

# 시스템 프로그래밍

2021563029 박상민

깃허브 주소 : <https://github.com/qkrtdals962/ks-WebpgmSys.git>

# 프로젝트 : C언어로 리눅스 명령어 구현하기

basename, basename2, cat, chgrp, chmod, chown, clear, cmp, cp, date, df, dirname, du, echo, env, factor, find, free, grep, head, hostname, id, kill, ln, ls, mkdir, mv, nl, od, od2, ps, pwd, rev, rev2, rm, rmdir, sleep, sort, sum, tail, tee, touch, tr, uname, uniq, uptime, wc, who, whoami, yes – 총 50개

```

-rwxrwxr-x 1 sangmin sangmin 16832 Jun 5 16:37 basename*
-rw-r--r-- 1 root root 345 Jun 5 16:29 basename.c
-rwxrwxr-x 1 sangmin sangmin 16920 Jun 5 16:37 basename2*
-rw-r--r-- 1 root root 694 Jun 5 16:31 basename2.c
-rwxrwxr-x 1 sangmin sangmin 16920 Jun 5 16:17 cat*
-rw-r--r-- 1 sangmin sangmin 531 Jun 5 16:13 cat.c
-rwxrwxr-x 1 sangmin sangmin 16920 Jun 5 16:37 chgrp*
-rw-r--r-- 1 sangmin sangmin 559 Jun 5 16:13 chgrp.c
-rwxrwxr-x 1 sangmin sangmin 16872 Jun 5 16:37 chmod*
-rw-r--r-- 1 sangmin sangmin 389 Jun 5 16:14 chmod.c
-rwxrwxr-x 1 sangmin sangmin 16968 Jun 5 16:37 chown*
-rw-r--r-- 1 root root 775 Jun 5 16:19 chown.c
-rwxrwxr-x 1 sangmin sangmin 16696 Jun 5 16:37 clear*
-rw-r--r-- 1 root root 109 Jun 5 16:20 clear.c
-rwxrwxr-x 1 sangmin sangmin 17000 Jun 5 16:37 cmp*
-rw-r--r-- 1 root root 1131 Jun 5 16:32 cmp.c
-rwxrwxr-x 1 sangmin sangmin 16968 Jun 5 16:37 cp*
-rw-r--r-- 1 root root 667 Jun 5 16:20 cp.c
-rwxrwxr-x 1 sangmin sangmin 16824 Jun 5 16:37 date*
-rw-r--r-- 1 root root 382 Jun 5 16:22 date.c
-rwxrwxr-x 1 sangmin sangmin 16840 Jun 5 16:37 df*
-rw-r--r-- 1 root root 649 Jun 5 16:26 df.c
-rwxrwxr-x 1 sangmin sangmin 16920 Jun 5 17:00 dirname*
-rw-r--r-- 1 root root 500 Jun 5 16:52 dirname.c
-rwxrwxr-x 1 sangmin sangmin 17064 Jun 5 16:37 du*
-rw-r--r-- 1 root root 550 Jun 5 16:27 du.c
-rwxrwxr-x 1 sangmin sangmin 16744 Jun 5 16:37 echo*
-rw-r--r-- 1 root root 212 Jun 5 16:22 echo.c
-rwxrwxr-x 1 sangmin sangmin 16768 Jun 5 16:37 env*
-rw-r--r-- 1 root root 169 Jun 5 16:28 env.c
-rwxrwxr-x 1 sangmin sangmin 16872 Jun 5 16:37 factor*
-rw-r--r-- 1 root root 737 Jun 5 16:32 factor.c
-rwxrwxr-x 1 sangmin sangmin 17192 Jun 5 16:37 find*
-rw-r--r-- 1 root root 1000 Jun 5 16:25 find.c
-rwxrwxr-x 1 sangmin sangmin 16880 Jun 5 16:37 free*
-rw-r--r-- 1 root root 469 Jun 5 16:28 free.c
-rwxrwxr-x 1 sangmin sangmin 17096 Jun 5 16:37 grep*
-rw-r--r-- 1 root root 590 Jun 5 16:25 grep.c
-rwxrwxr-x 1 sangmin sangmin 17144 Jun 5 16:37 head*
-rw-r--r-- 1 root root 859 Jun 5 16:24 head.c
-rwxrwxr-x 1 sangmin sangmin 16848 Jun 5 16:37 hostname*
-rw-r--r-- 1 root root 270 Jun 5 16:26 hostname.c
-rwxrwxr-x 1 sangmin sangmin 17152 Jun 5 17:00 id*
-rw-r--r-- 1 root root 800 Jun 5 16:52 id.c
-rwxrwxr-x 1 sangmin sangmin 16872 Jun 5 16:37 kill*
-rw-r--r-- 1 root root 652 Jun 5 16:27 kill.c
-rwxrwxr-x 1 sangmin sangmin 17160 Jun 5 16:37 wc*
-rw-r--r-- 1 root root 856 Jun 5 16:25 wc.c
-rwxrwxr-x 1 sangmin sangmin 16936 Jun 5 16:37 who*
-rw-r--r-- 1 root root 542 Jun 5 16:28 who.c
-rwxrwxr-x 1 sangmin sangmin 16832 Jun 5 16:37 whoami*
-rw-r--r-- 1 root root 276 Jun 5 16:23 whoami.c
-rwxrwxr-x 1 sangmin sangmin 16696 Jun 5 16:37 yes*
-rw-r--r-- 1 root root 198 Jun 5 16:30 yes.c

