SWE3004 Operating Systems, Fall 2023

Project 1. Introduction of xv6

TA)
Jongseok Kim
Shinhyun Park
Hyeonmyeong Lee
Gwanjong Park

# Project plan

- Schedule and points

	Assignment	Schedule	Point
1	Booting xv6 operating system	9/7 ~ 9/13	10
2	System call	9/14 ~ 9/27	15
3	CPU scheduling	9/28 ~ 10/11	25
4	Virtual memory	10/26 ~ 11/8	25
5	Page replacement	11/9 ~ 11/22	25
6	File systems	11/23 ~ 12/6	0 (optional)

Project plan

- Total 6 projects
  1) Booting xv6 operating system
  2) System call
  3) CPU scheduling
  4) Virtual memory
  5) Page replacement
  6) File systems

Operating System
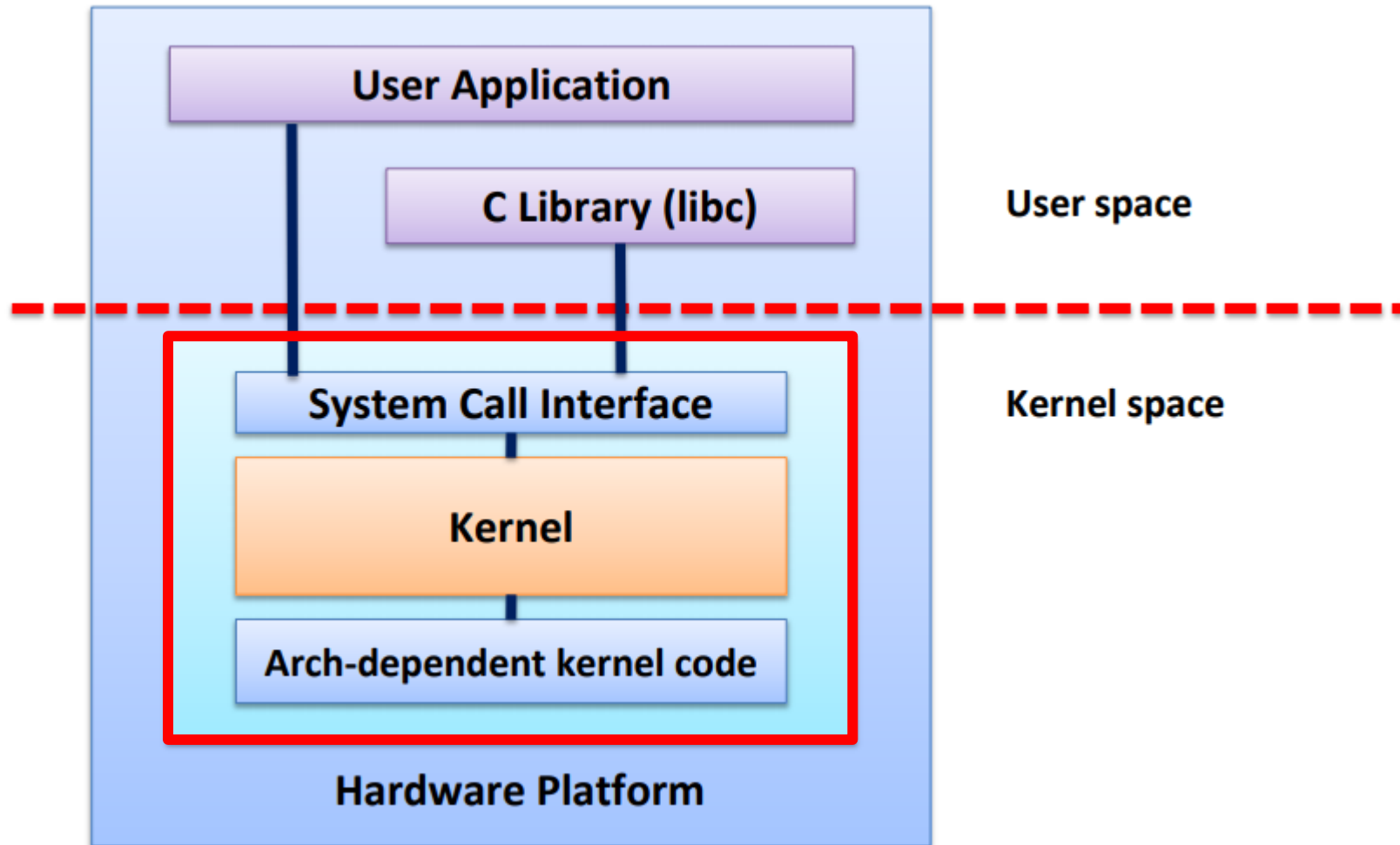
xv6 Operating System

- Unix-like teaching operating system developed by MIT

- Based on multiprocessor x86 system

```
cpu1: starting 1
cpu0: starting 0
sb: size 1000 nblocks 941 ninodes 200 nlog 30 logstart 2 inodestart 32 bmap start 58
init: starting sh
$ ls
.                1 1 512
..               1 1 512
README           2 2 2290
cat              2 3 13332
echo             2 4 12404
forktest         2 5 8120
grep             2 6 15152
init             2 7 12992
kill             2 8 12452
ln               2 9 12352
ls               2 10 14576
mkdir            2 11 12476
rm               2 12 12452
sh               2 13 23092
stressfs         2 14 13132
usertests        2 15 56004
wc               2 16 13980
zombie           2 17 12184
console          3 18 0
$
```

xv6 shell status with ls command

Development Environment

- Ji Server
  - Problems caused by development in other environments will be your own risk.

- Connection method example
  - $ ssh 2023123456@swji.skku.edu –p 1398
  - (If you are new to ji server) Password is pw23123456 (학번 뒤 8 자리)

Clone & booting xv6

- **$** git clone https://github.com/mit-pdos/xv6-public.git

- Run xv6
  - **$** cd xv6-public
  - **$** make qemu-nox

- Terminate xv6
  - ctrl + 'a' → 'x'

Project 1. Booting xv6

- Print your student id, name, and any message(optional) on boot message

```
cpu1: starting 1
cpu0: starting 0
sb: size 1000 nblocks 941 ninodes 200 nlog 30 logstart 2 inodestart 32 bmap start 58
init: starting sh
Student ID: 201xxxxxxx
Name: Gildong Hong
=========Any Message=========
$
```

- Hint : Use grep or other analysis tool

Submission

- Use the *submit & check-submission* file in Ji Server
  - **$ make clean**
