

시스템 프로그래밍

3차 프로그래밍 과제



SINCE 1947

서경대학교

SEOKYEONG UNIVERSITY



시스템 프로그래밍 3차

-2019301005 김경민-

목차

1.과제1

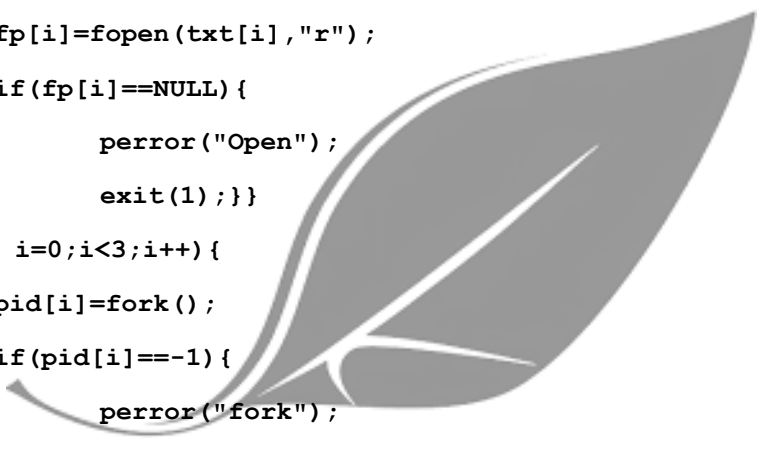
2.과제2



1. 과제1

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

int main() {
    pid_t pid[3];
    int status, endpid;
    char *txt[3]={"t1.txt", "t2.txt", "t3.txt"};
    FILE *fp[3];
    for(int i=0;i<3;i++){
        fp[i]=fopen(txt[i], "r");
        if(fp[i]==NULL) {
            perror("Open");
            exit(1);}
    for(int i=0;i<3;i++){
        pid[i]=fork();
        if(pid[i]==-1){
            perror("fork");
            exit(1);}
        else if(pid[i]==0){
            while(1){
                char buf=fgetc(fp[i]);
                if(buf == EOF)
                    break;
                else if(buf != '\n')
                    putchar(buf);}
            fclose(fp[i]);
            exit(status);
        }
        endpid = waitpid(pid[i], &status, 0);
        write(1, "\n", 1);}
    puts("parent end");
}
```



t1.txt ,t2.txt,t3.txt를 생성하고 그 안의 문자열을 받아서 처리하는 식으로 동작
FILE *fp 와 fork에 사용될 pid 배열 생성 후 차례대로 fopen
Fork 선언 후 (pid==0)일 때 문자 변수 Buf가 파일의 끝일때까지 무한루프
buf에 문자를 하나씩 읽으면서 삽입 buf가 개행문자가 아닐시 출력
완료 후 fclose로 fp 닫고 exit로 자식 프로세스 종료
endpid에 waitpid로 pid를 직접 지정하며 종료를 기다린 후 개행문자 출력

```
kkk@kkk-VirtualBox:~$ ./fork
hellothisisfirsttxt
hellothisissecondtxt
hellothisisthirdtxt
parent end
kkk@kkk-VirtualBox:~$
```

결과

```
hello
this
is
first
txt
```

t1. txt

```
hello
this
is
second
txt
```

t2.txt

```
hello
this
is
third
txt
```

t3.txt

2. 과제2

2-1. 첫 시도 exec.c

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

int main(int argc ,char **argv){
    int *arr;
    pid_t pid;
    int status;
    int n;
    pid=fork();
    if(pid==-1){
        perror("fork");
        exit(1);}
    else if(pid==0){
        puts("child");
        execvp("./prime",argv);}
    wait(&status);
    if(WIFEXITED(status)){puts("");
        arr = WEXITSTATUS(status);
        for(int i=0;i<argc-1;i++){
            printf("%d",arr[i]);
        }
        printf("parent\n"); }}
```

부모가 직접 자식에게 숫자들을 넘겨야한다는 문제를 읽고 가장 먼저 떠오른 방식
이번엔 인자로 숫자들을 받아서 execvp로 argv자체를 자식에게 넘겨서
정수로 바꾼 후에 소수만을 판별하여 다시 재구성 후 부모에게 전달 그리고 출력
fork후에 자식 프로세스라면 execvp로 prime을 실행 하는데 argv를 전달
그리고 prime은 int*를 return하려고 했지만 prime내의 지역변수인 배열을
return 하는것에 어려움을 느낌