```

```

-rwxrwxr-x 1 sangmin sangmin 16896 Jun 5 16:37 ln*
-rw-r--r-- 1 root root 768 Jun 5 16:23 ln.c
-rwxrwxr-x 1 sangmin sangmin 16920 Jun 5 16:37 ls*
-rw-r--r-- 1 root root 443 Jun 5 16:22 ls.c
-rwxrwxr-x 1 sangmin sangmin 16832 Jun 5 16:37 mkdir*
-rw-r--r-- 1 root root 335 Jun 5 16:21 mkdir.c
-rwxrwxr-x 1 sangmin sangmin 16824 Jun 5 16:37 mv*
-rw-r--r-- 1 root root 319 Jun 5 16:20 mv.c
-rwxrwxr-x 1 sangmin sangmin 17056 Jun 5 16:37 nl*
-rw-r--r-- 1 root root 590 Jun 5 16:31 nl.c
-rwxrwxr-x 1 sangmin sangmin 17136 Jun 5 16:37 od*
-rw-r--r-- 1 root root 968 Jun 5 16:32 od.c
-rwxrwxr-x 1 sangmin sangmin 17136 Jun 5 16:37 od2*
-rw-r--r-- 1 root root 966 Jun 5 16:33 od2.c
-rwxrwxr-x 1 sangmin sangmin 17208 Jun 5 16:37 ps*
-rw-r--r-- 1 root root 1007 Jun 5 16:27 ps.c
-rwxrwxr-x 1 sangmin sangmin 16832 Jun 5 16:37 pwd*
-rw-r--r-- 1 root root 274 Jun 5 16:22 pwd.c
-rwxrwxr-x 1 sangmin sangmin 17096 Jun 5 16:37 rev*
-rw-r--r-- 1 root root 794 Jun 5 16:31 rev.c
-rwxrwxr-x 1 sangmin sangmin 17256 Jun 5 16:37 rev2*
-rw-r--r-- 1 root root 859 Jun 5 16:33 rev2.c
-rwxrwxr-x 1 sangmin sangmin 16824 Jun 5 16:37 rm*
-rw-r--r-- 1 root root 340 Jun 5 16:21 rm.c
-rwxrwxr-x 1 sangmin sangmin 16832 Jun 5 16:37 rmdir*
-rw-r--r-- 1 root root 327 Jun 5 16:21 rmdir.c
-rwxrwxr-x 1 sangmin sangmin 16824 Jun 5 16:37 sleep*
-rw-r--r-- 1 root root 384 Jun 5 16:25 sleep.c
-rwxrwxr-x 1 sangmin sangmin 17344 Jun 5 16:37 sort*
-rw-r--r-- 1 root root 1206 Jun 5 16:29 sort.c
-rwxrwxr-x 1 sangmin sangmin 16824 Jun 5 16:37 sum*
-rw-r--r-- 1 root root 346 Jun 5 16:32 sum.c
-rwxrwxr-x 1 sangmin sangmin 17216 Jun 5 16:37 tail*
-rw-r--r-- 1 root root 1256 Jun 5 16:24 tail.c
-rwxrwxr-x 1 sangmin sangmin 17048 Jun 5 17:00 tee*
-rw-r--r-- 1 root root 737 Jun 5 16:53 tee.c
-rwxrwxr-x 1 sangmin sangmin 16992 Jun 5 16:37 touch*
-rw-r--r-- 1 root root 673 Jun 5 16:25 touch.c
-rwxrwxr-x 1 sangmin sangmin 16968 Jun 5 16:37 tr*
-rw-r--r-- 1 root root 734 Jun 5 16:31 tr.c
-rwxrwxr-x 1 sangmin sangmin 16832 Jun 5 16:37 uname*
-rw-r--r-- 1 root root 225 Jun 5 16:26 uname.c
-rwxrwxr-x 1 sangmin sangmin 17184 Jun 5 16:37 uniq*
-rw-r--r-- 1 root root 690 Jun 5 16:29 uniq.c
-rwxrwxr-x 1 sangmin sangmin 16840 Jun 5 16:37 uptime*
-rw-r--r-- 1 root root 566 Jun 5 16:28 uptime.c

