Fork0.c

```c
#include <sys/types.h>
#include <unistd.h>
#include <sys/wait.h>
#include <stdio.h>

int main(void)
{

   int pid, status;

   pid = fork();
   printf("pid = %d\n", pid);
   while ( 1 ) ;

}
```

```
(base) 202020827@cslinux:~/os/4$ ./fork &
[2] 1696000
bash: ./fork: No such file or directory
[2]+  Exit 127                ./fork
(base) 202020827@cslinux:~/os/4$ ps -l
F S   UID     PID    PPID  C PRI  NI ADDR SZ WCHAN  TTY          T
IME CMD
0 S  1096 1683183 1675412  0  80   0 -  3347 do_wai pts/129  00:00
:00 bash
0 R  1096 1694824 1683183 62  80   0 -   694 -      pts/129  00:00
:47 fork1
1 R  1096 1694826 1694824 71  80   0 -   694 -      pts/129  00:00
:54 fork1
0 R  1096 1696041 1683183  0  80   0 -  3496 -      pts/129  00:00
:00 ps
```

질의

1. 실행하면 몇개의 프로세스가 수행되는가? ->
   2 개
2. 각 프로세스의 결과(출력)는 무엇인가(각각) ->
   pid = 1697190
   pid = 0
3. 부모 프로세스와 자식 프로세의 번호(pid)는 무엇인가? ->
   부모 – 1694824 자식 – 1694826
4. 프로그램 문장(1), (2), (3) 각각의 부모 프로세스와 자식 프로세스 중 어느 것에서 실행되는가?

(1) pid = fork();

부모 프로세스가 실행이 함수를 호출함으로써 자식 프로세스가 생성, 자식 프로세스는 fork()를 “직접” 호출하지는 않지만, 생성 직후 “fork()로부터 반환된 지점”부터 실행을 시작

(2) printf("pid = %d₩n", pid);

부모 프로세스와 자식 프로세스 모두 실행. 단, 부모 프로세스는 pid에 자식의 PID 값이 저장되어 있고, 자식 프로세스는 pid = 0으로 찍힘

(3) while(1);

부모 프로세스와 자식 프로세스 모두 실행. 따라서 두 프로세스 모두 무한 루프에 빠져 종료되지 않음

Fork1.c

```c
#include <sys/types.h>
#include <unistd.h>
#include <sys/wait.h>
#include <stdio.h>
int main(void)
{
   int pid, status;
   pid = fork();
   if (pid > 0) {
   /* parent process */
      printf("PARENT: child=%d\n", pid);
      do { } while (1);
      } else { /* pid == 0 */
      /* child process */
      printf("CHILD: child process is running.\n");
      do { } while (1);
   }
}
```

```
(base) 202020827@cslinux:~/os/4$ ./fork1
PARENT: child=1699521
CHILD: child process is running.
^C
(base) 202020827@cslinux:~/os/4$ ./fork1 &
[1] 1699558
PARENT: child=1699560
CHILD: child process is running.
(base) 202020827@cslinux:~/os/4$ ps -l
F S   UID     PID    PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
0 S  1096 1683183 1675412  0  80   0 -  3347 do_wai pts/129  00:00:00 bash
0 R  1096 1699558 1683183 77  80   0 -   694 -      pts/129  00:00:02 fork1
1 R  1096 1699560 1699558 76  80   0 -   694 -      pts/129  00:00:02 fork1
0 R  1096 1699623 1683183  0  80   0 -  3496 -      pts/129  00:00:00 ps
```

질의

- 실행 시 생성되는 프로세스 수
  - fork()가 한 번 호출되므로, 총 2 개의 프로세스(부모 프로세스와 자식 프로세스)가 생성됨.
- 각 프로세스의 결과(출력)
  - 부모 프로세스
    - 실제로는 pid 값에 따라 달라지며, 자식 PID 가 1699521 로 나타남.
  - 자식 프로세스
    - 출력: CHILD: child process is running.
- 부모 프로세스와 자식 프로세스의 PID
    - 부모 PID: 1699520
    - 자식 PID: 1699521

- 프로그램 문장(1), (2), (3), (4) 각각의 부모 프로세스와 자식 프로세스 중 어느 것에서 실행되는가?

```
(1) pid = fork();
    // 부모와 자식 모두 이 문장을 실행.
    // 단, 부모는 pid>0, 자식은 pid==0 을 반환받음

(2) if (pid > 0) {
      // 부모 프로세스
      (3) printf("PARENT: child=%d\n", pid);
          do { } while (1);
    } else {
      // 자식 프로세스
      (4) printf("CHILD: child process is running.\n");
          do { } while (1);
    }

(3) 부분(부모): printf("PARENT: child=%d\n", pid); → 이후 무한루프

(4) 부분(자식): printf("CHILD: child process is running.\n"); → 이후 무한루프
```

Fork2.c

```c
#include <sys/types.h>
#include <unistd.h>
#include <sys/wait.h>
#include <stdio.h>

int main(void)
{
    int pid, status;
    pid = fork();
    if (pid > 0) {
        printf("PARENT: Child pid = %d\n", pid);
        waitpid(pid, &status, 0);
        printf("PARENT: Child exited (parent is still running)\n");
        do {} while (1);
    } else {
        printf("CHILD : Child process is running.\n");
        do {} while (1);
    }
}
```

```
(base) 202020827@cslinux:~/os/4$ ./fork2
PARENT: CHild pid = 1700739
CHILD : Child process is running.
^C
(base) 202020827@cslinux:~/os/4$ ./fork2 &
[1] 1700797
PARENT: CHild pid = 1700799
(base) 202020827@cslinux:~/os/4$ CHILD : Child process is running.
ps -l
F S   UID     PID    PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
0 S  1096 1683183 1675412  0  80   0 -  3347 do_wai pts/129  00:00:00 bash
0 S  1096 1700797 1683183  0  80   0 -   694 do_wai pts/129  00:00:00 fork2
1 R  1096 1700799 1700797 82  80   0 -   694 -      pts/129  00:00:01 fork2
0 R  1096 1700841 1683183  0  80   0 -  3496 -      pts/129  00:00:00 ps
(base) 202020827@cslinux:~/os/4$
```

- 부모 프로세스와 자식 프로세스의 번호(pid)는 각각 무엇인가?
  - 부모 프로세스의 pid: (예시) 1700738
  - 자식 프로세스의 pid: (예시) 1700739
    실제 실행 시점마다 달라질 수 있지만, 코드상에서 fork()의 반환값이 양수인 프로세스(부모)가 자식의 pid를 알 수 있고, 자식 프로세스 내부에서는 pid == 0 이 된다.
- 실행 직후 부모 프로세스에서 수행되는 문장은 무엇인가?
  - (1) printf("PARENT: Child pid = %d₩n", pid);
- 실행 직후 자식 프로세스에서 수행되는 문장은 무엇인가?

- (5) printf("CHILD: Child process is running.₩n");

- 자식 프로세스 제거 직후(parent 가 waitpid 로 자식 종료를 기다린 후) 부모 프로세스에서 수행되는 문장은 무엇인가?

  - (3) printf("PARENT: Child exited (parent is still running)₩n");