

**Xilinx Zynq FPGA, TI DSP, MCU 기반의  
프로그래밍 및 회로 설계 전문가 과정  
#22**

**강사 : Innova Lee(이 상훈)**

**학생 : 김 시윤**

## 1.배운내용 복습.

### 과제 Linux ls 구현

(default를 잘못입력했을경우로 설정하고 아무것도 입력이 없을 때 ls 옵션을 주고싶은데, 스위치 케이스문에 적용이 안되었습니다.)

---헤더---

```
#include <stdio.h>
#include <unistd.h>
#include <string.h>
#include <fcntl.h>
#include <sys/types.h>
#include <dirent.h>
#include <sys/stat.h>
#include <pwd.h>
#include <grp.h>
#include <time.h>
```

---사용한 함수---

```
void recursive_dir(char *dname); // aR옵션 함수
void recursive_dir2(char *dname); //R옵션 함수
void mylR(char *dname); //lR 옵션 함수
void myaop(char *dname); //a 옵션 함수
void nonop(char *dname); //옵션 x 그냥 ls 함수
void loption(char *dname); // l 옵션 함수
void aloption(char *dname); // al 옵션 함수
void myalR(char *dname); // alR 옵션 함수
```

---flag check---

```
void check_flag(int flag)
{
    int tmp = flag | flag;
    int t= flag | (~(flag));

    switch(tmp & t)
    {
        case 1:

            printf("mys a option\n");
            myaop(".");
            break;
        case 2:
            printf("my l option\n");
            loption(".");
            break;
        case 3:
            printf("my al option\n");
            aloption(".");
            break;
        case 4:
            printf("mys R option\n");
            recursive_dir2(".");
            break;
        case 5:
            printf("mys aR option\n");
            recursive_dir(".");
            break;
        case 6:
            printf("my lR option\n");
```

```

        mylR(".");
        break;
case 7:
    printf("my alR option\n");
    myalR(".");
    break;
default :
    printf("nonoption\n");
    nonop(".");
    break;
    }
}

```

### flag 설명

main 에 있는 getopt 의 cmd 는 문자열 alR을 한번씩 무조건 돌게된다.  
 따라서 그중 하나 또는 두 개만 고를라면 딱 그 값만 지정해 줘야한다.  
 이걸 비트 계산을 이용하였다.

우리가 쓸 옵션은 총 a,l,R,al,aR,lR,aLR 7개의 옵션이다. 이 경우를 비트단위로 표현해보면 아래와 같다.

#### a옵션일 경우

0	0	1
R(MSB)	1	a(LSB)

#### R옵션일 경우

1	0	0
R(MSB)	1	a(LSB)

#### aR옵션일 경우

1	0	1
R(MSB)	1	a(LSB)

#### l 옵션일 경우

0	1	0
R(MSB)	1	a(LSB)

#### al 옵션일 경우

0	1	1
R(MSB)	1	a(LSB)

#### lR 옵션일 경우

1	1	0
R(MSB)	1	a(LSB)

alR옵션일 경우

1	1	1
R(MSB)	1	a(LSB)

t

1	1	1
2	1	0

이와 같은 7가지 경우를 만들어 주기 위해서는 한번씩 다 순회하며 값을 읽어주는 flag를 이용해야한다.

a와 l 이 메인에서 동시에 입력되면 a,l 이 읽힌다 이는 001 과 010이 들어온다는 것이다 , 이것을 or 연산 해주면 011이 된다.

이걸 t와 and 연산하면 011의 값만 스위치문을 통과해 들어온 al값을 읽어준다.

t= 111

### ---main---

```

int main(int argc, char **argv)
{
    int i;
    int flag;
    int s = 0,cmd;
    DIR *dp;
    struct dirent *p;
    char name[1024];
    while((cmd = getopt(argc, argv, "alR")) > 0)
    {
        switch(cmd)
        {
            case 'a':
                flag |= 1 << 0;
                //printf("a option\n");
                break;
            case 'l':
                flag |= 1 << 1;

```

```
        //printf("b option\n");
        break;
    case 'R':
        flag |= 1 << 2;
        //printf("c option\n");
        break;
    }
}
check_flag(flag);

return 0;
}
```

