- 문제
  - 주기적으로 백업함수 실행하는 파일 짜서 메인함수에서 실행시키는게 시험이었음
- 답
  - 메인함수

```
#include<fcntl.h>
#include<unistd.h>
#include<stdio.h>
#include<string.h>
#include<dirent.h>
#include<ftw.h>
#include<signal.h>
#include<stdlib.h>
#include<sys/types.h>
#include<sys/wait.h>
#include<sys/stat.h>
#include<sys/mman.h>
#include<sys/ipc.h>
#include<sys/msg.h>
#include<sys/sem.h>
#include<signal.h>

#define BUFSIZE 512



void catchusr2(int signo){
    exit(1);
}

int main(void){
        char in[50], *res[20]={0};
        int i, status, pid, pid_b=0;

    static struct sigaction act;

    act.sa_handler = SIG_IGN;
    sigaction(SIGINT, &act, NULL);

    act.sa_handler = catchusr2;
    sigaction(SIGUSR2, &act,NULL);

    if((pid_b=fork()) == 0){
        execvp("backup_m",res);
        exit(0);
    }

        while (1){
                printf("> ");
            gets(in);
        if (in[0]=='\0')
            continue;
        i=0;
```

```
            res[i]=strtok(in, " ");
        while (res[i]){
            res[++i]=strtok(NULL, " ");
        }

            if (strcmp(res[0], "exit")==0){
            kill(pid_b, SIGUSR2);
            if(waitpid(pid_b, &status, WNOHANG) == 0){
                wait(0);
                printf("BACKUP 종료 확인...\n");
            }
                exit(0);
        }
            else if (strcmp(res[0], "cd_m")==0){
            chdir(res[1]);
        }
        else{
            if ((pid=fork())==0){
                act.sa_handler = SIG_DFL;
                sigaction(SIGINT, &act, NULL);

                execvp(res[0],res);
                exit(0);
            }
            else {
                waitpid(pid,&status,0);
            }
        }
    }

    return 0;
}
```

- 백업함수 짜기

```
#include<fcntl.h>
#include<unistd.h>
#include<stdio.h>
#include<string.h>
#include<dirent.h>
#include<ftw.h>
#include<signal.h>
#include<stdlib.h>
#include<sys/types.h>
#include<sys/wait.h>
#include<sys/stat.h>
#include<sys/mman.h>
#include<sys/ipc.h>
#include<sys/msg.h>
#include<sys/sem.h>
#include<signal.h>

#define BUFSIZE 512

int backup(const char *name, const struct stat *status, int type){
    char buf[BUFSIZE], file1[BUFSIZE], temp[BUFSIZE] = "./TEMP/";
    int fd1, fd2, i, n, len;
```

```
    if(type == FTW_F){
        strcpy(file1,name+2);
        len = strlen(file1);
        for (i=0; i < len; i++){
            if(file1[i] == '/')
                file1[i] = '_';
        }

        strcat(temp,file1);

        fd1 = open(name, O_RDONLY);
        fd2 = open(temp, O_CREAT | O_WRONLY | O_TRUNC, status-
>st_mode);

        while((n = read(fd1,buf,BUFSIZE))){
            write(fd2,buf,n);
        }
    }

    return 0;
}

void catchusr(int signo){
        return ;
}

void catchalarm(int signo){
        raise(SIGUSR1);
}

int main(int argc, char **argv){
    static struct sigaction act;

    act.sa_handler = catchalarm;
        sigaction(SIGALRM, &act, NULL);

    act.sa_handler = catchusr;
    sigaction(SIGUSR1, &act, NULL);

    sigset_t mask;
    sigemptyset(&mask);
    sigaddset(&mask, SIGUSR2);

    while(1){
        sigprocmask(SIG_SETMASK, &mask, NULL);
        printf("*****BACK-UP STARTS*****\n");
        mkdir("TEMP",0333);
        sleep(5);
        ftw(".",backup, 1);
        printf("*****BACK-UP ENDS*****\n");
        sigprocmask(SIG_UNBLOCK, &mask, NULL);

        alarm(30);

        pause();
    }
```

```
    exit(0);
}
```