1.

menu.h 소스

```
void showMenu(char a[], char b[], int c[]);
```

menu.c 소스

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

void showMenu(char a[], char b[], int c[]){
        int i=0;
        printf("***************\n");
        printf("%d. %s : %d\n", i+1, a, c[0]);
        printf("%d. %s : %d\n", i+2, b, c[1]);
        printf("***************\n");
}
```

guest.h 소스

```
void pickMenu(char a[], char b[]);
```

guest.c 소스

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

void pickMenu(char a[], char b[]) {
        int c;
        printf("원하는 메뉴는? ");
        scanf(" %d", &c);
        if (c==1) printf("%s 선택\n", a);
        if (c==2) printf("%s 선택\n", b);
}
```

main.c 소스

```
#include <stdio.h>
#include "menu.h"
#include "guest.h"

int main() {
        char b[20] = "pizza";
        char c[20] = "Chicken";
        int a[2]={20000,12000};
        showMenu(b, c, a);
        pickMenu(b, c);
        return 0;
}
```

makefile 소스

```
CC = gcc
```

```
TARGET = main.c
SRCS = menu.c guest.c
OBJS = $(SRCS:.c=.o)
EXE1 = shop

$(EXE1) : $(OBJS)
	$(CC) $(TARGET) $(OBJS) -o $(EXE1)
%.o: %.c
	$(CC) -c $(SRCS)

.PHONY: all clean
all : $(EXE1)
clean :
	rm -f $(OBJS) $(EXE1)
```

make shop 실행

```
s21900706@walab-HGU:~/2022OSS/lab5$ make shop
gcc -c menu.c guest.c
gcc main.c menu.o guest.o -o shop
```

shop을 만들기 위해 makefile이 실행 된 모습

make clean 실행

```
s21900706@walab-HGU:~/2022OSS/lab5$ make clean
rm -f menu.o guest.o shop
```

makefile 과정에서 만들어졌던 파일 3개를 다시 지움

make 실행

```
s21900706@walab-HGU:~/2022OSS/lab5$ make
gcc -c menu.c guest.c
gcc main.c menu.o guest.o -o shop
```

makefile이 기본파일명이기 때문에 make만 쳐도 해당 파일이 실행된다

./shop 실행

```
s21900706@walab-HGU:~/2022OSS/lab5$ ./shop
****************
1. pizza : 20000
2. Chicken : 12000
****************
원하는 메뉴는? 1
pizza 선택
```

2.

조건부 컴파일 하는 방법

#define, #if, #ifdef, #ifndef, #else, #elif, #endif등의 매크로 상수를 이용하여 조건부 컴파일이 가능

gdb 사용방법 (GNU Debugger)

gcc -g main.c -o main
gdb main

list	현재 위치에서 소스파일 보여줌 list, 2, 5
run	프로그램 시작 run arg0 arg1
break	특정라인이나 함수에 breakpoint 설정 break main, break 3
watch	breakpoint를 변수에 걸어 변수의 값이 변경될 때 break됨
clear	breakpoint 삭제
bt	현재 프로그램의 스택을 보여줌(backtrace)
display	현재 display된 명령의 목록을 보여줌
next	현재 파일에서 다음행을 수행 n 10
step	한줄씩 수행, 함수가 있으면 내부로 들어가 한줄씩 실행
print	수식의 값을 보여줌 print i
kill	현재 실행중인 프로그램의 실행을 취소
cout	Continue, 현재 위치에서 프로그램을 계속 실행
quit	gdb종료

```
[s21900706@walab-HGU:~/2022OSS/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) list
1       #include <stdio.h>
2
3       int main() {
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) list
11                      for (int j=0; j<bcnt; j++) printf(" ");
12                      for (int j=0; j<scnt; j++) printf("*");
13                      if (i<size) {
14                              scnt++;bcnt-=2;
15                      }
16                      else {
17                              scnt--;bcnt+=2;
18                      }
19              printf("\n");
20              }
[(gdb) b
No default breakpoint address now.
[(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 0x83f: 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   0x000000000000083f in main at lab6_3.c:18
[(gdb) r
Starting program: /home2/class2021/s21900706/2022OSS/lab6/lab6_3

Breakpoint 1, main () at lab6_3.c:3
3       int main() {
[(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              printf("\n");
(gdb) l 15, 18
15                      }
16                      else {
17                              scnt--;bcnt+=2;
18                      }
(gdb) display scnt
1: scnt = 2
(gdb) display bcnt
2: bcnt = 2
(gdb) display i
3: i = 0
(gdb) watch i
Hardware watchpoint 4: i
(gdb) c
Continuing.
*    *

Hardware watchpoint 4: i

Old value = 0
New value = 1
0x000055555555484d in main () at lab6_3.c:9
9               for (int i=0; i<size*2-1; i++) {
1: scnt = 2
2: bcnt = 2
3: i = 1
(gdb) bt
#0  0x000055555555484d in main () at lab6_3.c:9
(gdb) info locals
i = 1
size = 3
scnt = 2
bcnt = 2
(gdb) q
A debugging session is active.

        Inferior 1 [process 9219] will be killed.

Quit anyway? (y or n)
[s21900706@walab-HGU:~/2022OSS/lab5$ make shop_debug
 gcc -c menu.c guest.c
 gcc -DDEBUG main.c menu.o guest.o -o shop_debug
[s21900706@walab-HGU:~/2022OSS/lab5$ ./shop_debug
 -> DEBUGMODE
 ***************
 1. pizza : 20000
 2. Chicken : 12000
 ***************
[원하는 메뉴는? 1
 pizza 선택
```