---ls -al 옵션함수---

```
void aloption(char *dname)
{
    struct stat buf;
    struct dirent *p;
    struct passwd *pw;
    struct group *gr;
    struct tm *tm;
    char ch;
    char perm[11] = "-----";
    char rwx[4] = "rwx";
    char sst[4] = "sst";
    int i;
    DIR *dp;

    dp = opendir(".");
    while(p= readdir(dp))
    {
        stat((p->d_name),&buf);
        if(S_ISDIR(buf.st_mode))
```

```
        perm[0]='d';
        if(S_ISREG(buf.st_mode))
            perm[0]='-';
        if(S_ISFIFO(buf.st_mode))
            perm[0]='p';
        if(S_ISSOCK(buf.st_mode))
            perm[0]='s';
        if(S_ISCHR(buf.st_mode))
            perm[0]='c';
        if(S_ISBLK(buf.st_mode))
            perm[0]='b';
        for(i=0;i<9;i++)
            if((buf.st_mode >> (8-i))&1)
                perm[i+1]=rwx[i%3];
        for(i=0;i<3;i++)
            if((buf.st_mode >> (11-i))&1)
                if(perm[(i+1)*3]!='-')
                    perm[(i+1)*3]=sst[i]^0x20;
        else
            perm[(i+1)*3]=sst[i];
        printf("%s  ",perm);
        printf("%ld  ",buf.st_nlink);
        pw=getpwuid(buf.st_uid);
        printf("%s  ",pw->pw_name);
        gr=getgrgid(buf.st_gid);
        printf("%s  ",gr->gr_name);
        printf("%ld  ",buf.st_size);
        tm=localtime(&buf.st_mtime);
        printf("%d - %02d - %02d  %02d : %02d :“
,tm->tm_year+1900,tm->tm_mon+1,tm->tm_mday,tm->tm_hour,tm->tm_min)
;

        printf("%-16s", p->d_name);
        if((i+1) %1 ==0)
```

```

        printf("\n");
        i++;
    }
    printf("\n");
    closedir(dp);
}

```

### ---ls -l 옵션함수---

```

void loption(char *dname)
{
    struct stat buf;
    struct dirent *p;
    struct passwd *pw;
    struct group *gr;
    struct tm *tm;
    char ch;
    char perm[11] = "-----";
    char rwx[4] = "rwx";
    char sst[4]= "sst";
    int i;
    DIR *dp;
    dp = opendir(".");
    while(p= readdir(dp))
    {
        if(p->d_name[0]=='.')
            continue;
        stat((p->d_name),&buf);
        if(S_ISDIR(buf.st_mode))
            perm[0]='d';
        if(S_ISREG(buf.st_mode))
            perm[0]='-';
        if(S_ISFIFO(buf.st_mode))
            perm[0]='p';
        if(S_ISSOCK(buf.st_mode))

```

```

            perm[0]='s';
            if(S_ISCHR(buf.st_mode))
                perm[0]='c';
            if(S_ISBLK(buf.st_mode))
                perm[0]='b';
            for(i=0;i<9;i++)
                if((buf.st_mode >> (8-i))&1)
                    perm[i+1]=rwx[i%3];
            for(i=0;i<3;i++)
                if((buf.st_mode >> (11-i))&1)
                    if(perm[(i+1)*3]!='-')
                        perm[(i+1)*3]=sst[i]^0x20;
            else
                perm[(i+1)*3]=sst[i];
            printf("%s  ",perm);
            printf("%ld  ",buf.st_nlink);
            pw=getpwuid(buf.st_uid);
            printf("%s  ",pw->pw_name);
            gr=getgrgid(buf.st_gid);
            printf("%s  ",gr->gr_name);
            printf("%ld  ",buf.st_size);
            tm=localtime(&buf.st_mtime);
            printf("%d  -  %02d  -  %02d  %02d  :  %02d  ",tm->tm_year+1900,tm->tm_mon+1,tm->tm_mday,tm->tm_hour,tm->tm_min);

            printf("%-16s", p->d_name);
            if((i+1) %1 ==0)
                printf("\n");
                i++;
            }
            printf("\n");

    closedir(dp);