  - $ ~swe3004/bin/submit pa1 xv6-public

```
2020712258@swji:/home/2020712258$ ~swe3004/bin/submit pa1 xv6-public
user name          :2020712258
Submitted Files for pa1:
File Name                              File Size        Time
---------------------------------------------------------------------------
pa1-2020712258-Sep.04.11.35.852329576   18326388        Mon Sep  4 11:35:06 2023
pa1-2020712258-Sep.04.10.41.579741638   18326388        Mon Sep  4 10:41:14 2023
```

  - you can submit several times, and the submission history can be checked through check-submission
    - Only the last submission will be graded

```
2020712258@swji:/home/2020712258$ ~swe3004/bin/check-submission pa1
Submitted Files for pa1:
File Name                              File Size        Time
---------------------------------------------------------------------------
pa1-2020712258-Sep.04.11.35.852329576   18326388        Mon Sep  4 11:35:06 2023
pa1-2020712258-Sep.04.10.41.579741638   18326388        Mon Sep  4 10:41:14 2023

Your last submission
---------------------------------------------------------------------------
[#]          File Name                                    File Size
---------------------------------------------------------------------------
```

Submission

- ## Cautions

  – Please check that the submission was successful

  ```
  Submitted Files for pa0:
  File Name                                   File Size          Time
  --------------------------------------------            --------------------------
  pa0-swe3004-Sep.06.19.58.759206525  ①      18341208           Tue Sep  6 19:58:14 2022
  pa0-swe3004-Mar.07.21.14.438787364  ②      126                Tue Mar  7 21:14:18 2023
  pa0-swe3004-Mar.07.21.13.764123202  ③      45                 Tue Mar  7 21:13:10 2023
  ```

  ① Submitted successful xv6

  ② Submitted an empty file

  ③ Submit from the incorrect directory

Submission

- **PLEASE DO NOT COPY**
  - We will run inspection program on all the submissions
  - Any unannounced penalty can be given to **both students**
    - 0 points / negative points / F grade …

- Due date: 9/13(Wed.), 23:59:59 PM
  - -25% per day for delayed submission

Questions

- If you have questions, please ask on icampus
  - Please use the discussion board
  - Discussion board preferred over messages

