**Operating System Design – 19CS2106S**

**Skill – 6**

1. kill.c, grep.c (Xv6 design & implementation. (xv6 source code))

# Kill.c Code

#include "types.h"

#include "stat.h"

#include "user.h"

int main(int argc, char \* argv)

{

int i ;

if(argc < 2){

printf(2, "usage: kill pid...\n");

exit();

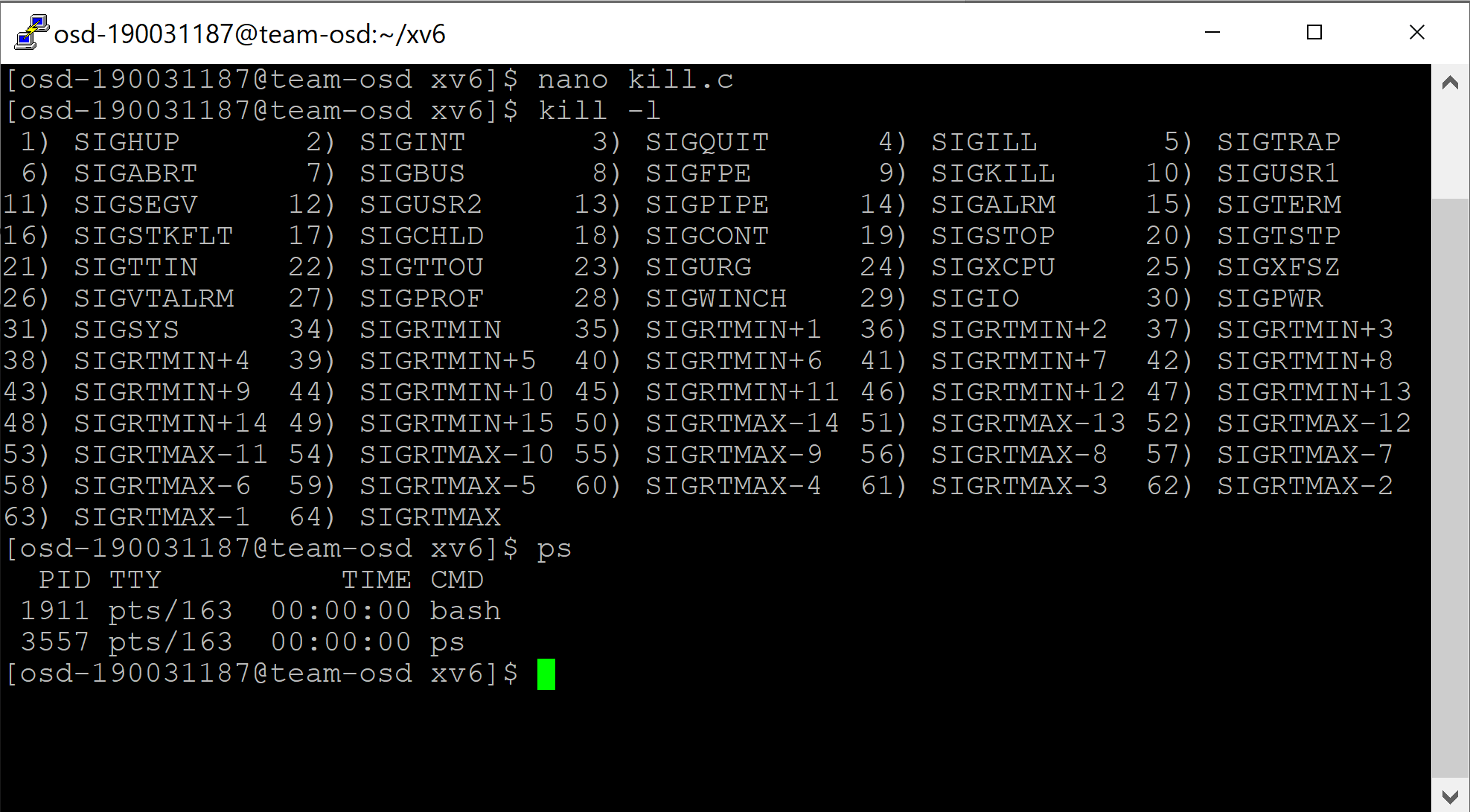
}

for(i=1; i<argc; i++)

kill(atoi(argv[i]));

exit();