```

```
}
```

### ---ls non 옵션함수---

```
void nonop(char *dname)
{
    int s = 0,cmd;
    DIR *dp;
    struct dirent *p;

    dp=opendir(".");
    while(p=readdir(dp))
    {
        if(p->d_name[0]!='.')
            continue;

        printf("%-16s",p->d_name);
        if((s+1) % 5 ==0)
            printf("\n");
        s++;
    }
    printf("\n");
    closedir(dp);
}
```

### --- ls -a 옵션---

```
void myaop(char *dname)
{
    int s = 0,cmd;
    DIR *dp;
    struct dirent *p;
```

```
dp=opendir(".");
    while(p=readdir(dp))
    {
        printf("%-16s",p->d_name);
        if((s+1) % 5 ==0)
            printf("\n");
        s++;
    }
    printf("\n");
    closedir(dp);
}
```

### ---ls -aR 옵션---

```
void recursive_dir(char *dname)
{
    struct dirent *p;
    struct stat buf;
    DIR *dp;

    int i;
    chdir(dname);
    dp = opendir(".");
    printf("\t%s : \n",dname);
    while(p= readdir(dp)){
        printf("%s\t",p->d_name);

        if((i+1) %3 ==0)
            printf("\n");
            i++;
    }

    printf("\n");
    rewinddir(dp);
    while(p=readdir(dp))
```

```

{
    stat(p->d_name,&buf);
    if(S_ISDIR(buf.st_mode))
    if(strcmp(p->d_name,".")&&strcmp(p->d_name,".."))
        recursive_dir(p->d_name);
}
chdir("..");
closedir(dp);
}

```

### ---ls -R 옵션---

```

void recursive_dir2(char *dname)
{
    struct dirent *p;
    struct stat buf;
    DIR *dp;
    int i;
    chdir(dname);
    dp = opendir(".");
    printf("\t%s : \n",dname);
    while(p= readdir(dp))
    {if(p->d_name[0]!='.')
        continue;

        printf("%s\t",p->d_name);
        if((i+1) %3 ==0)
            printf("\n");
            i++;
    }
    printf("\n");
    rewinddir(dp);
    while(p=readdir(dp))
    {

```

```

        stat(p->d_name,&buf);
        if(S_ISDIR(buf.st_mode))
        if(strcmp(p->d_name,".")&&strcmp(p->d_name,".."))
            recursive_dir2(p->d_name);
        }
        chdir("..");
        closedir(dp);
    }
}

```

### ---ls -lR 옵션---

```

void mylR(char *dname)
{
    struct dirent *p;
    struct stat buf;
    DIR *dp;
    int i;
    struct passwd *pw;
    struct group *gr;
    struct tm *tm;
    char ch;
    char perm[11]="-----";
    char rwx[4]="rwx";
    char sst[4]="sst";

    chdir(dname);
    dp = opendir(".");
    printf("\t%s : \n",dname);
    while(p= readdir(dp))
    {if(p->d_name[0]!='.')
        continue;
        stat((p->d_name),&buf);

```

```

if(S_ISDIR(buf.st_mode))
perm[0]='d';
if(S_ISREG(buf.st_mode))
perm[0]='-';
if(S_ISFIFO(buf.st_mode))
perm[0]='p';
if(S_ISSOCK(buf.st_mode))
perm[0]='s';
if(S_ISCHR(buf.st_mode))
perm[0]='c';
if(S_ISBLK(buf.st_mode))
perm[0]='b';

for(i=0;i<9;i++)
if((buf.st_mode >> (8-i))&1)
perm[i+1]=rwx[i%3];
for(i=0;i<3;i++)
if((buf.st_mode >> (11-i))&1)
if(perm[(i+1)*3]=='-')
perm[(i+1)*3]=sst[i]^0x20;
else
perm[(i+1)*3]=sst[i];

printf("%s  ",perm);
printf("%ld  ",buf.st_nlink);
pw=getpwuid(buf.st_uid);
printf("%s  ",pw->pw_name);
gr=getgrgid(buf.st_gid);
printf("%s  ",gr->gr_name);
printf("%ld  ",buf.st_size);
tm=localtime(&buf.st_mtime);
printf("%d      -      %02d      -      %02d      %02d      :      %02d
:",tm->tm_year+1900,tm->tm_mon+1,tm->tm_mday,tm->tm_hour,tm->tm_mi

