상명대학교 컴퓨터과학과 "유닉스 프로그래밍" 실습 지침서

- 실습 번호: lab-05
- 실습 디렉토리: ~/unix/lab-05 이름: _____
- 실습 날짜: 년 월 일 분반:
- 실습 제목: Process
- 실습 내용: 학번: _____

1. (5-1.c, 5-1) system()을 이용하는 아래의 프로그램 5-1.c를 작성하고 5-1로 컴파일하여 수행한 결과를 적으시오.

```
#include <stdlib.h>
#include <stdio.h>

int main(void) {
   int a;
   a = system("ps -ef | grep sssh");
   printf("Return Value : %d\n", a);

   return 0;
}
```

2. (5-2.c, 5-2) fork()를 이용하여 새로운 프로세스를 만드는 아래 프로그램을 5-2.c로 작성하고 이를 컴파일하여 5-2를 만들고 수행하여 결과를 보이시오.

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

int main(void) {
       pid_t pid;

       printf("Before fork ...\n");

       if((pid = fork())< 0) {
              perror("fork");
              exit(1);
       }
       if(pid > 0)
              printf("Parent process has pid = %d ppid = %d\n",
                     (int)getpid(), (int)getppid());
       else
              printf("Child process has pid = %d ppid = %d\n",
                     (int)getpid(), (int)getppid());

       printf("End of fork ...\n");

       return 0;
}
```

3. (5-3.c, 5-3) 아래와 같이 exec()를 테스트하는 프로그램 5-3.c를 작성하고 이를 컴파일 하여 5-3을 만들고 실행하여 결과를 확인하고 그에 대해 설명하시오.

```
#include <unistd.h>
#include <stdlib.h>
#include <stdio.h>

int main(void) {
    printf("--> Before exec() system call\n");

    if (execlp("ls", "ls", "-a", "-l", (char *)NULL) == -1) {
        perror("execlp");
        exit(1);
    }

    printf("--> After exec() system call\n");

    return 0;
}
```

4. (5-4.c, 5-4) 아래와 같이 fork()와 execv()를 테스트하는 프로그램 5-4.c를 작성하고 이를 컴파일하여 5-4를 만들고 실행하여 결과를 확인하고 그에 대해 설명하시오.

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

int main(void) {
        pid_t pid;
        char *a[3];

        if((pid = fork())< 0) {
                perror("fork");
                exit(1);
        }
        if(pid > 0){
                printf("Parent %d executes.\n", (int)getpid());
        }
        else
        {
                printf("Child %d executes.\n", (int)getpid());
                a[0] = "ls";
                a[1] = "-a";
                a[2] = NULL;
                if(execv("/bin/ls", a) == -1){
                        perror("exec");
                        exit(2);
                }
        }

        return 0;
}
```

5. (5-5.c, 5-5) 아래와 같이 wait()를 테스트하는 프로그램 5-5.c를 작성하고 이를 컴파일 하여 5-5을 만들고 실행하여 결과를 확인하고 그에 대해 설명하시오.

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

int main(void) {
	pid_t pid;
	int i, status;

	if((pid = fork())< 0) {
		perror("fork");
		exit(1);
	}
	if(pid > 0){
		printf("Parent %d waits child %d\n", (int)getpid(), (int)pid);
		wait(&status);
		printf("Child's exit status = %x\n", status);
	}
	else
	{
		for(i = 0; i < 5; i++){
			printf("Child %d executes.\n", (int)getpid());
			sleep(1);
		}
		exit(3);
	}

	return 0;
}
```

6. (5-6.c, 5-6) 아래의 프로그램을 5-6.c로 작성하고 이를 컴파일하여 5-6을 만들고 실행하여 결과를 보이고 결과에 대해 설명하시오.

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

int main(void) {
	pid_t pid, pid1, pid2;
	int i, status;

	if((pid1 = fork())< 0){
		perror("fork");
		exit(1);
	}
	if(pid1 > 0){
		printf("Parent forks 1st child\n");

		if((pid2 = fork()) < 0){
			perror("fork");
```

```
                exit(2);
        }
        if(pid2 > 0){
                printf("parent forks 2nd child\n");
                printf("Parent %d waits children\n", (int)getpid());
                while((pid = wait(&status)) > 0){
                printf("Child %d exit status = %x\n",(int)pid, status);
                }
        }
        else
        {
                for(i = 0; i < 3; i++){
                        sleep(1);
                        printf("Child %d executes.\n", (int)getpid());
                }
                exit(3);
        }
}
else
{
        for(i = 0; i < 3; i++){
                printf("Child %d executes.\n", (int)getpid());
                sleep(1);
        }
        exit(4);
}

return 0;
}
```

끝.