

# 1번

<menu.h, menu.c, guest.h, guest.c, main.c 소스>

## [menu.h]

```
#include <stdio.h>
```

```
void displayMenu();
```

## [menu.c]

```
//menu.c
```

```
#include "menu.h"
```

```
void displayMenu() {  
    printf("*****\n");  
    printf("1. Pizza : 20000\n");  
    printf("2. Spaghetti : 12000\n");  
    printf("*****\n");  
}
```

## [guest.h]

```
#include <stdio.h>
```

```
int addGuest();
```

```
void displayGuest(int menu);
```

## **[guest.c]**

```
#include "guest.h"
```

```
int addGuest() {  
  
    int menu;  
  
    printf("원하는 메뉴는? ");  
  
    scanf("%d", &menu);  
  
    return menu;  
}
```

```
void displayGuest(int menu) {  
  
    if(menu == 1)  
        printf("Pizza 선택 ");  
  
    else  
        printf("Spaghetti 선택");  
  
    printf("\n");  
}
```

## **[main.c]**

```
#include "menu.h"
```

```
#include "guest.h"
```

```
int main() {
```

```
int menu;

#ifdef DEBUG

    printf("=> DEBUGMODE\n");

#endif

    displayMenu();

    menu = addGuest();

    displayGuest(menu);

    return 0;

}
```

### <Makefile 생성>

shop: main.o menu.o guest.o

```
gcc -o shop main.o menu.o guest.o
```

menu.o: menu.c menu.h

```
gcc -c menu.c -o menu.o
```

guest.o: guest.c guest.h

```
gcc -c guest.c
```

clean:

```
rm *.o shop
```

### <make shop 실행>

```
s21800201@walab-HGU:~/2021OSS/lab5$ make shop
gcc -c guest.c
gcc -o shop main.o menu.o guest.o
```

설명: Makefile의 shop 커맨드를 이용해 전체적으로 한 꺼번에 컴파일시킨다.

### <make clean 실행>

```
s21800201@walab-HGU:~/20210SS/lab5$ make clean
rm *.o shop
```

설명: 오브젝트 파일과 shop 실행 파일을 제거시킨다

### <make 실행>

```
s21800201@walab-HGU:~/20210SS/lab5$ make
gcc -c menu.c -o menu.o
gcc -c guest.c
gcc -o shop main.c menu.o guest.o
```

설명: Makefile에서 저장된 컴파일 명령어로 오브젝트 파일과 실행파일을 생성시킨다.

### <./shop 실행>

```
s21800201@walab-HGU:~/20210SS/lab5$ ./shop
*****
1. Pizza : 20000
2. Spaghetti : 12000
*****
원하는 메뉴는? 1
Pizza 선택
```

## 2번

### <조건부 컴파일 하는 방법 정리>

조건부 컴파일을 사용하는 이유: 조건부 컴파일을 이용하면 지정한 조건에 따라 코드의 일정 부분을 컴파일 할지 안 할지를 지정할 수 있다.

#define을 통해 문자 혹은 상수의 이름을 정의할 수 있다.

#ifdef을 통해 식별자가 defined 되어있으면 해당 코드를 컴파일 한다.

#ifndef를 통해 식별자가 defined 되어있지 않으면 다음의 코드를 컴파일 한다.

If 문의 else if 또는 else문처럼 #if를 사용하게 되면 #elif, #else 또는 #endif를 통해 마무리를 해야한다.

#define으로 DEBUG가 정의되어 있다면 조건부 컴파일

gcc -DDEBUG -o main\_debug main.c 를 통해 main\_debug라는 DEBUG 조건문이 실행되는 파일을 생성할 수 있다.

### <gdb 사용방법 간단히 정리>

Gdb를 사용할 때 -g 라는 옵션을 이용한다.