```

```

n);

printf("%-16s\t",p->d_name);
if((i+1) %1 ==0)
printf("\n");
i++;
}
printf("\n");
rewinddir(dp);
while(p=readdir(dp))
{

stat(p->d_name,&buf);
if(S_ISDIR(buf.st_mode))
if(strcmp(p->d_name,".")&&strcmp(p->d_name,".."))
mylR(p->d_name);
}
chdir("..");
closedir(dp);
}

```

### ---ls -alR 옵션---

```

void myalR(char *dname)
{
    struct dirent *p;
    struct stat buf;
    DIR *dp;
    int i;
    struct passwd *pw;
    struct group *gr;
    struct tm *tm;
    char ch;
    char perm[11]="-----";
    char rwx[4]="rwx";

```



```

char sst[4]="sst";

chdir(dname);
dp = opendir(".");
printf("\t%s : \n",dname);
while(p= readdir(dp))
{

stat((p->d_name),&buf);

if(S_ISDIR(buf.st_mode))
perm[0]='d';
if(S_ISREG(buf.st_mode))
perm[0]='-';
if(S_ISFIFO(buf.st_mode))
perm[0]='p';
if(S_ISSOCK(buf.st_mode))
perm[0]='s';
if(S_ISCHR(buf.st_mode))
perm[0]='c';
if(S_ISBLK(buf.st_mode))
perm[0]='b';

for(i=0;i<9;i++)
if((buf.st_mode >> (8-i))&1)
perm[i+1]=rwx[i%3];
for(i=0;i<3;i++)
if((buf.st_mode >> (11-i))&1)
if(perm[(i+1)*3]=='-')
perm[(i+1)*3]=sst[i]^0x20;
else
perm[(i+1)*3]=sst[i];

```

```

printf("%s ",perm);
printf("%ld ",buf.st_nlink);
pw=getpwuid(buf.st_uid);
printf("%s ",pw->pw_name);
gr=getgrgid(buf.st_gid);
printf("%s ",gr->gr_name);
printf("%ld ",buf.st_size);
tm=localtime(&buf.st_mtime);
printf("%d - %02d - %02d %02d : %02d
:",tm->tm_year+1900,tm->tm_mon+1,tm->tm_mday,tm->tm_hour,tm->tm_min);

printf("%-16s\t",p->d_name);
if((i+1) %1 ==0)
printf("\n");
i++;
}
printf("\n");
rewinddir(dp);
while(p=readdir(dp))
{

stat(p->d_name,&buf);
if(S_ISDIR(buf.st_mode))
if(strcmp(p->d_name,".")&&strcmp(p->d_name,".."))
myalR(p->d_name);
}
chdir("..");
closedir(dp);
}

```

## 과제 Naver Cafe 문제 (미완성 ..)

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

int main(int argc, char **argv)
{
    int fd;
    int nread;
    char buf[1024];
    int i=0,j=0;
    int num[32]={0};
    int res=0;
    int cnt=0;

    fd = open("a.txt",O_RDONLY);

    while(nread = read(fd,buf,sizeof(buf)))
    {
        for(i=0;buf[i];i++){

            if(buf[i]==10){

                num[j+1]=i;
                printf("num[%d]= %d  \n",j,num[j]);
                j++;

                printf("\n");
            }
        }
    }
}
```

```
continue;}

//printf("buf[%d]=%d  ",i,buf[i]);
    }

    for(j=(num[cnt]);j<num[cnt+1];j++)
    {
        printf("j = %d\n",j);
        res += buf[j];
        printf("%d  \n",buf[j]);
        //printf("Res = %d  \n",res);
    }

    printf("res = %d\n",res);
    cnt++;

    }
    return 0;
}
```

버퍼에 저장되어있는 값을 확인하니 아스키코드였다 . 그래서 버퍼에 있는 아스키 코드에서 \n가 나오는 부분을 num에 뽑아냈다.

그리고 넘에 저장된 값으로 범위를 정해서 더하는 작업을하고 res 에 넣어주려했는데 원하는 결과값이 계속 나오지않아 결국 실패 하였다 ..