```

## sum 명령어

사용법: ./sum <숫자1> <숫자2> ...

기능: 전달된 정수 인자들의 합을  
계산해 출력합니다.

```
sangmin@DESKTOP-V3KNJ2P:~/project$ ./sum
사용법: ./sum <숫자1> [숫자2 ...]
sangmin@DESKTOP-V3KNJ2P:~/project$ ./sum 1 2
3
sangmin@DESKTOP-V3KNJ2P:~/project$ ./sum 1000 1000
2000
sangmin@DESKTOP-V3KNJ2P:~/project$
```

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

int main(int argc, char *argv[]) {
    if (argc < 2) {
        fprintf(stderr, "사용법: ./sum <숫자1> [숫자2 ...]\n");
        return 1;
    }

    long total = 0;
    for (int i = 1; i < argc; i++) {
        total += atol(argv[i]);
    }
    printf("%ld\n", total);
    return 0;
}
```

## uptime 명령어

사용법: ./uptime

기능: 시스템 부팅 이후 경과된 시간을 일.  
시:분:초 형식으로 표시합니다.

```
sangmin@DESKTOP-V3KNJ2P:~/project$ ./uptime
up 01:19:28
sangmin@DESKTOP-V3KNJ2P:~/project$ ./uptime
up 01:19:30
sangmin@DESKTOP-V3KNJ2P:~/project$ ./uptime
up 01:19:31
sangmin@DESKTOP-V3KNJ2P:~/project$
```

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/sysinfo.h>

int main(void) {
    struct sysinfo info;
    if (sysinfo(&info) != 0) {
        perror("sysinfo 실패");
        return 1;
    }

    long total_seconds = info.uptime;
    int days = total_seconds / 86400;
    int hours = (total_seconds % 86400) / 3600;
    int minutes = (total_seconds % 3600) / 60;
    int seconds = total_seconds % 60;

    printf("up ");
    if (days > 0) printf("%d days, ", days);
    printf("%02d:%02d:%02d\n", hours, minutes, seconds);
    return 0;
}
```

# 깃허브 정리 0314~0530

History for [ks-WebpgmSys](#) / [software](#) / README.md on [main](#)

Create README.md

qkrtdals962 authored on Mar 27

End of commit history for this file

1주 늦음

Commits on Mar 27, 2025

Update README.md

qkrtdals962 authored on Mar 27

Update README.md

qkrtdals962 authored on Mar 27

Create README.md

qkrtdals962 authored on Mar 27

Commits on Mar 28, 2025

Update README.md

qkrtdals962 authored on Mar 28

Update README.md

qkrtdals962 authored on Mar 28

Update README.md

qkrtdals962 authored on Mar 28

Update README.md

qkrtdals962 authored on Mar 28

Update README.md

qkrtdals962 authored on Mar 28

Update README.md

qkrtdals962 authored on Mar 28

Update README.md

qkrtdals962 authored on Mar 28

Update README.md

qkrtdals962 authored on Mar 28

Create README.md

qkrtdals962 authored on Mar 28

History for [ks-WebpgmSys](#) / [0404](#) on [main](#)

