팀 프로젝트 1

교과목명	시스템프로그래밍		분반	001
제출일자	2021-12-13		교수자명	장희숙
팀명	이름	학번	이름	학번
초코파이썬칩	김기태	20173152		
	안대현	20173217		

▣ 프로젝트 주제

팀 프로젝트 1. 간단한 쉘 프로그램을 만들고 다음과 같이 동작하도록 수정하시오.

▣ 프로젝트 결과

1. 간단한 쉘 프로그램을 만들고 다음과 같이 동작하도록 수정하시오..

A) "exit"를 치면 프로그램을 끝내도록 프로그램을 수정하시오.

team_project_1.c

```
else if(pid >0) {
    wait((int*)0);
    if(!strcmp(argv[0], "exit")) {   // "exit"
        exit(0);
    }
```

실행 결과

```
[/root] $ exit
root@9eb578df892c:~/SystemProgramming/lab3/no11# ./team_project_1
```

strcmp를 통해 입력받은 시스템 인자가 exit인 경우 exit()를 실행시켜 프로세스를 종료한다.

B) csh, bash 등에서처럼 쉘 명령의 마지막에 '&'을 입력하면 백그라운드로 실행되도록 프로그램을 수정하시오.

team_project_1.c

```
if(!strcmp(argv[narg-1], "&")) {   //"&"
    printf("& ½ÇÇà¥n");
}
else {
    exit(0);
}
```

실행 결과

```
root@9eb578df892c:~/SystemProgramming/lab3/no11# ./team_project_1
[/root/SystemProgramming/lab3/no11] $ &
[/root/SystemProgramming/lab3/no11] $ ^Z
[1]+  Stopped                 ./team_project_1
root@9eb578df892c:~/SystemProgramming/lab3/no11# ps
  PID TTY          TIME CMD
 3301 pts/0    00:00:00 bash
 4542 pts/0    00:00:00 team_project_1
 4553 pts/0    00:00:00 ps
```

입력받은 시스템 인자가 '&'인 경우 백그라운드에서 프로세스를 실행한다.

C) csh, bash 등에서처럼 인터럽트키 (SIGINT: Ctarl-C, SIGQUIT: Ctrl-Z)가 동작하도록 프로그램을 수정하시오.

team_project_1.c

```
int Signal_Interrupt(struct sigaction *def, sigset_t *mask, void(*handler)(int)) {
    struct sigaction catch;
    catch.sa_handler = handler;
    catch.sa_flags =0;
    def->sa_handler = SIG_DFL;
    def->sa_flags =0;
    if ((sigemptyset(&(def->sa_mask)) ==-1) || (sigemptyset(&(catch.sa_mask)) ==-1) ||
(sigaddset(&(catch.sa_mask), SIGINT) ==-1) || (sigaddset(&(catch.sa_mask), SIGQUIT)
==-1) || (sigaction(SIGINT, &catch, NULL) ==-1) || (sigaction(SIGQUIT, &catch, NULL)
==-1) || (sigemptyset(mask) ==-1) || (sigaddset(mask, SIGINT) ==-1) || (sigaddset(mask,
SIGQUIT) ==-1))
        return -1;
    return 0;
}
void sig_handler(int signo)
{
    pid_t pid ;
    int stat ;
    while ((pid = waitpid(-1, &stat, WNOHANG)) >0)
        printf("child %d terminated normaly₩n", pid) ;
}
```

실행 결과

```
root@9eb578df892c:~/SystemProgramming/lab3/no11# ./team_project_1
[/root/SystemProgramming/lab3/no11] $ &
[/root/SystemProgramming/lab3/no11] $ ^Z
```

SIGINT, SIGQUIT를 통해 사용자가 입력한 인터럽트키를 핸들러를 통해 받아와 실행시킨다. SIGINT의 경우 종료, SIGQUIT의 경우 그만두기를 실행한다.

D) 파일 재지향(>, <)및 파이프(|) 기능이 가능하도록 프로그램을 수정하시오.

