc "JBK", mynumber = 0}
(gdb) c
Continuing.

Breakpoint 3, main () at student.c:32
32      printf("[after] myname : %s, tname: %s\n", me.myname, teacher.myname);
(gdb) info local
me = {myname = 0x4007dc "JBK", mynumber = 0}
teacher = {myname = 0x4007d8 "JSE", mynumber = 201404377}
```

gcc -g -o 명령어를 통해 gdb 실행이 가능한 파일을 만들어준다. 그리고 해당 파일을 gdb '해당파일' 명령어로 디버깅을 시작해준다. 그 후 l 명령어로 전체코드를 보고 swap 하기전인 30라인에 b 30으로 breakpoint를 걸어준다. (b swap은 밑에 문제를 위해 breakpoint를 걸었다.) 그리고 swap 이후인 32라인에더 breakpoint를 걸어준다. 이렇게 breakpoint를 다 설정해준 후 r명령어로 디버깅을 시작하고 30라인 breakpoint에서 멈춘다. 여기서 info local명령어를 통해 학생, 조교 저장 데이터 확인을 할 수가 있었다. 그 후 c명령어로(다음 브레이크포인트로 이동) 32라인 breakpoint를 가서 다시 한번 info local 명령어로 swap이후이기 때문에 값이 서로 교환되어 저장됨을 볼 수 있었다.

```
(gdb) b swap
Breakpoint 2 at 0x4005e2: file student.c, line 9.
```

```
c201404377@2018-sp: ~/week04
me = {myname = 0x4007d8 "JSE", mynumber = 201404377}
teacher = {myname = 0x4007dc "JBK", mynumber = 0}
(gdb) c
Continuing.
swap Function call !!
Breakpoint 2, swap (a=0x7fffffff4f0, b=0x7fffffff500) at student.c:9
9      tmp = *a;
(gdb) n
10     *a = *b;
(gdb) n
11     *b = tmp;
(gdb) n
12   }
(gdb) display tmp
1: tmp = {myname = 0x4007d8 "JSE", mynumber = 201404377}
(gdb) display a.myname
2: a.myname = 0x4007dc "JBK"
(gdb) display a.mynumber
3: a.mynumber = 0
(gdb) display b.myname
4: b.myname = 0x4007d8 "JSE"
(gdb) display b.mynumber
5: b.mynumber = 201404377
```

먼저 swap 메소드에 b swap으로 breakpoint를 걸어준다. 그리고 c로 swap메소드 breakpoint까지가서 info local로 tmp의 값을 위 사진과 같이 알아낼 수 있고 값이 바뀌어서 저장된 a(나)와 b(튜터)의 값을 display a.myname , display a.mynumber, display b.name, display b.mynumber 명령어를 통해 알 수 있었다. 그리고 이것을 통해 값이 swap되서 교환되었음을 볼 수 있었다.