Commits on Apr 4, 2025

Update README.md

qkrtdals962 authored on Apr 4

0404제출

qkrtdals962 authored on Apr 4

Create README.md

qkrtdals962 authored on Apr 4

End of commit history for this file

Commits on Apr 11, 2025

Create README.md

qkrtdals962 authored on Apr 11

Commits on May 2, 2025

Create README.md

qkrtdals962 authored on May 2

End of commit history for this file

2주 늦음

Commits on May 9, 2025

Create README.md

qkrtdals962 authored last month

End of commit history for this file

1주 늦음

Commits on May 9, 2025

Create README.md

qkrtdals962 authored last month

End of commit history for this file

History for [ks-WebpgmSys](#) / [0516](#) / README.md

Commits on May 23, 2025

Create README.md

qkrtdals962 authored 2 weeks ago

End of commit history for this file

1주 늦음

History for [ks-WebpgmSys](#) / [0523](#) / README.md on [main](#)

Commits on Jun 4, 2025

Update README.md

qkrtdals962 authored 15 hours ago

Commits on May 23, 2025

Create README.md

qkrtdals962 authored 2 weeks ago

End of commit history for this file

History for [ks-WebpgmSys](#) / [0530](#) on [main](#)

Commits on Jun 5, 2025

Create README.md

qkrtdals962 authored now

# 0314 1주 늦음

Name	Last commit message	Last commit date
..		
app	0314수업	3 months ago
system	0314수업	3 months ago
README.md	Update README.md	3 months ago
software tree.png	tree구조 결과	3 months ago

README.md

## software폴더에 하위폴더를 만들고 tree로 표현하기


### 사용한 명령어

- `mkdir`
  - 디렉토리(폴더) 생성
- `cd` 디렉토리이름, ..
  - 해당 디렉토리로 이동
  - 부모 디렉토리로 이동
- `cat`
  - `cat > 파일이름` 명령어를 이용하여 새로운 txt를 만들고 바로 Ctrl+D를 사용하여 빈 txt를 생성
- `tree`
  - `tree`명령어를 사용하여 tree구조 표현

### 결과

 [software 폴더에 올려놨습니다.](#)

Create README.md

 qkrtdals962 authored on Mar 27




End of commit history for this file

# 0321

Name	Last commit message	Last commit date
..		
README.md	Update README.md	3 months ago
a.out	0321수업	3 months ago
hello.c	0321수업	3 months ago
mission.c	0321수업	3 months ago
mission.png	README.md 참고용	3 months ago
README.md		

## 문서 작성및 코드 작성

### 문서 작성

- **cat > 파일**
    - 표준입력 내용을 모두 파일에 저장한다. 파일이 없으면 새로 만듦
  - **touch 파일**
    - 파일 크기가 0인 이름만 있는 빈 파일을 만들어줌
  - **gcc example.c**
    - C언어 혹은 C++언어를 컴파일하여 실행파일을 만들어줌
    - 만든 결과로는 a.out이 만들어짐
    - 컴파일된 a.out은 ./a.out이라고 치면 컴파일된 프로그램이 실행됨
-  [mission.c파일을 ./a.out을 사용하여 c언어 파일을 실행시킴\(본 레파지토리에 사진을 올려놨습니다.\)](#)



Commits on Mar 27, 2025

### Update README.md



qkrtdals962 authored on Mar 27

### Update README.md



qkrtdals962 authored on Mar 27

### Create README.md



qkrtdals962 authored on Mar 27

0328

Name	Last commit message	Last commit date
..		
README.md	Update README.md	3 months ago
a.out	0328수업	3 months ago
hello.c	0328수업	3 months ago
hellosym.c	0328수업	3 months ago
helloworld.c	0328수업	3 months ago
hw.c	0328수업	3 months ago
mission.c	0328수업	3 months ago