team_project_1.c

```
void redirection(int narg, char **argv) {
    pid;
    int i =0;
    int fd;
    int split_index =0, is_write =0;
```

```
    int write_flags = O_WRONLY | O_CREAT | O_TRUNC;
    mode_t mode = S_IRUSR | S_IWUSR | S_IRGRP | S_IROTH;
    char *cmd[10] = {'₩0'}; // redirection을 수행할 명령어 저장.
    for(i =0; i < narg; i++){
        if(!strcmp(argv[i], ">")){
            split_index = i;
            is_write =1;
        }
        else if(!strcmp(argv[i], "<")){
            split_index = i;
            is_write =0;
        }
    }
    for(i =0; i < split_index; i++){
        cmd[i] = argv[i];
    }
    pid = fork();
    if(pid ==0) {
        // > 연산자
        if (is_write){
            if ((fd = open(argv[split_index +1], write_flags, mode)) ==-1) {
                perror("[ERROR] OPEN: ");
                exit(1);
            }
            if (dup2(fd, 1) ==-1) {
                perror("[ERROR] DUP2: ");
                exit(1);
            }
        }
        // < 연산자
        else{
            if ((fd = open(argv[split_index +1], O_RDONLY)) ==-1) {
                perror("[ERROR] OPEN: ");
                exit(1);
            }
            if (dup2(fd, 0) ==-1) {
                perror("[ERROR] DUP2: ");
                exit(1);
            }
        }

        if (close(fd) ==-1) {
            perror("[ERROR] CLOSE: ");
            exit(1);
        }
        execvp(cmd[0], cmd);
    }
    else if (pid >0) {
        wait(pid);
    }
}
```

E) ls, pwd, cd, mkdir, rmdir, ln, cp, rm, mv, cat 명령을 팀원이 공평하게 나누어 구현하시오.