예를 들어 main.c 라는 파일을 gdb를 사용하여 컴파일 하려면

Gcc -g main.c -o main

### <각 변경된 소스 및 실행결과 화면캡처 제출>

```
s21800201@walab-HGU:~/2021OSS/lab5$ make shop_debug
gcc -W -Wall -c -o menu.o menu.c
gcc -W -Wall -c -o guest.o guest.c
gcc -W -Wall -DDEBUG -o shop_debug main.c menu.o guest.o
s21800201@walab-HGU:~/2021OSS/lab5$ ./shop_debug
=> DEBUGMODE
*****
1. Pizza : 20000
2. Spaghetti : 12000
*****
원하는 메뉴는? 2
Spaghetti 선택
```

### [Makefile 소스코드]

CC = gcc

CFLAGS = -W -Wall

shop : main.c menu.o guest.o

\$(CC) \$(CFLAGS) -o \$@ \$^

shop\_debug : main.c menu.o guest.o

\$(CC) \$(CFLAGS) -DDEBUG -o \$@ \$^

clean :

rm \*.o shop

rm shop\_debug

### [Makefile\_macro 소스코드]

```
CC = gcc

CFLAGS = -W -Wall

TARGET = shop

DTARGET = shop_debug

OBJECTS = main.o menu.o guest.o

all : $(TARGET)

$(TARGET) : $(OBJECTS)

    $(CC) $(CFLAGS) -o $@ $^

$(DTARGET) : $(OBJECTS)

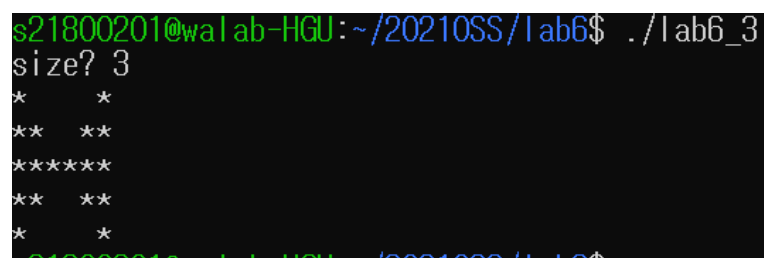
    $(CC) $(CFLAGS) -DDEBUG -o $@ $^

clean:

    rm *.o shop

    rm shop shop_debug
```

### [Gdb 실행 결과]



```
s21800201@wallab-HGU:~/20210SS/lab6$ ./lab6_3
size? 3
* *
** **
*****
** **
* *
```

```

s21800201@walab-HGU:~/2021OSS/lab6$ gcc -g -o lab6_3 lab6_3.c
s21800201@walab-HGU:~/2021OSS/lab6$ gdb lab6_3
GNU gdb (Ubuntu 8.1.1-0ubuntu1) 8.1.1
Copyright (C) 2018 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
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 lab6_3...done.
(gdb)

```

```

(gdb) list
1      #include <stdio.h>
2
3      int main(void) {
4          int size, scnt, bcnt;
5          printf("size? ");
6          scanf("%d", &size);
7          scnt = 1;
8          bcnt = (size - 1) * 2;
9          for(int i = 0 ; i < size*2-1 ; i++) {
10             for(int j = 0 ; j < scnt ; j++) printf("*");

```

```

(gdb) l
11             for(int j = 0 ; j < bcnt ; j++) printf(" ");
12             for(int j = 0 ; j < scnt ; j++) printf("*");
13             if(i < size-1) {
14                 scnt++; bcnt -=2;
15             } else {
16                 scnt--; bcnt +=2;
17             } printf("\n");
18         }
19         return 0;
20     }

```

```

(gdb) b main
Breakpoint 1 at 0x772: file lab6_3.c, line 3.
(gdb) b 9
Breakpoint 2 at 0x7bc: file lab6_3.c, line 9.
(gdb) b 18
Breakpoint 3 at 0x861: file lab6_3.c, line 18.
(gdb) info break
Num    Type           Disp Enb Address            What
1      breakpoint     keep y   0x0000000000000772 in main at lab6_3.c:3
2      breakpoint     keep y   0x00000000000007bc in main at lab6_3.c:9
3      breakpoint     keep y   0x0000000000000861 in main at lab6_3.c:18

```

```

(gdb) r
Starting program: /home2/class2021/s21800201/2021OSS/lab6/lab6_3

Breakpoint 1, main () at lab6_3.c:3
3      int main(void) {
(gdb) c
Continuing.
size? 3

Breakpoint 2, main () at lab6_3.c:9
9      for(int i = 0 ; i < size*2-1 ; i++) {
(gdb) p size
$1 = 3
(gdb) c
Continuing.
*      *
**    **
*****
**    **
*      *

Breakpoint 3, main () at lab6_3.c:19
19     return 0;
(gdb) l 15, 18
15         } else {
16             scnt--; bcnt +=2;
17         } printf("#n");
18     }

```

```

(gdb) display scnt
1: scnt = 0
(gdb) display bcnt
2: bcnt = 6
(gdb) display
1: scnt = 0
2: bcnt = 6
(gdb) watch
Argument required (expression to compute).
(gdb) c
Continuing.
[Inferior 1 (process 9372) exited normally]
(gdb) bt
No stack.
(gdb) info locals
No frame selected.