README.md

프로그램 언어(C언어)

- 변수
  - 정의 : 데이터를 저장하고, 그 데이터를 프로그램 내에서 참조하거나 수정할 수 있도록 하는 저장 공간
- 데이터타입
  - int(정수형), char(문자형), float(부동소수점형), double(배정밀도 실수형), long(긴 정수형), long long(더 긴 정수형), unsigned int(부호 없는 정수형), unsigned char(부호 없는 문자형), unsigned long(부호 없는 긴 정수형), void(무형 데이터형)
- 연산자
  - 산술 연산자 : +, -, \*, /, % (덧셈, 뺄셈, 곱셈, 나눗셈, 나머지)
  - 비교 연산자 : ==, !=, >, <, >=, <= (같음, 다름, 크기 비교)
  - 논리 연산자 : &&, ||, ! (AND, OR, NOT)
  - 대입 연산자 : =, +=, -=, \*=, /= (값을 변수에 할당)
  - 증감 연산자 : ++, -- (변수 값을 1씩 증가 또는 감소)
  - 비트 연산자 : &, |, ^, <<, >> (비트 단위로 연산)
- 제어문

Commits on Mar 28, 2025

Update README.md

qkrtdals962 authored on Mar 28

Update README.md

qkrtdals962 authored on Mar 28

Update README.md

qkrtdals962 authored on Mar 28

Update README.md

qkrtdals962 authored on Mar 28

Update README.md

qkrtdals962 authored on Mar 28

Update README.md

qkrtdals962 authored on Mar 28

Update README.md

qkrtdals962 authored on Mar 28

Update README.md

qkrtdals962 authored on Mar 28

Create README.md

qkrtdals962 authored on Mar 28



# 0404

Name	Last commit message	Last commit date
..		
README.md	Update README.md	2 months ago
a.out	0404제출	2 months ago
mission.c	0404제출	2 months ago
README.md		

## AI의 활용법

- 1. [AI의 시장](#)
- 2. [AI의 작동 및 활용 방식](#)
- 3. [AI의 종류 및 특징](#)

## AI의 시장

최근 AI 시장은 급격히 성장하고 있으며, AI는 개인에서 기업, 그리고 산업 전반으로 확장되고 있다. 이러한 확산은 기존의 일자리를 감소시키는 동시에 새로운 일자리를 창출하고 있다.

AI가 산업 전반에 적용될 경우, 기업들은 연간 2조 6천억 달러에서 4조 4천억 달러에 달하는 경제적 가치를 창출할 수 있을 것으로 예상된다. 예를 들어, EPC(설계, 조달, 시공)와 같은 입찰 사업, 건물 설계, 화학 분야 등에서 AI는 실질적인 변화를 이끌고 있으며, AI 기반 시각화 장치는 실무 및 일상생활에서 매우 중요한 역할을 한다.

국내 기업의 72%가 생성형 AI를 채택하고 있으며, IT 및 사이버보안 부서를 포함한 여러 부서에서 AI를 적극적으로 활용하고 있다. 그러나 AI 기술의 광범위한 사용은 오남용의 위험을 내포하고 있다. AI 시스템의 보안 취약점을 악용해 개인정보를 탈취하거나, 기업의 기밀 정보를 유출하는 사례가 발생할 가능성이 있다.

따라서 AI는 단순한 도구가 아니라, 보안과 윤리적 사용에 대한 철저한 관리가 요구되는 기술로 자리 잡고 있다. 이제는 어떤 회사에 입사하더라도 AI의 활용 능력이 필수 역량으로 간주될 것이다.

