컴퓨터네트워크

과제 #03 문제 및 보고서

이름	김대욱
학번	202255513
소속 학과/대학	정보의생명공학대학 정보컴퓨터공학부
분반	061 (담당교수: 김태운)

〈실습 과제〉

[Q 1] 텍스트 에디터 [배점: 10]
텍스트 에디터는 소스코드를 작성하기 위해 반드시 필요한 프로그램이며, 리눅스 OS 에서 사용할 수 있는 다양한 텍스트 에디터가 있습니다. GUI 지원 여부에 따라서 command-line interface (CLI) text editor 와 GUI text editor 로 분류할 수 있습니다. Kali 에 기본적으로 설치된 CL 에디터로는 vim, nano 등이 있고, GUI 에디터로는 mousepad 등이 있습니다. 그 외에, VS Code 등을 추가로 설치하여 사용할 수 있습니다.

`hello.c` 라는 텍스트 파일을 만들고(`$touch  hello.c`), 텍스트 편집기를 사용해서 "`Hello world!\n`"를 터미널에 출력하는 소스코드를 작성하세요.

문제 1) 터미널 화면에서 `$cat hello.c` 를 입력하고, 터미널 화면을 캡처하여 아래에 첨부하세요.

답변 1)

```
┌──(nozerose㉿ nozerose)-[~/Documents/HW3]
└─$ cat hello.c
#include <stdio.h>

int main() {
        printf("Hello World!\n");
        return 0;
}
```

[Q 2] 소스코드 컴파일 [배점: 10]
C 언어로 작성된 소스코드를 컴파일 하고, 실행 가능한 파일을 생성해 보겠습니다.

소스코드를 컴파일 하는 방법은 다음과 같습니다.
`$ls` (컴파일 전에 디렉토리에 저장된 파일 목록 확인)
`$gcc hello.c -o hello.exe` (`hello.c` 를 컴파일 하여 `hello.exe` 라는 실행파일 만들기[1])
`$ls` (컴파일 후에 디렉토리에 저장된 파일 목록 확인)
직전 문제에서 작성한 코드를 컴파일하여 '`hello.exe`'라는 실행파일이 만들어 졌습니다.
상대경로를 사용하여 이 파일을 실행하는 방법은 `$./hello.exe` 라고 입력하면 됩니다.

문제 1) `hello.exe` 파일을 실행하고, 터미널 화면을 캡처하여 아래에 첨부하세요.

답변 1)

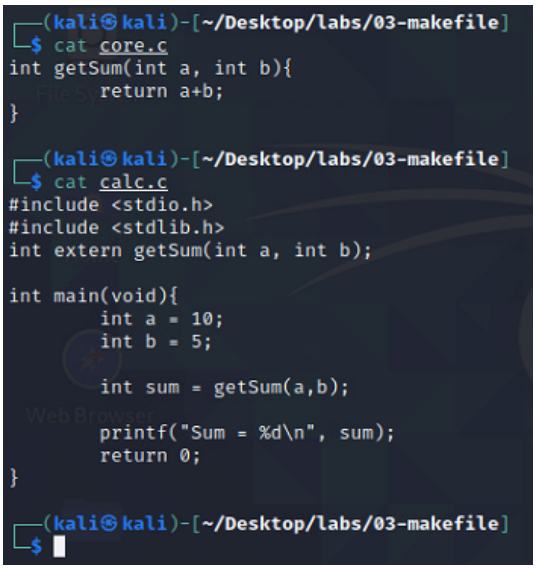
```
┌──(nozerose㉿ nozerose)-[~/Documents/HW3]
└─$ gcc hello.c -o hello.exe

┌──(nozerose㉿ nozerose)-[~/Documents/HW3]
└─$ ls
hello  hello.c  hello.exe

┌──(nozerose㉿ nozerose)-[~/Documents/HW3]
└─$ ./hello.exe
Hello world!
```

[1] 여기서 .exe 라는 확장자를 붙인다고 해서 실행파일이 되는 것은 아닙니다. hello.exe 대신, hello.out 등 아무 확장자를 입력해도 실행파일이 됩니다. 윈도우 사용자에게는 exe 확장자가 실행 파일의 확장자로 익숙하기 때문에 hello.exe 이름을 사용한 것 입니다.

