

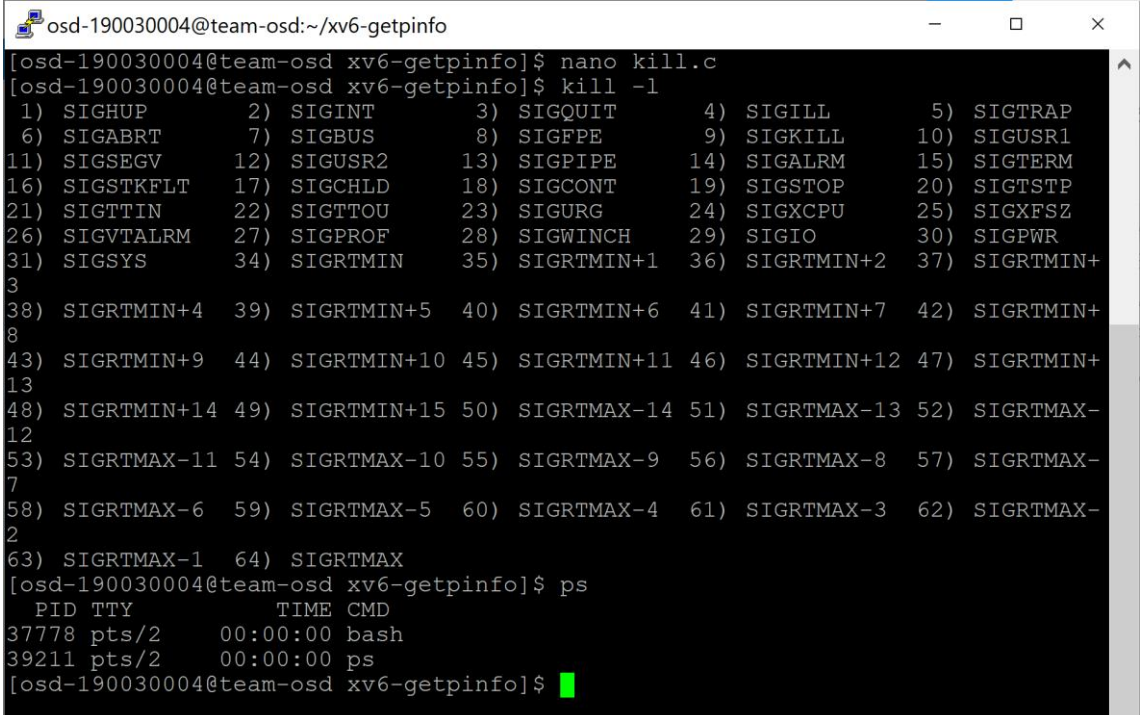
Operating System Design – 19CS2106S

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();
}
```



```
osd-190030004@team-osd:~/xv6-getpinfo
[osd-190030004@team-osd xv6-getpinfo]$ nano kill.c
[osd-190030004@team-osd xv6-getpinfo]$ kill -l
 1) SIGHUP      2) SIGINT      3) SIGQUIT      4) SIGILL      5) SIGTRAP
 6) SIGABRT     7) SIGBUS     8) SIGFPE      9) SIGKILL     10) SIGUSR1
11) SIGSEGV    12) SIGUSR2   13) SIGPIPE    14) SIGALRM    15) SIGTERM
16) SIGSTKFLT  17) SIGCHLD  18) SIGCONT    19) SIGSTOP    20) SIGTSTP
21) SIGTTIN    22) SIGTTOU   23) SIGURG     24) SIGXCPU    25) SIGXFSZ
26) SIGVTALRM  27) SIGPROF  28) SIGWINCH   29) SIGIO       30) SIGPWR
31) SIGSYS     34) SIGRTMIN  35) SIGRTMIN+1 36) SIGRTMIN+2 37) SIGRTMIN+3
38) SIGRTMIN+4 39) SIGRTMIN+5 40) SIGRTMIN+6 41) SIGRTMIN+7 42) SIGRTMIN+8
43) SIGRTMIN+9 44) SIGRTMIN+10 45) SIGRTMIN+11 46) SIGRTMIN+12 47) SIGRTMIN+13
48) SIGRTMIN+14 49) SIGRTMIN+15 50) SIGRTMAX-14 51) SIGRTMAX-13 52) SIGRTMAX-12
53) SIGRTMAX-11 54) SIGRTMAX-10 55) SIGRTMAX-9  56) SIGRTMAX-8  57) SIGRTMAX-7
58) SIGRTMAX-6  59) SIGRTMAX-5 60) SIGRTMAX-4  61) SIGRTMAX-3  62) SIGRTMAX-2
63) SIGRTMAX-1  64) SIGRTMAX
[osd-190030004@team-osd xv6-getpinfo]$ ps
  PID TTY          TIME CMD
 37778 pts/2    00:00:00 bash
 39211 pts/2    00:00:00 ps
[osd-190030004@team-osd xv6-getpinfo]$
```

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;
}
```