```

```

(gdb) q
s21800201@walab-HGU:~/2021OSS/lab6$

```

### 3번

<https://dynamlist.io/d/kB2eEAS-wFzWA-jAGY3VKEcC>



## 4번

<Git 실습과제 text 복사 및 마지막 단계 캡처>

b.

```
C:\Users\Wvldrj>ssh s21800201@walab.handong.edu
```

c.

```
s21800201@walab-HGU:~$ cd 2021OSS
```

```
s21800201@walab-HGU:~/2021OSS$
```

d.

```
s21800201@walab-HGU:~$ git clone https://github.com/jerry10004/calculator-2
```

```
Cloning into 'calculator-2'...
```

```
remote: Enumerating objects: 153, done.
```

```
remote: Total 153 (delta 0), reused 0 (delta 0), pack-reused 153
```

```
Receiving objects: 100% (153/153), 41.99 KiB | 3.23 MiB/s, done.
```

```
Resolving deltas: 100% (85/85), done.
```

e.

```
s21800201@walab-HGU:~$ ls -al
```

```
total 124
```

```
drwx-----  9 s21800201 s21800201  4096  4월  8 16:44 .
```

```
drwxr-xr-x 90 root      root      4096  3월 18 22:19 ..
```

```
-rw-----  1 s21800201 s21800201  8397  4월  8 15:54 .bash_history
```

```
-rw-r--r--  1 s21800201 s21800201   220  4월  5  2018 .bash_logout
```

```

-rw-r--r-- 1 s21800201 s21800201 3771 4월 5 2018 .bashrc
drwx----- 2 s21800201 s21800201 4096 3월 23 10:42 .cache
drwx----- 3 s21800201 s21800201 4096 3월 23 10:42 .gnupg
-rw-r--r-- 1 s21800201 s21800201 807 4월 5 2018 .profile
-rw----- 1 s21800201 s21800201 14455 4월 8 16:28 .viminfo
drwxrwxr-x 6 s21800201 s21800201 4096 4월 8 15:54 2021OSS
drwxrwxr-x 2 s21800201 s21800201 4096 3월 27 13:50 Test
drwxrwxr-x 3 s21800201 s21800201 4096 4월 8 16:44 calculator-2
-rw-rw-r-- 1 s21800201 s21800201 30 3월 28 16:27 data2.txt
-rw-rw-r-- 2 s21800201 s21800201 30 3월 27 13:45 data3.txt
-rw-rw-r-- 1 s21800201 s21800201 0 3월 28 16:29 data4.txt
-rw-rw-r-- 2 s21800201 s21800201 30 3월 27 13:45 data_link
lrwxrwxrwx 1 s21800201 s21800201 9 3월 28 16:24 data_sf_link -> data3.txt
-rw-rw-r-- 1 s21800201 s21800201 39 3월 26 10:45 error
-rw-r--r-- 2 s21800201 s21800201 8980 4월 16 2018 examples.desktop
-rw-r--r-- 2 s21800201 s21800201 8980 4월 16 2018 examples.desktop2
lrwxrwxrwx 1 s21800201 s21800201 16 3월 26 10:24 examples_link -> examples.desktop
drwxr-xr-x 2 s21800201 s21800201 4096 3월 18 21:42 html
lrwxrwxrwx 1 s21800201 s21800201 15 4월 6 10:09 lab6 -> ./2021OSS/lab6/
-rw-rw-r-- 1 s21800201 s21800201 919 3월 26 10:32 list.txt
-rw-r--r-- 1 s21800201 s21800201 344 3월 27 13:28 magic.mgc
lrwxrwxrwx 1 s21800201 s21800201 12 4월 4 18:57 ossl_lab5 -> 2021OSS/lab5
drwxr-xr-x 2 s21800201 s21800201 4096 2월 27 13:39 webapps