}



# 

# Grep.c code

#include "types.h"

#include "stat.h"

#include "user.h"

char buf[1024];

int match(char\*, char\*);

void

grep(char \*pattern, int fd)

{

int n, m;

char \*p, \*q;

m = 0;

while((n = read(fd, buf+m, sizeof(buf)-m)) > 0){

m += n;

p = buf;

while((q = strchr(p, '\n')) != 0){

\*q = 0;

if(match(pattern, p)){

\*q = '\n';

write(1, p, q+1 - p);

}

p = q+1;

}

if(p == buf)

m = 0;

if(m > 0){

m -= p - buf;

memmove(buf, p, m);

}

}

}

int

main(int argc, char \*argv[])

{

int fd, i;

char \*pattern;

if(argc <= 1){

printf(2, "usage: grep pattern [file ...]\n");

exit();

}

pattern = argv[1];

if(argc <= 2){

grep(pattern, 0);

exit();

}

for(i = 2; i < argc; i++){

if((fd = open(argv[i], 0)) < 0){

printf(1, "grep: cannot open %s\n", argv[i]);

exit();

}

grep(pattern, fd);

close(fd);

}

exit();

}

// Regexp matcher from Kernighan & Pike,

// The Practice of Programming, Chapter 9.

int matchhere(char\*, char\*);

int matchstar(int, char\*, char\*);

int

match(char \*re, char \*text)

{

if(re[0] == '^')

return matchhere(re+1, text);

do{ // must look at empty string

if(matchhere(re, text))

return 1;

}while(\*text++ != '\0');

return 0;

}

// matchhere: search for re at beginning of text

int matchhere(char \*re, char \*text)

{

if(re[0] == '\0')

return 1;

if(re[1] == '\*')

return matchstar(re[0], re+2, text);

if(re[0] == '$' && re[1] == '\0')

return \*text == '\0';

if(\*text!='\0' && (re[0]=='.' || re[0]==\*text))

return matchhere(re+1, text+1);

return 0;

}

// matchstar: search for c\*re at beginning of text

int matchstar(int c, char \*re, char \*text)