A terminal window titled 'osd-190030004@team-osd:~/xv6-getpinf' displays the output of several grep commands. The first command, 'grep -i "Rithik" a.txt', shows '190030004 Rithik Sarab'. The second, 'grep -c "Rithik" a.txt', shows '1'. The third, 'grep -w "Rithik" a.txt', also shows '190030004 Rithik Sarab'. The fourth, 'grep -o "190030004" a.txt', shows '190030004'. The fifth, 'grep -n "190030004" a.txt', shows '1:190030004 Rithik Sarab'. The sixth, 'grep -v "190030004" a.txt', shows no output. The prompt is green.

```
osd-190030004@team-osd:~/xv6-getpinf
[osd-190030004@team-osd xv6-getpinf]$ nano grep.c
[osd-190030004@team-osd xv6-getpinf]$ grep -i "Rithik" a.txt
190030004 Rithik Sarab
[osd-190030004@team-osd xv6-getpinf]$ grep -c "Rithik" a.txt
1
[osd-190030004@team-osd xv6-getpinf]$ grep -w "Rithik" a.txt
190030004 Rithik Sarab
[osd-190030004@team-osd xv6-getpinf]$ grep -o "190030004" a.txt
190030004
[osd-190030004@team-osd xv6-getpinf]$ grep -n "190030004" a.txt
1:190030004 Rithik Sarab
[osd-190030004@team-osd xv6-getpinf]$ grep -v "190030004" a.txt

[osd-190030004@team-osd xv6-getpinf]$
```

2. 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;

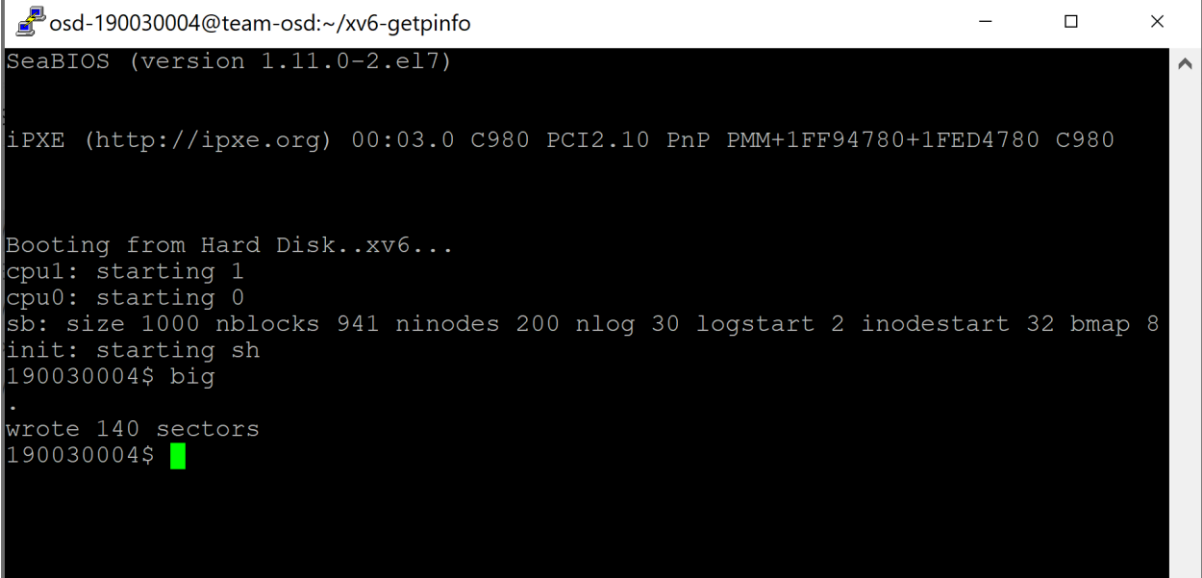
    fd = open("big.file", O_CREATE | O_WRONLY);
    if(fd < 0){
        printf(2, "big: cannot open big.file for writing\n");
        exit();
    }

    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();
}

```

A terminal window titled 'osd-190030004@team-osd:~/xv6-getpinfo' with standard window controls. The terminal output shows the SeaBIOS boot process, including iPXE booting from a hard disk, initializing CPU and SB, and the user '190030004' typing 'big' at the prompt. A green cursor is visible at the end of the prompt.

```
osd-190030004@team-osd:~/xv6-getpinfo
SeaBIOS (version 1.11.0-2.el7)

iPXE (http://ipxe.org) 00:03.0 C980 PCI2.10 PnP PMM+1FF94780+1FED4780 C980

Booting from Hard Disk..xv6...
cpu1: starting 1
cpu0: starting 0
sb: size 1000 nblocks 941 ninodes 200 nlog 30 logstart 2 inodestart 32 bmap 8
init: starting sh
190030004$ big
.
wrote 140 sectors
190030004$ █
```