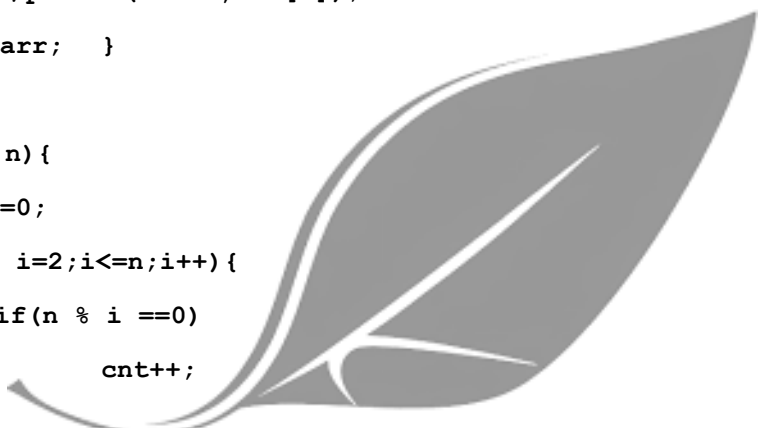
prime.c

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

int Prime(int n);

int* main(int argc, char**argv){
    int arr[argc-1],n=0;
    for(int i=1;i<argc;i++){
        if(Prime(atoi(argv[i]))){
            arr[n] = atoi(argv[i]);
            n++;}}
    //for(int i=0;i<n;i++)
        //printf("%d ",arr[i]);
    return arr; }

int Prime(int n){
    int cnt=0;
    for(int i=2;i<=n;i++){
        if(n % i ==0)
            cnt++;
    }
    if(cnt==1)
        return 1;
    else
        return 0;
}
```



위의 `execvp`를 위한 자식 프로세스인 `prime.c` `argv`를 인수로 받아서
최대 배열의 개수는 `argv-1`(0번째는 `./exec`) 이니 `arr[argc-1]` 배열 생성 후
반복문으로 `argv`하나씩 숫자로 변환 후 `Prime` 함수로 소수 판별 후 `arr`에 삽입
자식프로세스에서 출력하면 잘 되지만 `int *` 형 `main`함수에 `return arr;` 해도 불가능

2-2. fork2.c

```
int Prime(int n);

int main(int argc, char **argv) {
    int arr[argc-1]; pid_t pid; int status; int n;
    pid=fork();
    if(pid==-1) {
        perror("fork");
        exit(1);
    } else if(pid==0) {
        puts("child");
        for(int i=1; i<argc; i++) {
            if(Prime(atoi(argv[i]))) {
                arr[n] = atoi(argv[i]);
                n++;
            }
        }
        for(int i=0; i<n; i++)
            printf("%d ", arr[i]);
        return arr;
    }
    wait(&status);
    if(WIFEXITED(status)) {
        //arr = WEXITSTATUS(status);
        for(int i=0; i<argc-1; i++) {
            printf("%d", arr[i]);
        }
        printf("parent\n");
    }
}
```

위의 exec.c의 fork버전 (prime함수 생략) 마찬가지로 arr return이 안됨
WEXITSTATUS(status)로 종료된 자식 프로세스의 return값 반환도 실패

과제 2의 문제는

1. 부모 프로세스가 자식에게 숫자들을 넘겨준다
 - 1-1. 부모가 인자를 받아서 직접 넘겨준다(exec)
 - 1-2. 부모가 받은 인자를 바로 사용한다(fork)
 - 1-3. 부모에서 숫자가 적혀있는 txt파일을 읽어 배열을 생성 후 넘김(read)
 - 1-4. 부모가 txt파일을 file descriptor로 오픈(open)
2. 자식 프로세스는 받은 숫자들중 소수를 추출하여 return 한다
 - 2-1. Prime.c가 argv를 받아 prime함수로 추출 후 int* arr return(X)
 - 2-2. 같은 argv를 사용하여 prime함수로 추출 후 arr로 return(X)
 - 2-3. 배열을 받아서 prime함수로 추출 후 arr return(X)
 - 2-4. 자식이 open된 파일을 read하여 소수만 추출 후 write (write)
3. 부모 프로세스가 자식이 넘겨준 소수를 출력
 - 3-1. 받지못함
 - 3-2. 받지못함
 - 3-3. 받지못함
 - 3-4. 다시 자식이 write한 파일을 read하여 출력(read)

문제가 부모로부터 받고 자식이 넘겨준 이라고 적혀있기 때문에

4번 경로는 넘겨주지 않고 txt 파일을 read & write만 있어서 제외

1,2,3 방법은 시도해봤지만 return 시도에 전부 막혔음

배열을 extern으로 설정하는것도 시도해봤지만 실패

```
kkm@kkm-VirtualBox:~$ ./exec 1 2 3 4 5 6 7 8
child
2 3 5 7
세그멘테이션 오류 (코어 덤프됨)
kkm@kkm-VirtualBox:~$
```

감사합니다.