- You can also visit Corporate Collaboration Center #85533
  - Please iCampus message TA before visiting

- Reading xv6 commentary will help you a lot
  - http://csl.skku.edu/uploads/SSE3044S20/book-rev11.pdf

Appendix. Vim

- Vi Improved

- Text Editor program for Unix

- sudo apt-get install vim
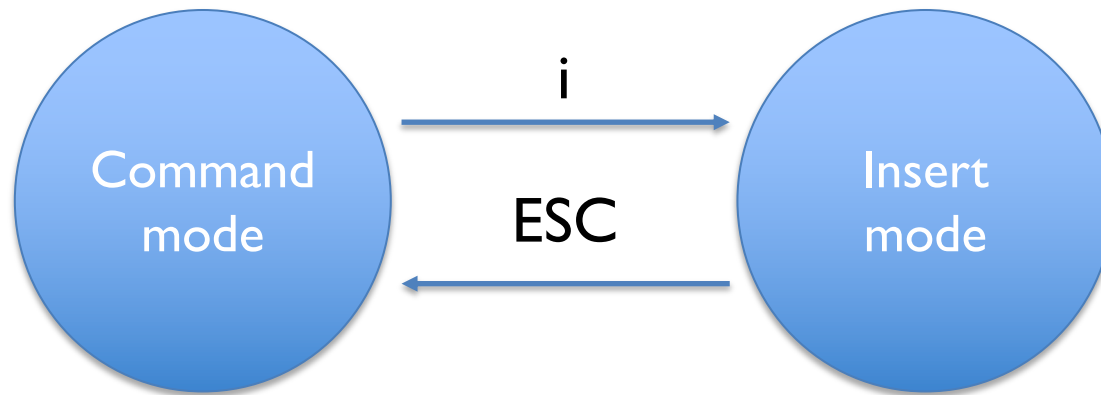
Useful Vim command

- In command mode

Command	Operation
:w	save changes to a file
:q	quit Vim
:wq	save changes and quit Vim
/[string]	search the string from forward
?[string]	search the string from backward
:sp	Divide the window (horizontal)
:vs	Divide the window (vertical)
ctrl + w -> w	move window-to-window

Command	Operation
dd	delete the line
yy	copy the line
p	paste

Vim Setting

- Vim Preference Setting
  - vimrc : settings file used by Vim

Command	Operation
set nu	display line number
set nonu	hide line number
set mouse=a	enable auto visual mode using mouse
set mouse-=a	disable auto visual mode using mouse
set autoindent	set automatic indentation
set ts=[#]	adjust tap size
set et	expand tap

```
~/.vimrc

set et ts=4
set nu
set mouse=a
set autoindent
```

Source code analysis tool

- Ctags, cscope are useful when analyzes source code
  - Also vim editor supports ctags and cscope

- Ctags
  - it recognizes symbols in source code and connects file to each other
  - Allow definitions to be quickly and easily located by a text editor

- Cscope
  - Search for a string or function that calls a function

Ctags

- Vim setting for ctags
  - $ sudo apt-get install ctags
  - $ ctags –R (use in top-level kernel source directory)
  - $ vi ~/.vimrc
  - Add "set tags=[Location of tag file]/tags"
    - Ex)

```
set tags=/home/seungwoo/xv6-sse-local/tags
```

check your path!