```

siyun@siyun-CR62-6M:~/my_proj/hw$ vi test.c
siyun@siyun-CR62-6M:~/my_proj/hw$ gcc test.c
siyun@siyun-CR62-6M:~/my_proj/hw$ ./a.out
num[0]= 0

num[1]= 5

num[2]= 12

num[3]= 18

num[4]= 24

num[5]= 35

num[6]= 42

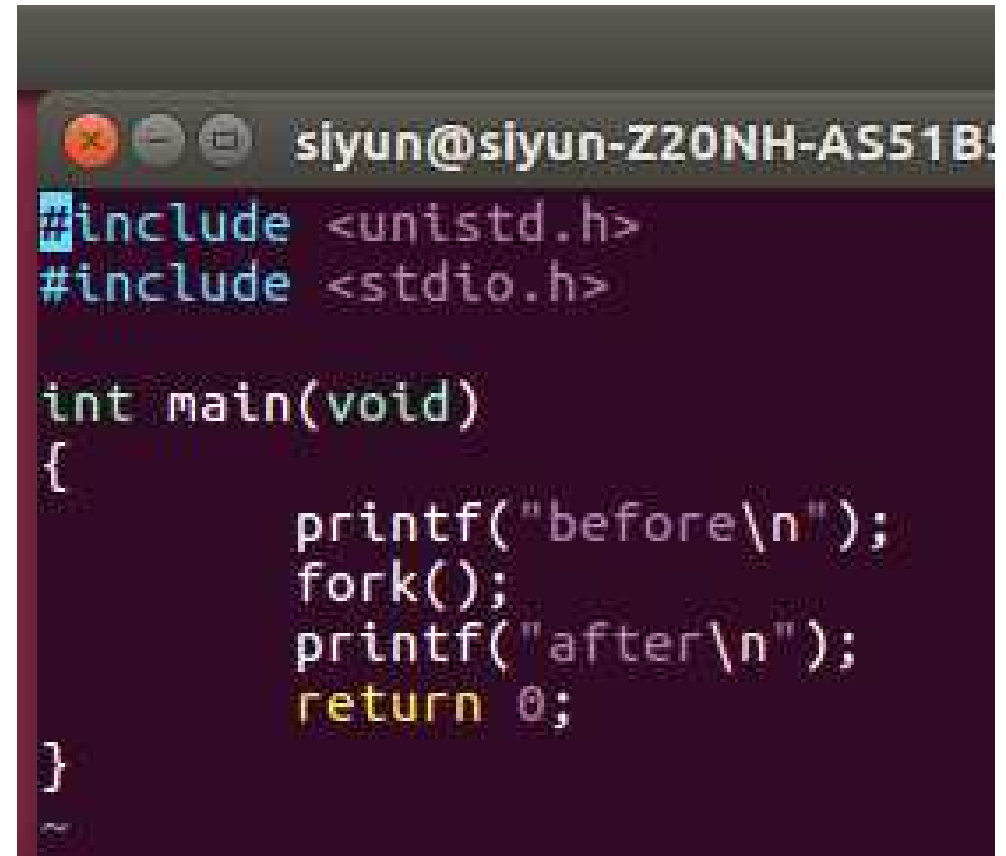
j = 0
97
j = 1
112
j = 2
112
j = 3
108
j = 4
101
res = 530
siyun@siyun-CR62-6M:~/my_proj/hw$

```

변수 하나를 더사용해서 res에 더한값을 하나씩 저장하고  
그저정한 값으로 파일생성해서 거기에 적을라했는데 ..  
생각처럼 써지지 않았다 ...

### ---fork---

포크는 fork(); 형식으로 쓰고 포크가 선언되면 fork 의 밑에 부분을 복사하여 프로세스를 하나 더 생성한다. fork 의 리턴값은 자식의 프로세스의 PID이다.



