실습3

Process_01

</1>	default
</2>	case 0
</3>	getpid()
</4>	0

결과 스크린샷:

```
kgy@2021320314:/mnt/shared_folder/lab3$ ./process_01
[1187] (Parent) I am old and calm.
[1188] (Child) I am young and wild.
```

Process_02

</1>	pid > 0
</2>	pid == 0
</3>	0
</4>	val

결과 스크린샷:

```
kgy@2021320314:/mnt/shared_folder/lab3$ ./process_02
The original value is 10
The parent will now add 1 and the child will subtract 3
The value of parent is 11.
kgy@2021320314:/mnt/shared_folder/lab3$ The value of child is 7.
```

Process_03

</1>	"/bin/pwd", "pwd", NULL

결과 스크린샷:

```
kgy@2021320314:/mnt/shared_folder/lab3$ ./process_03
where am I?
/mnt/shared_folder/lab3
```

Process_04

</1>	default
</2>	case 0
</3>	pid2
</4>	child_name, name, getpid()
</5>	case 0
</6>	grandchild_name, child_name, name, getpid()

결과 스크린샷:

```
kgy@2021320314:/mnt/shared_folder/lab3$ ./process_04
My name is Jeffrey and I am a parent.
My name is Michael and my father is Jeffrey. My pid is 1234
My name is Steven! My father's name is Michael and my grandpa is called Jeffrey. My pid is 1235
```

Process_05

</1>	pid > 0
</2>	&status
</3>	status/256
</4>	pid == 0
</5>	3

결과 스크린샷:

```
kgy@2021320314:/mnt/shared_folder/lab3$ ./process_05
Counting to three.
1!
2!
3!
```

Process_06

</1>	pid > 0
</2>	pid2 > 0
</3>	pid, &status, 0
</4>	pid2, &status, 0

</5>	pid2 == 0
</6>	pid == 0

결과 스크린샷:

```
kgy@2021320314:/mnt/shared_folder/lab3$ ./process_06
counting to 5!
1!
2!
3!
4!
5!
```

Process_07

process_07.c 3

C: > Users > gykoh > shared_folder > lab3 > process_07.c > main(int, char * [])

```c
#include <stdio.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/wait.h>
#include <stdlib.h>
#define SLEEP_TIME 2

int main(int argc, char* argv[]){
    pid_t pid;
    int status;
    int parent_pid, child_pid, favorite_number;
    char favorite_fruit[] = "apple";


    printf("Final Question!\n");

    pid = fork();

    switch(pid) {
        default:
            parent_pid = getpid();
            printf("[%d] I am a parent\n", parent_pid);
            wait(&status);
            printf("[%d] ....and my child's favorite number is %d\n", parent_pid, status/256);
            break;
        case 0:
            favorite_number = 5;
            sleep(SLEEP_TIME);
            child_pid = getpid();
            printf("[%d] and I am a child!\n", child_pid);
            printf("[%d] my parent's favorite fruit is %s but he doesn't know that my favorite number is %d\n", child_pid, favorite_fruit, favorite_number);
            exit(favorite_number);
            break;
        case -1:
            printf("and I am ???");
            break;
    }


    return 0;
}
```

설명:

line 1~6: 필요한 헤더 파일 포함 및 매크로 정의

line 8~12 : main함수 시작 및 변수 선언

line 17: fork()로 자식 프로세스 생성, pid에 리턴값 저장.

line 20~25: 부모 프로세스 블록.

getpid()로 자신의 pid값를 얻고 첫번째 printf를 출력 후 wait()로 자식이 끝나기를 기다림. 이후 자식이 exit()로 반환한 값은 상위 8비트에 저장되기 때문에 status/256하여 원래 자식이 반환한 값으로 변환하여 두 번째 printf문 출력할때 출력.

line 26~33: 자식 프로세스 블록.

자신만 아는 favorite_number 설정 후 SLEEP_TIME만큼 대기해서 부모의 첫번째 printf문이 먼저 출력될 수 있도록 함. getpid()로 자신의 pid 얻어 온 후 2줄의 문장 출력. 이후 exit(favorite_number) 호출하여 종료하고 부모가 favorite_number받을 수 있도록 함.

line 34~36: fork() 실패시 실행하는 블록

line 40: main() 종료

.

.

.