Ctags

- Ctags usage (in Vim)
  - Ctrl +] : follow tag

```
for(i=1; i<argc; i++)
  kill(atoi(argv[i]));
exit();
```
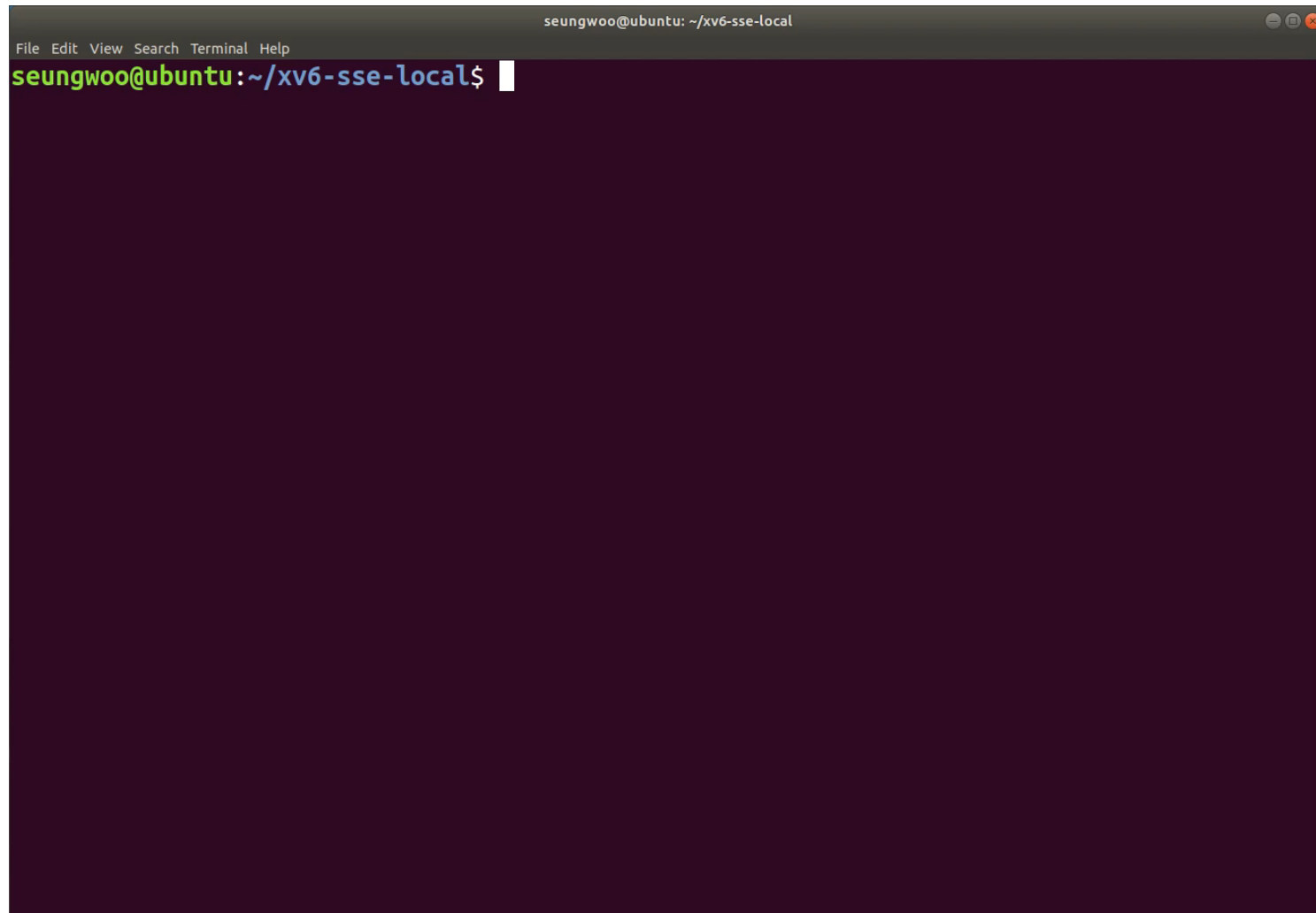
```
int
kill(int pid)
{
  struct proc *p;

  acquire(&ptable.lock);
```

  - Ctrl + T : back to last tag

```
int
kill(int pid)
{
  struct proc *p;

  acquire(&ptable.lock);
```

```
for(i=1; i<argc; i++)
  kill(atoi(argv[i]));
exit();
```

Ctags

Cscope

- Vim setting for cscope
  - $ sudo apt-get install cscope
  - $ cscope –R (use in top-level kernel source directory)
  - Ctrl + D
  - vi ~/.vimrc
  - Add

```
set cst
cs add /home/seungwoo/xv6-sse-local/cscope.out /home/seungwoo/xv6-sse-local/
```

Check your path!

Cscope

- Cscope usage (in Vim)
  - :cs find [type] [search]

