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)

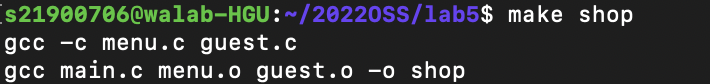
.PHONY: all clean

all : $(EXE1)

clean :

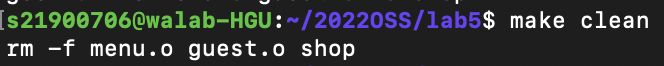
rm -f $(OBJS) $(EXE1)

make shop 실행



shop을 만들기 위해 makefile이 실행 된 모습

make clean 실행



makefile 과정에서 만들어졌던 파일 3개를 다시 지움

make 실행

텍스트이(가) 표시된 사진

자동 생성된 설명

makefile이 기본파일명이기 때문에 make만 쳐도 해당 파일이 실행된다

./shop 실행

텍스트이(가) 표시된 사진

자동 생성된 설명

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 |

텍스트이(가) 표시된 사진

자동 생성된 설명

텍스트이(가) 표시된 사진

자동 생성된 설명텍스트이(가) 표시된 사진

자동 생성된 설명텍스트이(가) 표시된 사진

자동 생성된 설명

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

텍스트이(가) 표시된 사진

자동 생성된 설명