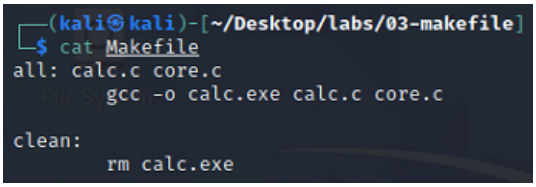
[Q 3] Makefile [배점: 10]

Makefile 은 컴파일을 자동화해주는 도구입니다. 여러 개의 소스코드로 구성된 프로젝트의 경우 특히나 유용합니다. 최근에 수정된 소스 코드만 컴파일 해 주는 기능을 가지고 있어서, 대규모 프로젝트의 경우 컴파일 시간을 절약하는데 도움이 됩니다. 그리고, 컴파일 명령이 복잡한 경우, 복잡한 컴파일 명령을 매번 입력하지 않고도 컴파일 할 수 있어서 유용합니다.

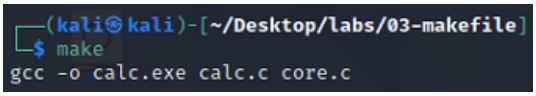
'03-makefile' 디렉토리를 생성하고, 그 아래에 다음과 같이 두개의 소스 코드를 생성하세요 (core.c, calc.c).



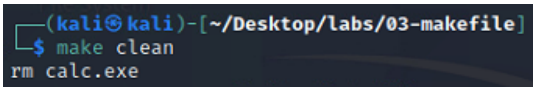
두개의 소스코드를 동시에 컴파일 하는 방법은: `$gcc calc.c core.c -o calc.exe` 입니다. 매번 컴파일 할 때 마다 위의 명령어를 입력하는 대신, Makefile 을 이용하면 간편하게 컴파일 할 수 있습니다. Makefile 이라는 이름의 파일을 만들고, 아래와 같이 입력하세요.



터미널 화면에서 $make 라고 입력하면 아래와 같이 자동을 컴파일을 수행합니다.



실행파일을 삭제하고자 하면 $make clean 을 입력하세요.



Makefile 을 제대로 만들어봅시다. 03-makefile 디렉토리를 복사해서 04-makefile-all 디렉토리를 만들고, Makefile 을 아래와 같이 수정하세요.

```
┌──(kali㉿kali)-[~/Desktop/labs/04-makefile-all]
└─$ cat Makefile
CC=gcc

all: calc.exe

calc.exe: calc.o core.o
	$(CC) -o calc.exe calc.o core.o

calc.o: calc.c
	$(CC) -c -o calc.o calc.c

core.o: core.c
	$(CC) -c -o core.o core.c

clean:
	rm -f calc.exe *.o
```

문제 1) `$make` 를 입력해서 컴파일하세요. 그리고, 다시 한번 `$make` 를 입력하면 어떤 메시지가 나오나요? 왜 이런 메시지가 나오나요?

문제 2) `core.c` 파일을 열고 `return a+b;`를 `return a+b+0;`로 수정한 뒤 다시 `$make` 를 입력하세요. 모든 소스코드 (`calc.c, core.c`)가 재컴파일 되었나요? 예/아니오로 답하고, 그 이유는 무엇인지 설명하세요.

답변 1)

```
┌──(nozerose㉿nozerose)-[~/Documents/HW3/04-makefile-all]
└─$ make
gcc -c -o calc.o calc.c
gcc -c -o core.o core.c
gcc -o calc.exe calc.o core.o

┌──(nozerose㉿nozerose)-[~/Documents/HW3/04-makefile-all]
└─$ make
make: Nothing to be done for 'all'.
```

make: Noting to be done for 'all'. 이라고 나오는 이유는 모든 의존성과 현재 상태를 검사하고, 변경된 소스 파일이나 의존성이 없을 때 나온다. 즉, 현재 프로젝트의 모든 소스파일을 다시 컴파일 할 필요가 없기 때문에 해당 메시지가 뜬다.

답변 2)

```
┌──(nozerose㉿nozerose)-[~/Documents/HW3/04-makefile-all]
└─$ make
gcc -c -o core.o core.c
gcc -o calc.exe calc.o core.o
```

예. core.o 와 core.c 파일이 재컴파일 되었다. 의미상으로는 return a + b; 와 return a + b + 0;은 같은 명령이지만, 다른 코드이므로 make 는 변경된 부분을 재컴파일 하게 된다.

[Q 4] 동적 메모리 [배점: 10]
아래와 같이 동작하는 프로그램을 작성하세요. 'malloc' 또는 'calloc' 을 사용하고, main 함수에서 `return` 을 호출하기 전에 동적으로 할당한 모든 메모리 공간을 `free` 하세요.