## AI의 작동 및 활용 방식

History for [ks-WebpgmSys](#) / 0404 on [main](#)

Commits on Apr 4, 2025

### Update README.md

 qkrtdals962 authored on Apr 4

### 0404제출

 qkrtdals962 authored on Apr 4

### Create README.md



 qkrtdals962 authored on Apr 4

End of commit history for this file

# 0411

[ks-WebpgmSys](#) / 0411 / ↑ Top

 test1	0411수업	2 months ago
 test13	0411수업	2 months ago
 test14	0411수업	2 months ago
 test2	0411수업	2 months ago
 test3	0411수업	2 months ago
 test5	0411수업	2 months ago
 test6	0411수업	2 months ago
 test7	0411수업	2 months ago
 test8	0411수업	2 months ago
 test9	0411수업	2 months ago

README.md  

## vi명령어

리눅스 시스템에서 vi명령어란 기본 텍스트 편집기 이다. 기본 명령어 구조로는 `vi 파일이름` 이렇게 사용하고 만약 `vi test.txt` 를 사용하면 `test.txt`라는 파일을 열거나 새로 만들어서 편집할수있습니다.

### vi명령어의 3가지 주요 모드

1. 명령모드

- 텍스트 입력 불가, 복사, 붙여넣기, 삭제, 이동가능한 모드
- ESC를 누르면 다른 모드에서 명령모드로 돌아옴

3. 입력모드

- 실제로 글을 작성, 수정할 수 있는모드



Commits on Apr 11, 2025

Create README.md



qkrtdals962 authored on Apr 11

# 0418 2주늦음

ks-WebpgmSys / 0418 /

Add file



qkrtdals962 Create README.md

45adc1d · last month

History

Name	Last commit message	Last commit date
..		
prej	Add files via upload	last month
README.md	Create README.md	last month
copy.o	Add files via upload	last month
longest	Add files via upload	last month
longest.c	Add files via upload	last month
main	Add files via upload	last month
main.o	Add files via upload	last month

README.md



## 리눅스 개발 도구 개요

### 1. gcc

- gcc는 C 언어로 작성된 소스 코드를 컴파일하여 실행 파일로 만들어주는 컴파일러이다
- GNU Compiler Collection의 약자로, 다양한 언어를 지원하지만 일반적으로는 C 컴파일러로 많이 사용된다
- 컴파일 과정은 전처리, 컴파일, 어셈블, 링크 단계로 구성되어 있다
- gcc는 다양한 옵션을 통해 경고 출력, 최적화, 출력 파일 지정 등을 설정할 수 있다

### 2. make

- make는 소스 코드의 변경 사항을 추적하여 필요한 부분만 컴파일하는 빌드 자동화 도구이다

Commits on May 2, 2025

### Create README.md



qkrtdals962 authored on May 2

End of commit history for this file

# 0502 1주눓음

ks-WebpgmSys / 0502 /

Add file



qkrtdals962 Create README.md

0b0191b · last month

History

Name	Last commit message	Last commit date
..		
3.txt	Add files via upload	last month
README.md	Create README.md	last month
argument	Add files via upload	last month
argument.c	Add files via upload	last month
b.txt	Add files via upload	last month
openex.c	Add files via upload	last month

README.md



## 시스템 호출과 파일

- 시스템 호출은 커널에 기능을 요청하는 방식
- 파일을 사용하려면 open으로 열고, read/write로 처리 후 close로 닫음

## 주요 파일 관련 호출

- open: 파일 열기
- creat: 파일 생성
- read / write: 데이터 읽기, 쓰기
- close: 파일 닫기
- dup / dup2: 파일 디스크립터 복제
- lseek: 파일 내 위치 이동



Commits on May 9, 2025

## Create README.md



qkrtdals962 authored last month



End of commit history for this file

# 0509

ks-WebpgmSys / 0509 /

Add file

...

qkrtdals962 Create README.md

9bfc400 · last month

History

Name	Last commit message	Last commit date
..		
README.md	Create README.md	last month
dbcreate.c	Add files via upload	last month
dbquery.c	Add files via upload	last month
dup.c	Add files via upload	last month
list2	Add files via upload	last month
list2.c	Add files via upload	last month
student.h	Add files via upload	last month

README.md

## 유닉스 파일 시스템 구조

- **부트 블록 (Boot Block):** 부팅 시 사용되는 부트스트랩 코드 저장
- **슈퍼 블록 (Super Block):** 파일 시스템 전체 정보 포함 (블록 수, i-node 수 등)
- **i-리스트 (i-list):** i-node들의 목록 (파일 메타데이터 저장)
- **데이터 블록 (Data Block):** 실제 파일 내용 저장