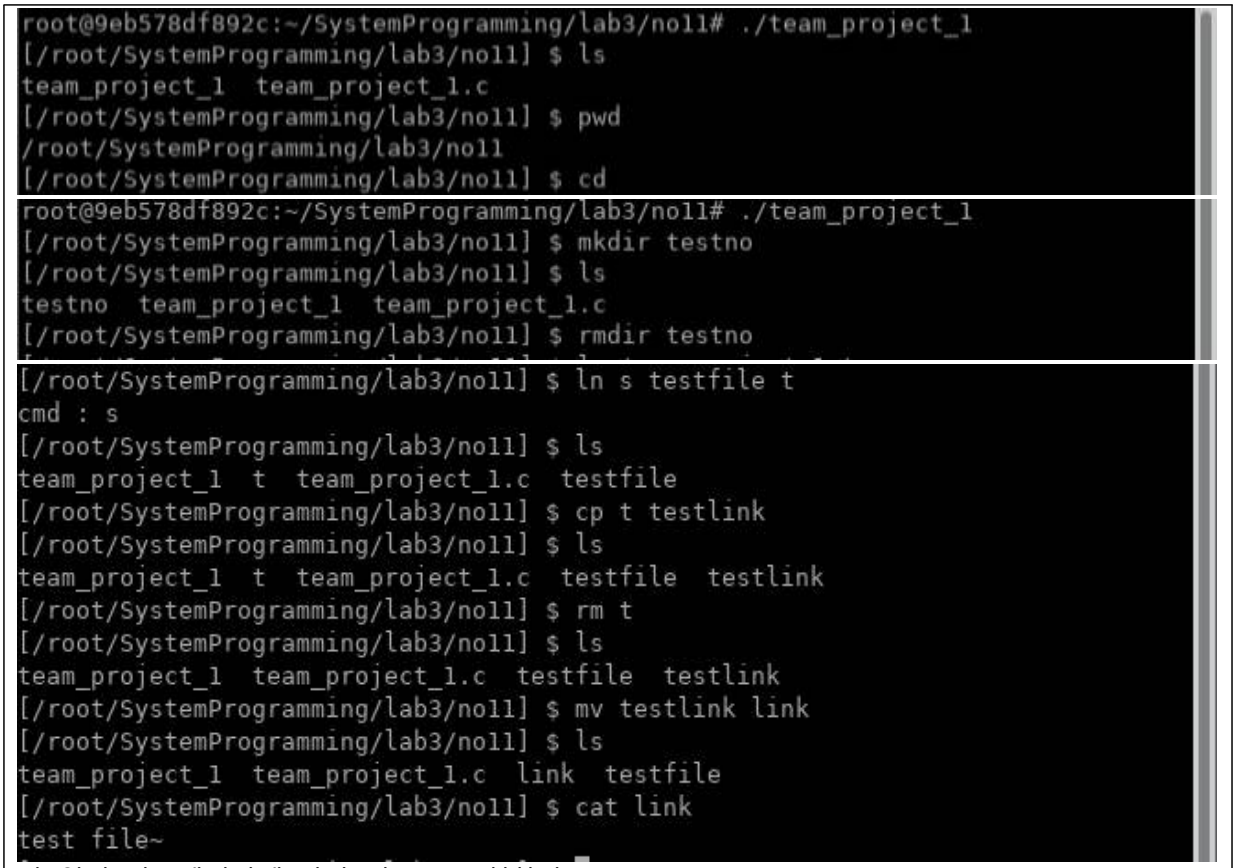
team_project_1.c

```
if(pid ==0) {
            if(!strcmp(argv[0], "ls")) {   // "ls"
```

```
                    strcpy(dir_path, (char*)shmaddr);
                    pdir = opendir(dir_path);

                    while ((pde = readdir(pdir)) !=NULL) {
                        if(strncmp(".",  pde->d_name,1)  ==0  ||  strncmp("..",pde->d_name,2)
==0)
                            continue;
                        printf("%s  ", pde->d_name);
                    }
                    printf("₩n");
                    closedir(pdir);
                }

                else if(!strcmp(argv[0], "pwd")) { // "pwd"

                    getcwd(buf, 1024);
                    printf("%s₩n", buf);
                }

                else if(!strcmp(argv[0], "mkdir")) { // "mkdir"

                    strcpy(dir_name, argv[1]);
                    strcpy(dir_path, getcwd(buf, 1024));
                    strcat(dir_path, "/");
                    strcat(dir_path, dir_name);
                    if(mkdir(dir_path, 0755) >0)
                        printf("can not create dir.");
                }
                else if(!strcmp(argv[0], "rmdir")) { //" rmdir"

                    strcpy(dir_name, argv[1]);
                    strcpy(dir_path, getcwd(buf, 1024));
                    strcat(dir_path, "/");
                    strcat(dir_path, dir_name);
                    if(rmdir(dir_path) >0)
                        printf("can not remove dir.");
                }

                else if(!strcmp(argv[0], "ln")) { // "ln"

                    cmd = (char) *argv[1];
                    printf("cmd : %c₩n", cmd);
                    if (cmd =='|' ) {
                        if (narg <4) {
                            fprintf(stderr, "file_link l src target [link]₩n");
                            exit(1);
                        }
                        src = argv[2];
                        target = argv[3];
                        if (link(src, target) <0) {
                            perror("link");
                            exit(1);
                        }
                    }
                    else if (cmd =='s') {
                        if (narg <4) {
                            fprintf(stderr, "file_link l src target [link]₩n");
                            exit(1);
                        }
```

```
            src = argv[2];
            target = argv[3];
            if (symlink(src, target) <0) {
                perror("symlink");
                exit(1);
            }
        }
        else if (cmd =='u') {
            src = argv[2];
            if (unlink(src) <0) {
                perror("unlink");
                exit(1);
            }
        } else {
            fprintf(stderr, "Unknown command...₩n");
        }
    }
    else if(!strcmp(argv[0], "cp")) { // "cp"

        cp_src = open(argv[1], O_RDONLY);
        cp_dst = open(argv[2], O_WRONLY | O_CREAT | O_TRUNC, 0644);
        while(read(cp_src, &ch, 1))
            write(cp_dst, &ch, 1);
        close(cp_src);
        close(cp_dst);
    }
    else if(!strcmp(argv[0], "rm")) { // "rm"

        unlink(argv[1]);
    }
    else if(!strcmp(argv[0], "mv")) { // "mv"

        char file_name[50];
        strcpy(file_name, argv[1]);
        cp_src = open(argv[1], O_RDONLY);
        cp_dst = open(argv[2], O_RDWR|O_CREAT|O_EXCL, 0664);
        r_size = read(cp_src,buf, 1024);
        w_size = write(cp_dst,buf,r_size);
        while(r_size ==1024) {
            r_size = read(cp_src,buf,1024);
            w_size = write(cp_dst,buf,r_size);
        }
        unlink(file_name);
        close(cp_src);
        close(cp_dst);
    }
    else if (strcmp(argv[0], "clear") ==0) // "clear"
        system("clear");
    else if(!strcmp(argv[0], "cat")) { // "cat"

        cp_src = open(argv[1], O_RDONLY);
        while(read(cp_src, &ch, 1))
            write(1, &ch, 1);
        close(cp_src);
    }
```

실행 결과



각 입력 시스템인자에 따라 기능을 구현한다.
파일 관련된 명령어이므로 파일 open, write, read, remove등 저수준 파일처리 함수부터 getcwd, opendir, readdir, mkdir, rmdir 등 디렉토리 함수, 링크관련 함수등 각 기능에 따른 함수를 통해 기능을 구현한다.
ls, pwd, cd, mkdir, rmdir : 기능 구현자(김기태)
ln, cp, rm, mv, cat : 기능 구현자(안대현)