```

**f.**

```
s21800201@walab-HGU:~$ cd calculator-2
```

**g.**

```
s21800201@walab-HGU:~/calculator-2$ cp Makefile myMakefile
```

**h.**

```
s21800201@walab-HGU:~/calculator-2$ vim myMakefile
```

```
#21800201_김현욱
```

```
CC = gcc
```

```
CFLAGS = -c -Wall
```

```
LFLAGS = -Wall
```

```
LIBS = -lm
```

```
calc: stack.o calculator.c
```

```
$(CC) $(LFLAGS) -o calc calculator.c stack.c $(LIBS)
```

```
stack_test: stack.o stack_test.c
```

```
$(CC) $(LFLAGS) -o stack_test stack_test.c stack.c $(LIBS)
```

```
stack.o: stack.c stack.h
```

```
$(CC) $(CFLAGS) stack.c $(LIBS)
```

```
clean:
```

```
rm -f *.o calc stack_test
```

.PHONY: calc

**i.**

s21800201@walab-HGU:~/calculator-2\$ cat myMakefile

#21800201\_김현욱

CC = gcc

CFLAGS = -c -Wall

LFLAGS = -Wall

LIBS = -lm

calc: stack.o calculator.c

\$(CC) \$(LFLAGS) -o calc calculator.c stack.c \$(LIBS)

stack\_test: stack.o stack\_test.c

\$(CC) \$(LFLAGS) -o stack\_test stack\_test.c stack.c \$(LIBS)

stack.o: stack.c stack.h

\$(CC) \$(CFLAGS) stack.c \$(LIBS)

clean:

rm -f \*.o calc stack\_test

.PHONY: calc

**j.**

```
s21800201@walab-HGU:~/calculator-2$ git status
```

On branch master

Your branch is up to date with 'origin/master'.

Untracked files:

(use "git add <file>..." to include in what will be committed)

myMakefile

nothing added to commit but untracked files present (use "git add" to track)

**k.**

```
s21800201@walab-HGU:~/calculator-2$ git add .
```

**L.**

```
s21800201@walab-HGU:~/calculator-2$ git commit -m "First Commit"
```

[master d5c642b] First Commit

1 file changed, 19 insertions(+)

create mode 100644 myMakefile

**m.**

```
s21800201@walab-HGU:~/calculator-2$ git log -5 --oneline
```

d5c642b (HEAD -> master) First Commit

1114ec2 (origin/master, origin/HEAD) Merge pull request #16 from TomTheBear/fix-failing-functions

a3c3e8e Merge pull request #15 from TomTheBear/fix-nan-and-inf-types

c4c995c Merge pull request #14 from TomTheBear/fix-double-free

54520c1 Function evaluation should push something on the stack. Return type should be int like doOp

**n.**

```
s21800201@walab-HGU:~/calculator-2$ make
```

```
gcc -c -Wall stack.c -lm
```

```
gcc -Wall -o calc calculator.c stack.c -lm
```

**o.**

```
s21800201@walab-HGU:~/calculator-2$ ./calc
```

2+3

= 5.00000

3\*5

= 15.00000

3+4/2

= 5.00000

3+3\*5/6

= 5.50000

**p.**

```
s21800201@walab-HGU:~/calculator-2$ s21800201@walab-HGU:~/calculator-2$ ls -al
total 112
drwxrwxr-x 3 s21800201 s21800201 4096 4월 8 21:25 .
drwx----- 9 s21800201 s21800201 4096 4월 8 21:21 ..
drwxrwxr-x 8 s21800201 s21800201 4096 4월 8 21:23 .git
-rw-rw-r-- 1 s21800201 s21800201 25 4월 8 16:44 .gitignore
-rw-rw-r-- 1 s21800201 s21800201 1086 4월 8 16:44 LICENSE
-rw-rw-r-- 1 s21800201 s21800201 337 4월 8 16:44 Makefile
-rw-rw-r-- 1 s21800201 s21800201 3613 4월 8 16:44 README.md
-rwxrwxr-x 1 s21800201 s21800201 31072 4월 8 21:25 calc
-rw-rw-r-- 1 s21800201 s21800201 30868 4월 8 16:44 calculator.c
-rw-rw-r-- 1 s21800201 s21800201 358 4월 8 16:51 myMakefile
-rw-rw-r-- 1 s21800201 s21800201 822 4월 8 16:44 stack.c
-rw-rw-r-- 1 s21800201 s21800201 241 4월 8 16:44 stack.h
-rw-rw-r-- 1 s21800201 s21800201 2280 4월 8 21:25 stack.o
-rw-rw-r-- 1 s21800201 s21800201 1940 4월 8 16:44 stack_test.c
```