## i-node 구조

- 하나의 파일은 하나의 i-node를 가짐
- 파일 타입 크기 권한 소유자 블록 포인터 등 메타 정보 포함



Commits on May 9, 2025

Create README.md



qkrtdals962 authored last month



End of commit history for this file

# 0516 1주눅음

ks-WebpgmSys / 0516 /

Add file

...

qkrtdals962 Create README.md

0f3f72e · 3 weeks ago History

Name	Last commit message	Last commit date
..		
README.md	Create README.md	3 weeks ago

README.md

⋮

## 프로세스 구조 및 제어 요약

### ✓ 프로세스란?

- 실행 중인 프로그램을 의미함.
- 각 프로세스는 고유한 PID (Process ID) 를 가짐.
- 대부분 부모-자식 관계를 통해 생성됨.

### ✓ 프로세스의 종류

종류	설명
시스템 프로세스	시스템 기능 수행. 부팅 시 자동 실행됨 (예: 데몬 프로세스)
사용자 프로세스	사용자의 명령어 실행으로 생성됨

### ✓ 프로세스 상태 확인

- `ps` : 현재 실행 중인 프로세스 목록 확인
- `top` : 특정 이름이나 주기에 맞는 프로세스 찾기

History for ks-WebpgmSys / 0516 / README.md

Commits on May 23, 2025

Create README.md

qkrtdals962 authored 2 weeks ago

End of commit history for this file

# 0523

ks-WebpgmSys / 0523 /			↑ Top
📄 README.md	Update README.md	last week	
📄 fork1	Add files via upload	3 weeks ago	
📄 fork1.c	Add files via upload	3 weeks ago	
📄 fork2	Add files via upload	3 weeks ago	
📄 fork2.c	Add files via upload	3 weeks ago	
📄 fork3	Add files via upload	3 weeks ago	
📄 fork3.c	Add files via upload	3 weeks ago	
📄 fork4	Add files via upload	3 weeks ago	
📄 fork4.c	Add files via upload	3 weeks ago	
📄 fork5	Add files via upload	3 weeks ago	
📄 fork5.c	Add files via upload	3 weeks ago	
README.md			✎ ☰

## 프로세스 제어 요약 정리

### ✓ fork1: 기본 프로세스 생성 원리

- `fork()` 시스템 호출을 통해 자식 프로세스를 생성한다.
- `fork()` 는 한 번 호출되지만 두 개의 리턴값이 있다:
  - 부모 프로세스에는 자식 PID 반환
  - 자식 프로세스에는 0 반환
- 이로 인해 부모와 자식은 병렬로 독립적으로 실행된다.

history for ks-WebpgmSys / 0523 / README.md on main

🔑 Commits on Jun 4, 2025

#### Update README.md

👤 qkrkdals962 authored 15 hours ago

🔑 Commits on May 23, 2025

#### Create README.md

👤 qkrkdals962 authored 2 weeks ago

🔑 End of commit history for this file

# 0530

ks-WebpgmSys / 0530 /



Add file



qkrtkdals962 Create README.md

8d035e7 · last week

History

Name	Last commit message	Last commit date
..		
README.md	Create README.md	last week

README.md

## 정규표현식(Regular Expression) 및 grep 명령어 사용법

### 정규표현식(Regular Expression)이란?

- 특정한 패턴을 가진 문자열을 검색하거나 치환하기 위한 표현 방식
- grep, sed, awk, vi, find 등 다양한 명령어에서 사용됨

### 기본 정규표현식 패턴

패턴	의미
.	임의의 한 문자
*	앞 문자의 0개 이상 반복
^	문자열의 시작
\$	문자열의 끝
[]	대괄호 안 문자 중 하나
[^]	대괄호 안 문자 제외
{n}	정확히 n번 반복

History for ks-WebpgmSys / 0530 on main

Commits on Jun 5, 2025

Create README.md

qkrtkdals962 authored now



# 점수 : 28.5점

총 30(15+15)점 중

모든 수업내용 11회를 정리하였지만

늦게 제출한 횡수가 4회이기때문에 -1.5점해서 13.5점 입니다.

프로젝트는 감점사항이 없다고 생각되어 15점입니다.