```

siyun@siyun-Z20NH-AS51B$ vi test.c
#include <unistd.h>
#include <stdio.h>

int main(void)
{
    printf("before\n");
    fork();
    printf("after\n");
    return 0;
}

```

//fork after 가 두 개 나온다 이는 fork 가 복사하여 프로세스가 하나 더 생성되고 2번째 프로세스와 첫 번째 프로세스가 printf를 한 것을 알 수있다.

### ---fork pid---

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

int main(void)
{
    pid_t pid;
    pid = fork();
    if(pid > 0)
        printf("parent : pid = %d , cpid = %d \n",getpid(),pid);
    else if(pid ==0)
        printf("child : pid = %d, cpid = %d\n",getpid(),pid);
    else
    {
        perror("fork()");
        exit(-1);
    }
    return 0;
}
```

/\* fork 의 리턴값은 자식의 pid이다.

그러면 fork의 위의 소스코드로 확인해보았다.

pid가 0보다 크다는 뜻은 부모가 돌고있다는 뜻이고 pid가 0과 같다는 소리는 부모가 없다는소리 즉 자식이 돈다는 소리이다. \*/

```
siyun@siyun-Z20NH-AS51B5U:~/my_proj/class22$ gcc for3.c
siyun@siyun-Z20NH-AS51B5U:~/my_proj/class22$ ./a.out
parent : pid = 4576 , cpid = 4577
child : pid = 4577, cpid = 0
siyun@siyun-Z20NH-AS51B5U:~/my_proj/class22$
```

### ---sleep---

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

```
int main(void)
{
    pid_t pid;
    if((pid = fork())>0)
        sleep(1000);
    else if(pid ==0)
        ;
    else
    {
        perror("fork()");
        exit(-1);
    }
    return 0;
}
```

//자식이 생명을 다해서 죽었는데 부모가 자고있어서 시신을 수습을 못하는 경우다.

```
siyun@siyun-Z20NH-AS51B5U:~/my_proj/class22$ ps -ef | grep a.out
siyun      6332   1677    0 15:40 pts/18    00:00:00 ./a.out
siyun      6348   1677    0 15:42 pts/18    00:00:00 ./a.out
siyun      6366   1677    0 15:44 pts/18    00:00:00 ./a.out
siyun      6387   1677    0 15:47 pts/18    00:00:00 ./a.out
siyun      6403   6247    0 15:48 pts/18    00:00:00 ./a.out
siyun      6404   6403    0 15:48 pts/18    00:00:00 [a.out] <defunct>
siyun      6406   5901    0 15:48 pts/2     00:00:00 grep --color=auto a.out
siyun@siyun-Z20NH-AS51B5U:~/my_proj/class22$
```

이 경우 자식 프로세스의 defunct 가 생기는데 부모가 자고 일어나서 처리하면 없어진다. sleep 시간만큼 defunct가 머문다

### ---Context Switching---

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

int main(void)
{
    pid_t pid;
    int i;
    pid = fork();
    if(pid>0)
    {
        while(1)
        {
            for(i =0; i<26; i++)
            {
                printf("%c",i+'A');
                fflush(stdout);
            }
        }
    }
    else if(pid ==0)
    {
        while(1)
        {
            for(i=0;i<26;i++)
            {
                printf("%c",i+'a');
                fflush(stdout);
            }
        }
    }
    else
    {
        perror("fork()");
        exit(-1);
    }
    printf("\n");
    return 0;
}
```

[illegible]

---ddd000---

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

int main(void)
{
    pid_t pid;
    int status;
    if((pid=fork())>0)
    {
        wait(&status);
        printf("status :%d\n",status);//exit 한 데이터를 갖고있다 8bit s
ift 하면 알수있다.
    }
    else if(pid ==0)
        abort();//
    else
    {
        perror("fork()");
}wait4.c" 26L, 41SC
```

exit 된 데이터를 갖고있어서 shift 8 하면 볼수 있다.



```
siyun@siyun-Z20NH-AS51B5U:~/my_proj/class22$ vi wait4.c
siyun@siyun-Z20NH-AS51B5U:~/my_proj/class22$ gcc wait4.c
siyun@siyun-Z20NH-AS51B5U:~/my_proj/class22$ ./a.out
status :0x6
siyun@siyun-Z20NH-AS51B5U:~/my_proj/class22$
```

status 함수를 사용하여 exit 한 위치를 확인할 수 있다.

종료한 내용을 확인하고싶으면 stty 정보에 6번을 보면 알수 있다.

## 부모자식 관계도

