2024.10.24	실습 과제	학번	22312072
과제 7	리눅스 프로그래밍 환경	이름	차민경

- **참고사항**

모든 실습 과제는 각 문항에서 요구하는 문제의 명령어와 그 출력 결과를 동시에 기재하여야 합니다.

예: **오늘 날짜를 출력하는 명령어를 쓰시오.**
답:
date
2024. 10. 24. (목) 15:00:00 KST

1. 다음은 `hello_gcc.c` 파일이다. 해당 파일을 `gcc` 컴파일러로 실행 파일을 생성하고, 실행 파일을 실행하시오.

```c
#include <stdio.h>

int main(void) {
    printf("Hello GCC!");

    return 0;
}
```

답:

```
yu22312072@acslab-146:~$ cat hello_gcc.c
#include <stdio.h>

int main(void){
        printf("Hello GCC!");

        return 0;
}
yu22312072@acslab-146:~$ gcc hello_gcc.c
yu22312072@acslab-146:~$ ./a.out
Hello GCC!yu22312072@acslab-146:~$
```

2. 다음은 강의 자료 3-1(11p)의 `ex6.c` 파일을 수정한 것이다. 해당 파일을 `gcc`로 컴파일하고 실행한 결과가 "Hello, 1234"로 나오도록 명령어를 작성하시오.

```c
#include <stdio.h>

int main(int argc, char** argv) {
    printf("Hello, %d\n", VALUE);

    return 0;
}
```

답:

```
yu22312072@acslab-146:~$ cat ex6.c
#include <stdio.h>
#define VALUE 1234
int main(int argc, char** argv){
        printf("Hello, %d\n", VALUE);

        return 0;
}
yu22312072@acslab-146:~$ gcc ex6.c
yu22312072@acslab-146:~$ ./a.out
Hello, 1234
```

3. [2]에서 작성한 `ex6.c` 파일을 `gcc`로 컴파일할 때 중간 결과를 저장하여 수행하고, `ex6.i` 내용의 일부를 출력하시오.

답:

```
yu22312072@acslab-146:~$ cd 6/
yu22312072@acslab-146:~/6$ ls
ex6.c
yu22312072@acslab-146:~/6$ gcc -save-temps ex6.c
yu22312072@acslab-146:~/6$ ls
a-ex6.i  a-ex6.o  a-ex6.s  a.out  ex6.c
yu22312072@acslab-146:~/6$ head -5 a-ex6.i
# 0 "ex6.c"
# 0 "<built-in>"
# 0 "<command-line>"
# 1 "/usr/include/stdc-predef.h" 1 3 4
# 0 "<command-line>" 2
yu22312072@acslab-146:~/6$ tail -5 a-ex6.i
int main(int argc, char** argv){
 printf("Hello, %d\n", 1234);

 return 0;
}
```

4. 다음 코드는 컴파일러 최적화가 가능한 코드이다. 컴파일러 최적화를 수행하시오.

```c
#include <stdio.h>

#define PI 3.14159

int main(void) {
    double radius = 10.0;
    double area = PI;
    area = area * radius;
    area = area * radius;

    int value = 10000;
    if (0) {
        printf("value : %d\n", value);
    }

    int max = 100;
    for (int i = 0; i < max + 100; i++) {
        printf("%d ", i * 2);
    }
    printf("\n");

    printf("Area of circle : %f\n", area);
    return 0;
}
```

답:

```c
#include <stdio.h>

#define PI 3.14159

int main(void){
        double area = PI * 10.0;

        int num=0;
        for (int i=0; i<200; i++, num+=2){
                printf("%d ", num);
        }
        printf("\n");

        printf("Area of circle %f\n", area);
        return 0;
}
```

5. 강의 자료 3-1(15p)의 `long.c` 파일을 `main.c`, `copy.c`, `copy.h` 파일로 분리하여 작성하고, 컴파일한 뒤 실행하시오.

답:

cc main.c copy.c
./a.out

```
yu22312072@acslab-146:~/5$ ./a.out
abc
abcdegoa
ejkdn
aie
abcdegoayu22312072@acslab-146:~/5$
```

// cc -c mian.c // cc -c copy.c // cc main.o copy.o // a.out

6. [5]에서 작성한 `main.c`, `copy.c`, `copy.h` 파일을 컴파일하고, 실행 파일 `app`을 생성하는 `Makefile`을 작성하시오.

답:

vi Makefile

```
app: main.o copy.o
	gcc -o app main.o copy.o

main.o: main.c copy.h
	gcc -c main.c

copy.o: copy.c copy.h
	gcc -c copy.c
~
```

7. [6]에서 작성한 `Makefile`에 `clean` 레이블을 추가하고, `clean` 레이블을 통해 오브젝트 파일과 실행 파일을 삭제하도록 작성하시오.

답:

```
app: main.o copy.o
	gcc -o app main.o copy.o

main.o: main.c copy.h
	gcc -c main.c

copy.o: copy.c copy.h
	gcc -c copy.c

clean:
	rm -f *.o
	rm app
~
```

8. `Makefile`에서 `NAME`에 `string`을 정의한 매크로를 만드시오. 그리고, `NAME2`에서 `NAME`을 참조하도록 매크로를 만드시오.

답:

```
yu22312072@acslab-146:~$ vi Makefile
yu22312072@acslab-146:~$ cat Makefile
NAME = string
NAME2 = ${NAME}
```

9. 다음 `Makefile` 파일에 오토 매크로를 적용 가능한 부분을 모두 적용하시오.

```
1   OBJECTS=add.c sub.c main.c
2   TARGET=app
3
4   ${TARGET}: ${OBJECTS}
5       gcc -o ${TARGET} ${OBJECTS}
6
7   main.o: main.c app.h
8       gcc -c main.c -o main.o
9   add.o: add.c app.h
10      gcc -c add.c -o add.o
11  sub.o: sub.c app.h
12      gcc -c sub.c -o sub.o
```

답:

```
OBJECTS=add.c sub.c main.c
TARGET=app

${TARGET}: ${OBJECTS}
        gcc -o $@ ${OBJECTS}

main.o: main.c app.h
        gcc -c $< -o $@
add.o: add.c app.h
        gcc -c $< -o $@
sub.o: sub.c app.h
        gcc -c $< -o $@
```

10. 다음 소스 코드를 컴파일하는 `Makefile`을 작성하시오.

add.c

```c
#include <stdio.h>

int add(int a, int b) {
    return a + b;
}
```

sub.c

```c
#include <stdio.h>

int sub(int a, int b) {
    return a - b;
}
```

main.c

```c
#include <stdio.h>

int add(int, int);
int sub(int, int);

int main(void) {
    int a = 5, b = 3;

    printf("a + b = %d\n", add(a, b));
    printf("a - b = %d\n", sub(a, b));
    return 0;
}
```

답:

```
OBJECTS=add.c sub.c main.c
TARGET=app

${TARGET}: ${OBJECTS}
        gcc -o $@ ${OBJECTS}
```