{

do{ // a \* matches zero or more instances

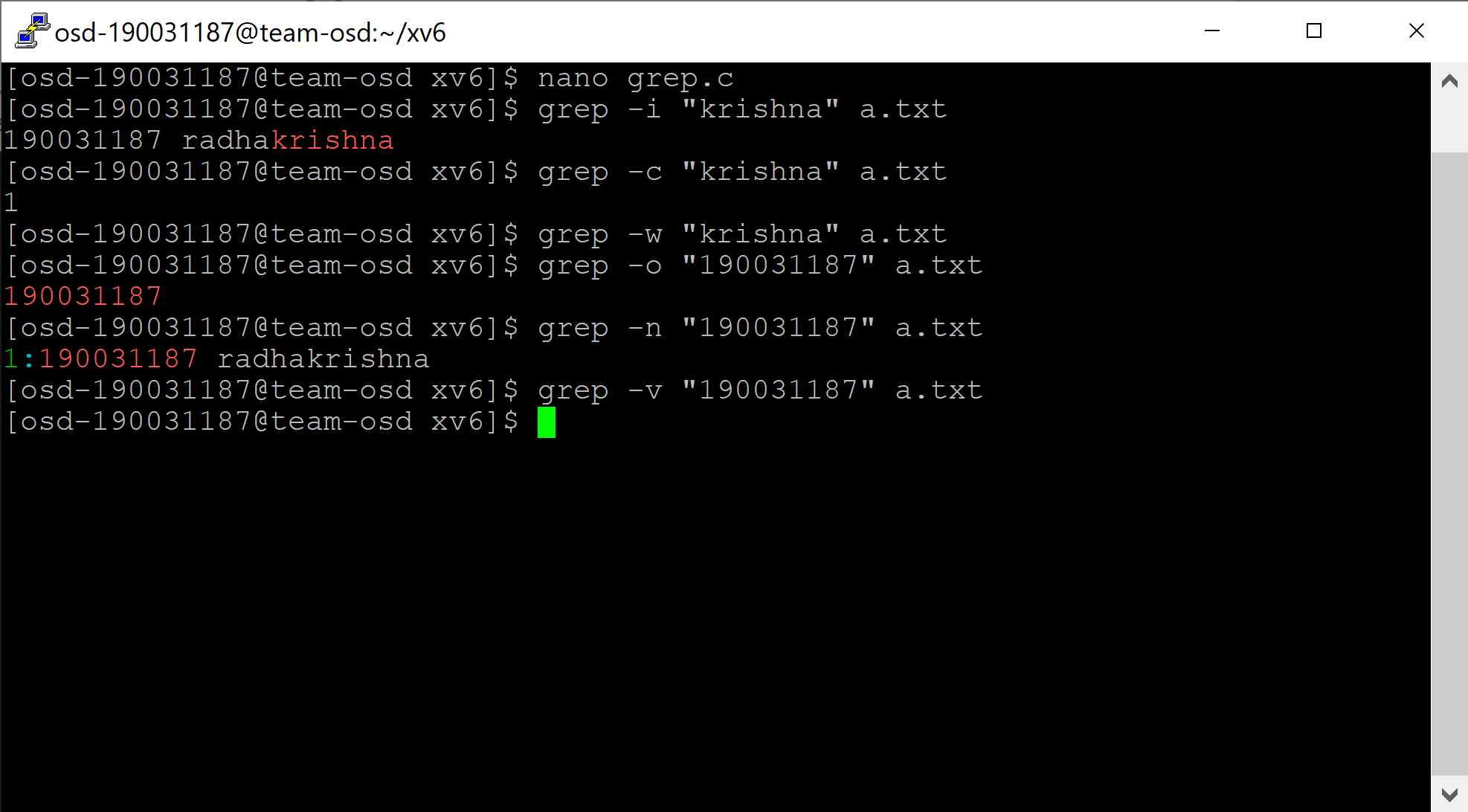
if(matchhere(re, text))

return 1;

}while(\*text!='\0' && (\*text++==c || c=='.'));

return 0;

}



1. Triply-Indirect Block filesystem in xv6 and xv6 filesystem visualizer (xv6 customization)

#include "types.h"

#include "stat.h"

#include "user.h"

#include "fcntl.h"

int

main()

{

char buf[512];

int fd, i, sectors;

fd = open("big.file", O\_CREATE | O\_WRONLY);

if(fd < 0){

printf(2, "big: cannot open big.file for writing\n");

exit();

}

sectors = 0;

while(1){

\*(int\*)buf = sectors;

int cc = write(fd, buf, sizeof(buf));

if(cc <= 0)

break;

sectors++;

if (sectors % 100 == 0)

printf(2, ".");

}

printf(1, "\nwrote %d sectors\n", sectors);

close(fd);

fd = open("big.file", O\_RDONLY);

if(fd < 0){

printf(2, "big: cannot re-open big.file for reading\n");

exit();

}

for(i = 0; i < sectors; i++){

int cc = read(fd, buf, sizeof(buf));

if(cc <= 0){

printf(2, "big: read error at sector %d\n", i);

exit();

}

if(\*(int\*)buf != i){

printf(2, "big: read the wrong data (%d) for sector %d\n",

\*(int\*)buf, i);

exit();

}

}

exit();

}