1) 터미널에서 사용자로부터 정수를 입력 받음 (입력 받은 정수를 n 이라고 하겠습니다)
2) 사용자가 입력한 n 을 크기로 하는 정수 배열을 만들고, 배열에 1 부터 n 까지의 정수를 순차적으로 저장
3) 배열의 처음부터 끝까지 저장된 숫자의 합을 계산하는 반복문을 작성하고, 합을 sum 이라는 int 형 변수에 저장
4) 변수 sum 에 저장된 숫자를 출력

예를 들어, 사용자가 5 라는 숫자를 입력하면 크기가 5 인 정수 배열을 만들고, 1,2,3,4,5 를 배열에 저장하고, 반복문을 이용해서 1 부터 5 까지 합을 계산한 후, 그 결과인 15 를 출력함.

문제 1) 10 을 입력으로 해서 프로그램을 실행하고, 터미널 화면을 캡처해서 아래에 붙여 넣으세요.
문제 2) 25 를 입력으로 해서 프로그램을 실행하고, 터미널 화면을 캡처해서 아래에 붙여 넣으세요.

답변 1)

```
┌──(nozerose㉿ nozerose)-[~/Documents/HW3]
└─$ ./Q4.exe
Please enter the size of the array: 10
Sum is 55
```

답변 2)

```
┌──(nozerose㉿ nozerose)-[~/Documents/HW3]
└─$ ./Q4.exe
Please enter the size of the array: 25
Sum is 325
```

[Q 5] 구조체 [배점: 10]

`struct info {int n;};` 라는 구조체를 아래와 같이 사용하는 `main.c` 를 작성하세요.

'`main.c`' 코드를 아래와 같이 코딩하세요. 먼저, `main.c` 코드에서 main 함수 밖에 아래와 같이 두 개의 함수를 정의하세요.

1) `void callByVal` 이라는 함수를 정의하고, 함수 내에서 `info` 구조체의 `n` 값을 20 으로 설정. 이 함수는 call by value 를 이용해서 `info` 구조체를 전달받음.
2) `void callByRef` 라는 함수를 정의하고, 함수 내에서 `info` 구조체의 `n` 값을 30 으로 설정. 이 함수는 call by reference 를 이용해서 `info` 구조체를 전달받음

'`main`' 함수를 아래와 같이 정의하세요.

3) struct info 구조체 선언 : `struct info myinfo;`
4) `myinfo.n=10;` 실행 후, `myinfo.n` 정수값을 터미널에 출력
5) `callByVal` 에 `myinfo` 전달하고, `callByVal` 에서 리턴하면 `myinfo.n` 정수값을 터미널에 출력
6) `myinfo.n=10;` 을 다시 실행 후, `myinfo.n` 정수값을 터미널에 출력
7) `callByRef` 에 `myinfo` 전달하고, `callByRef` 에서 리턴하면 `myinfo.n` 정수값을 터미널에 출력

문제 1) `main.c` 를 컴파일 후 실행하고, 터미널 화면을 캡처하여 아래에 첨부하세요.
문제 2) 위의 4 번 과정에서 출력한 `myinfo.n` 과 5 번 과정에서 출력한 `myinfo.n` 의 값이 같습니다. 이유는 무엇인가요?
문제 3) 위의 6 번 과정에서 출력한 `myinfo.n` 과 7 번 과정에서 출력한 `myinfo.n` 의 값이 다릅니다. 이유는 무엇인가요?

답변 1)

```
┌──(nozerose㉿ nozerose)-[~/Documents/HW3/Q5]
└─$ vi main.c

┌──(nozerose㉿ nozerose)-[~/Documents/HW3/Q5]
└─$ gcc main.c -o main.exe

┌──(nozerose㉿ nozerose)-[~/Documents/HW3/Q5]
└─$ ./main.exe
10
10
30
```

답변 2) callByVal 함수의 인자로 들어간 구조체는 copy 되어 함수를 호출하므로, main 함수에서 선언된 myinfo 의 n 값을 변경할 수 없으므로, 두 값이 같다.

답변 3) callByRef 함수의 인자로 들어간 구조체 포인터는 main 함수에서 선언된 myinfo 를 가리키고 있으므로, callByRef 함수 내에서 n 의 값을 변경하여 두 값이 다르다.

[Q 6] IP 주소 변환 :10 진수 ➔ 2 진수 [배점: 10]

아래와 같이 동작하는 프로그램을 작성하세요.