3. git 명령어

- git add index.html index2.html
  - index.html과 index2.html파일을 staging area에 이동
- git add *.*
  - 현재 있는 모든 파일을 staging area에 이동
- git add *.html
  - html파일들을 모두 staging area에 이동
- git commit –m “commit msg”
  - 변경사항에 대한 메시지인 commit msg를 적음
- git commit -am “commit msg”
  - 한번 tracked 파일이 된 파일의 경우 add와 commit을 한번에 할 수 있다
- git commit
  - 커밋을 하고 어떤 코맨트를 남길것인지 물어본다
- git status
  - 현재 디렉토리에 있는 untracked file, tracked file, commit file등의 대한 정보를 알 수 있다
- git clean -f
  - git 저장소에서 untracked file들을 모두 삭제한다
- git log --pretty=oneline –graph
  - git에 커밋된 내역을 그래프 형식으로 한 줄로 나타낸다
- git log --author=Brandon
  - brandon이라는 사람이 커밋한 내용을 볼 수 있다
- git log –oneline
  - 커밋 로그 내용을 한 줄로 보여준다

4.

```
s21900706@walab-HGU:~/2022OSS$ 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 | 2.62 MiB/s, done.
Resolving deltas: 100% (85/85), done.

s21900706@walab-HGU:~/2022OSS$ ls -al
total 28
drwxrwxr-x 7 s21900706 s21900706 4096 4월 5 22:45 .
drwx------ 8 s21900706 s21900706 4096 4월 5 22:18 ..
drwxrwxr-x 3 s21900706 s21900706 4096 4월 5 22:45 calculator-2
drwxrwxr-x 3 s21900706 s21900706 4096 3월 23 11:45 lab4
drwxrwxr-x 2 s21900706 s21900706 4096 4월 5 21:06 lab5
drwxrwxr-x 3 s21900706 s21900706 4096 4월 5 22:02 lab6
drwxrwxr-x 2 s21900706 s21900706 4096 4월 5 00:38 miniproject

s21900706@walab-HGU:~/2022OSS$ ls
calculator-2 lab4 lab5 lab6 miniproject
s21900706@walab-HGU:~/2022OSS$ cd calculator-2/
s21900706@walab-HGU:~/2022OSS/calculator-2$ cp Makefile newMakefile
s21900706@walab-HGU:~/2022OSS/calculator-2$ ls
LICENSE Makefile README.md calculator.c newMakefile stack.c stack.h stack_test.c
s21900706@walab-HGU:~/2022OSS/calculator-2$ vim newMakefile
s21900706@walab-HGU:~/2022OSS/calculator-2$ cat newMakefile
#21900706 조영관

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

s21900706@walab-HGU:~/2022OSS/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)

newMakefile

nothing added to commit but untracked files present (use "git add" to track)
s21900706@walab-HGU:~/2022OSS/calculator-2$ git add .
s21900706@walab-HGU:~/2022OSS/calculator-2$ git commit -m "조영관 Makefile commit"
[master 0e2c20e] 조영관 Makefile commit
1 file changed, 20 insertions(+)
create mode 100644 newMakefile

s21900706@walab-HGU:~/2022OSS/calculator-2$ git log -5 --oneline
0e2c20e (HEAD -> master) 조영관 Makefile 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

s21900706@walab-HGU:~/2022OSS/calculator-2$ make -f newMakefile
gcc -c -Wall stack.c -lm
gcc -Wall -o calc calculator.c stack.c -lm

s21900706@walab-HGU:~/2022OSS/calculator-2$ ./calc
12
= 12
3123
= 3123
1*2
= 2.00000
43^2
= 1849.00000

```
s21900706@walab-HGU:~/2022OSS/calculator-2$ ls -al
total 76
drwxrwxr-x 3 s21900706 s21900706  4096  4월   5 22:59 .
drwxrwxr-x 7 s21900706 s21900706  4096  4월   5 22:45 ..
drwxrwxr-x 8 s21900706 s21900706  4096  4월   5 22:50 .git
-rw-rw-r-- 1 s21900706 s21900706    25  4월   5 22:45 .gitignore
-rw-rw-r-- 1 s21900706 s21900706  1086  4월   5 22:45 LICENSE
-rw-rw-r-- 1 s21900706 s21900706   337  4월   5 22:45 Makefile
-rw-rw-r-- 1 s21900706 s21900706  3613  4월   5 22:45 README.md
-rw-rw-r-- 1 s21900706 s21900706 30868  4월   5 22:45 calculator.c
-rw-rw-r-- 1 s21900706 s21900706   358  4월   5 22:49 newMakefile
-rw-rw-r-- 1 s21900706 s21900706   822  4월   5 22:45 stack.c
-rw-rw-r-- 1 s21900706 s21900706   241  4월   5 22:45 stack.h
-rw-rw-r-- 1 s21900706 s21900706  1940  4월   5 22:45 stack_test.c
```