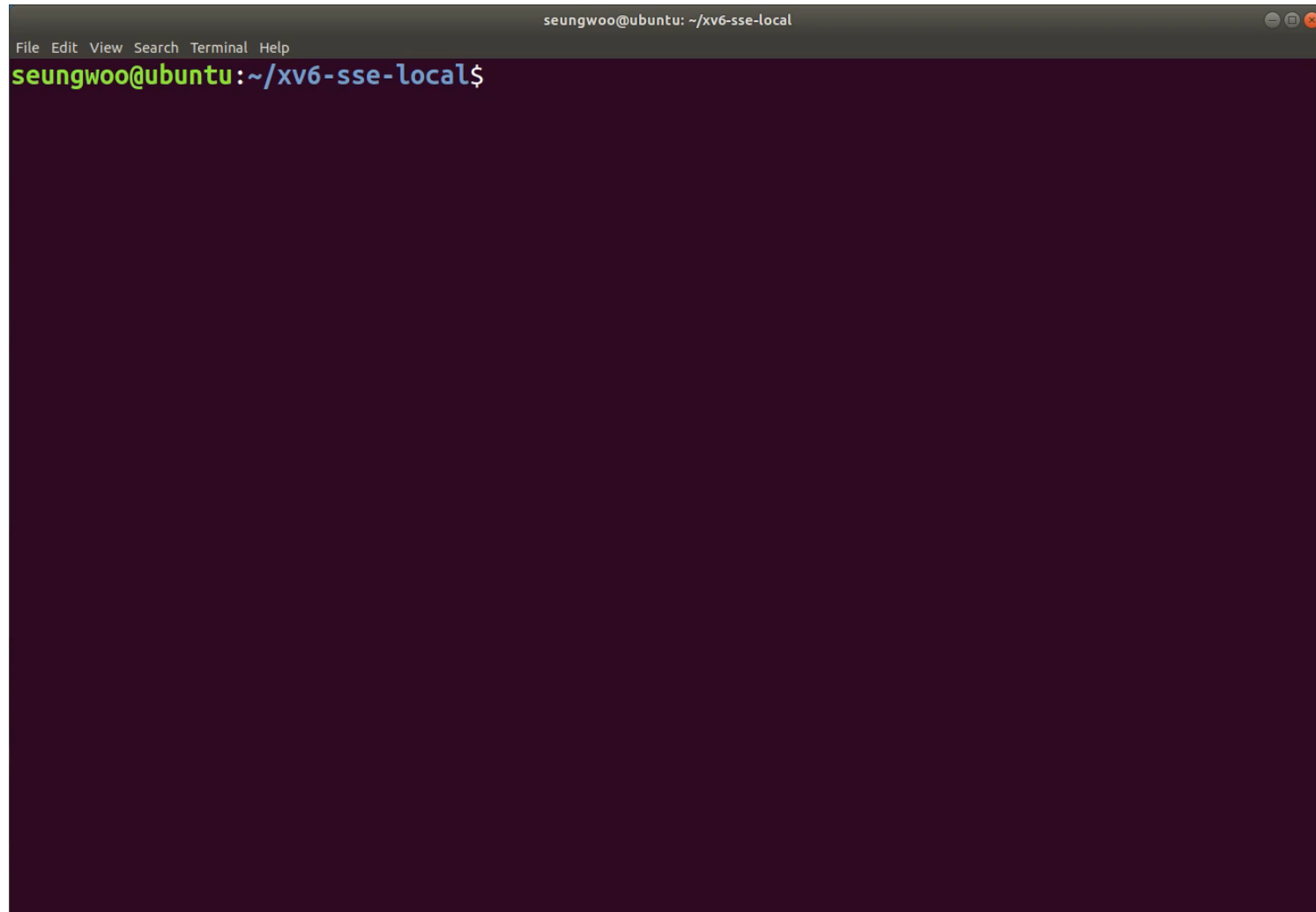| type	Operation
0 or s	Find this C symbol
1 or g	Find this definition
2 or d	Find functions called by this function
3 or c	Find functions calling this function
4 or t	Find assignments to

  - ex) :cs find c [search function]

```
:cs find c fetchint
```

```
Cscope tag: fetchint
   #   line  filename / context / line
   1    155  defs.h <<panic>>
             int fetchint(uint, int *);
   2    155  defs.h <<scheduler>>
             int fetchint(uint, int *);
```

Cscope

Any other useful shell command

- Grep usage
  - grep –nR "[string to search]"
    - Find the file where the string exists and also show line number
    - Easier to use than cscope

    - Ex)

```
grep -nR fetchint
```

```
syscall.c:18:fetchint(uint addr, int *ip)
syscall.c:52:  return fetchint((myproc()->tf->e
kernel.sym:161:801047e0 fetchint
dist/defs.h:154:int            fetchint(uint,
dist/syscall.c:18:fetchint(uint addr, int *ip)
```

- Silversearcher-ag usage
- $ ag [string to search]