1) 터미널에서 사용자로부터 IP 주소를 입력 받음. 사용자는 `1.2.3.4` 와 같은 방식으로 IP 주소를 입력함 (즉, 10 진수 숫자 4 개를 3 개의 점으로 구분한 형태의 문자열로 입력)
2) 입력 받은 IP 주소를 32 비트 이진수로 변환해서 아래와 같은 형식으로 화면에 출력하기. 예를 들어, 사용자가 `1.2.3.4` 라고 IP 를 입력한 경우, 출력은 다음과 같음
   `00000001.00000010.00000011.00000100`

문제 1) 사용자가 `1.2.3.4` 를 입력한 경우, 그 결과(터미널 화면)를 캡처해서 아래에 첨부

문제 2) 사용자가 `210.115.229.76` 을 입력한 경우, 그 결과(터미널 화면)를 캡처해서 아래에 첨부

답변 1)

```
┌──(nozerose㉿ nozerose)-[~/Documents/HW3/Q6]
└─$ ./main.exe
1.2.3.4
00000001.00000010.00000011.00000100
```

답변 2)

```
┌──(nozerose㉿ nozerose)-[~/Documents/HW3/Q6]
└─$ ./main.exe
210.115.229.76
11010010.01110011.11100101.01001100
```

[Q 7] 라우팅 & 비트마스킹 [배점: 20]

3 개의 출력 포트를 가진 라우터가 있고, 라우터에 저장된 라우팅 규칙은 아래와 같음.

우선순위	목적지	서브넷 마스크	인터페이스(=출력 포트)
1	10.0.2.0	255.255.255.0	IF0
2	192.168.0.0	255.255.0.0	IF1
3	0.0.0.0	0.0.0.0	IF2

위의 라우팅 테이블을 기반으로, 목적지 주소가 입력되면 어떤 인터페이스를 사용할지를 출력하는 프로그램 `router.c` 를 작성하시오. 동작 방식은 다음과 같다.(참고: 아래에서 AND 는 비트 연산자 AND 를 의미함)

목적지 주소가 `aaa.bbb.ccc.ddd` 인 IP 주소가 입력으로 주어지면 (IP 주소는 터미널에서 사용자 입력으로 받음),

```
IF( aaa.bbb.ccc.ddd AND 255.255.255.0 == 10.0.2.0 ) PRINT(“Send to IF0);
ELSE-IF (aaa.bbb.ccc.ddd and 255.255.0.0 == 192.168.0.0 ) PRINT(“Send to IF1);
ELSE-IF (aaa.bbb.ccc.ddd and 0.0.0.0 == 0.0.0.0) PRINT(“Send to IF2);
ELSE PRINT("Send to default interface");
```

위와 같이 동작하는 `router.c` 를 코딩하세요.

문제 1) 사용자 입력이 `10.0.2.50` 인 경우, 그 결과(터미널 화면)를 캡처하여 아래에 첨부
문제 2) 사용자 입력이 `192.168.0.199` 인 경우, 그 결과(터미널 화면)를 캡처하여 아래에 첨부
문제 3) 사용자 입력이 `10.20.30.40` 인 경우, 그 결과(터미널 화면)를 캡처하여 아래에 첨부

답변 1)

```
┌──(nozerose㉿ nozerose)-[~/Documents/HW3/Q7]
└─$ ./router.exe
10.0.2.50
Send to IF0
```

답변 2)

```
┌──(nozerose㉿ nozerose)-[~/Documents/HW3/Q7]
└─$ ./router.exe
192.168.0.199
Send to IF1
```

답변 3)

```
┌──(nozerose㉿ nozerose)-[~/Documents/HW3/Q7]
└─$ ./router.exe
10.20.30.40
Sent to IF2
```

[Q 8] 매크로 [배점: 10]

`08-macro` 라는 디렉토리를 만들고, 그 안에 `main.c` 소스코드를 아래와 같이 작성하시오.

```
┌──(kali㉿kali)-[~/Desktop/labs/08-macro]
└─$ cat main.c
#include <stdio.h>
#include <stdlib.h>

int main(void){

#ifdef DEBUG
        printf("This is my debug message ... \n");
#endif

        printf("Hello world\n");

#ifdef DEBUG
        printf("Debugging never stops ... \n");
#endif

        return 0;
}
```

`$gcc -o main-release.exe main.c` 명령을 실행하세요. 다음으로, `$gcc -DDEBUG -o main-debug.exe main.c` 명령을 실행하세요.

문제 1) 두개의 exe 파일을 각각 실행하세요. 출력 구문에 어떤 차이가 있나요?
문제 2) 이와 같은 차이가 발생하는 이유를 설명하시오..

답변 1)

```
┌──(nozerose㉿nozerose)-[~/Documents/HW3/08-macro]
└─$ ./main-release.exe
Hello world

┌──(nozerose㉿nozerose)-[~/Documents/HW3/08-macro]
└─$ ./main-debug.exe
This is my debug message...
Hello world
Debugging never stops...
```

답변 2)

gcc 를 이용하여 컴파일할 때, -DDEBUG 플래그를 넣게되면, main.c 코드 내의 DEBUG 매크로가 정의되어 있는 경우의 내용도 출력하기 때문에 두 실행파일 간의 출력 메시지가 다르다.

[Q 9] Sleep & Kill [배점: 10]

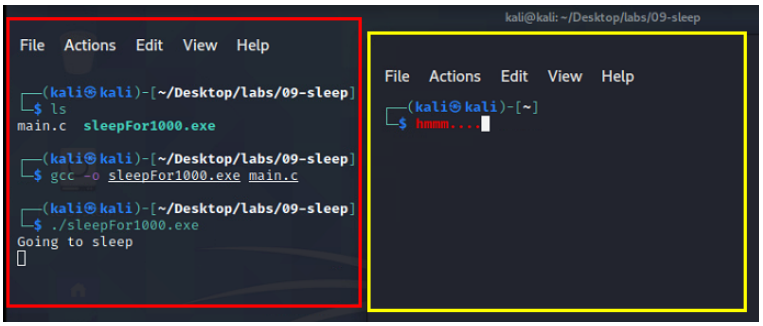
`09-sleep` 이라는 디렉토리를 만들고, `main.c` 함수를 아래와 같이 작성하세요.

```
┌──(kali㉿kali)-[~/Desktop/labs/09-sleep]
└─$ cat main.c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>

int main(void){
        printf("Going to sleep\n");
        sleep(1000); // sleep for 1000 seconds
        printf("Just woke up\n");
        return 0;
}
```

소스코드를 컴파일 해서 `sleepFor1000.exe` 라는 실행파일을 만드세요. 실행파일을 실행하면 다음과 같이 `1000` 초 동안 프로세스가 응답하지 않습니다.

```
┌──(kali㉿kali)-[~/Desktop/labs/09-sleep]
└─$ gcc -o sleepFor1000.exe main.c

┌──(kali㉿kali)-[~/Desktop/labs/09-sleep]
└─$ ./sleepFor1000.exe
Going to sleep
```

여기서 `CTRL+C` 를 누르면 프로세스를 종료할 수 있지만, 다른 방법으로 종료하는 방법을 찾는 문제입니다. 터미널 프로그램을 하나 더 실행하세요. 아래와 같이 붉은색 터미널에서 `sleepFor1000.exe` 프로그램이 실행 중이고, 새로 실행한 노란색 터미널이 있습니다.



문제 1) 새로 실행한 터미널(= 노란색 터미널)에서 `sleepFor1000.exe` 프로그램/프로세스를 종료시킬 수 있는 방법을 설명하세요.

문제 2) 해당 방법을 직접 실행하고, 두 개의 터미널 화면을 캡처하여 아래에 첨부하세요

답변 1)

새로 연 터미널 창에서, ps 명령어를 이용하여 실행중인 "sleepFor1000.exe" 파일의 PID 를 찾아야한다. PID 는 프로세스 id 로, 해당 값을 이용하면 kill -9 <pid> 명령어를 통해 실행되고 있는 프로세스를 강제종료시킬 수 있다.

답변 2)

```
┌──(nozerose㉿ nozerose)-[~]
└─$ ps aux | grep ./sleepFor1000.exe
nozerose  125958  0.0  0.0   2188  1024 pts/0    S+   22:14   0:00 ./sleepFor1000.exe
nozerose  126055  0.0  0.0   3084  1408 pts/2    S+   22:15   0:00 grep --color=auto ./sleepFor1000.exe

┌──(nozerose㉿ nozerose)-[~]
└─$ kill -9 125958
```

```
┌──(nozerose㉿ nozerose)-[~/Documents/HW3/09-sleep]
└─$ ./sleepFor1000.exe
Going to sleep
zsh: killed     ./sleepFor1000.exe

┌──(nozerose㉿ nozerose)-[~/Documents/HW3/09-sleep]
└─$
```