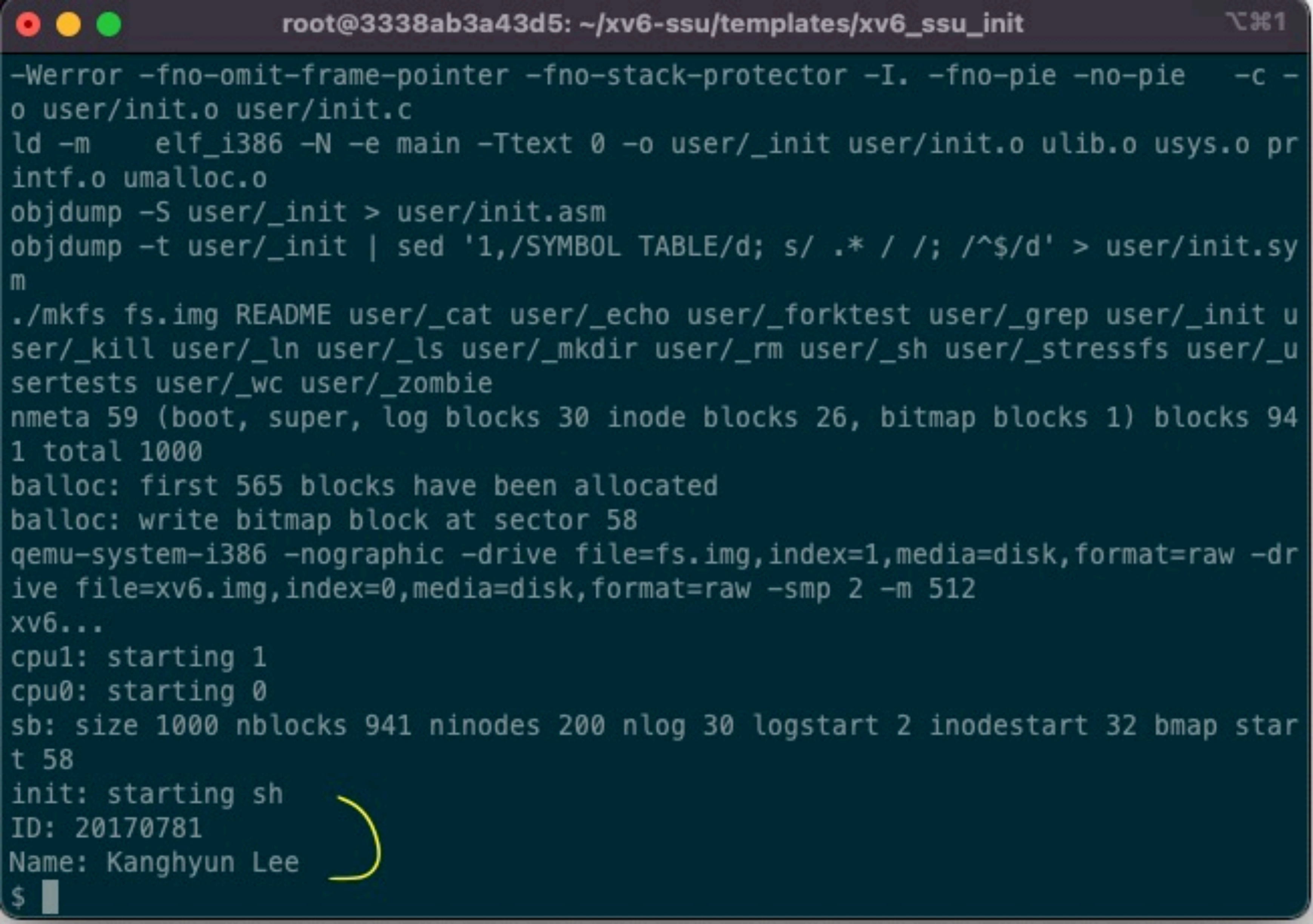
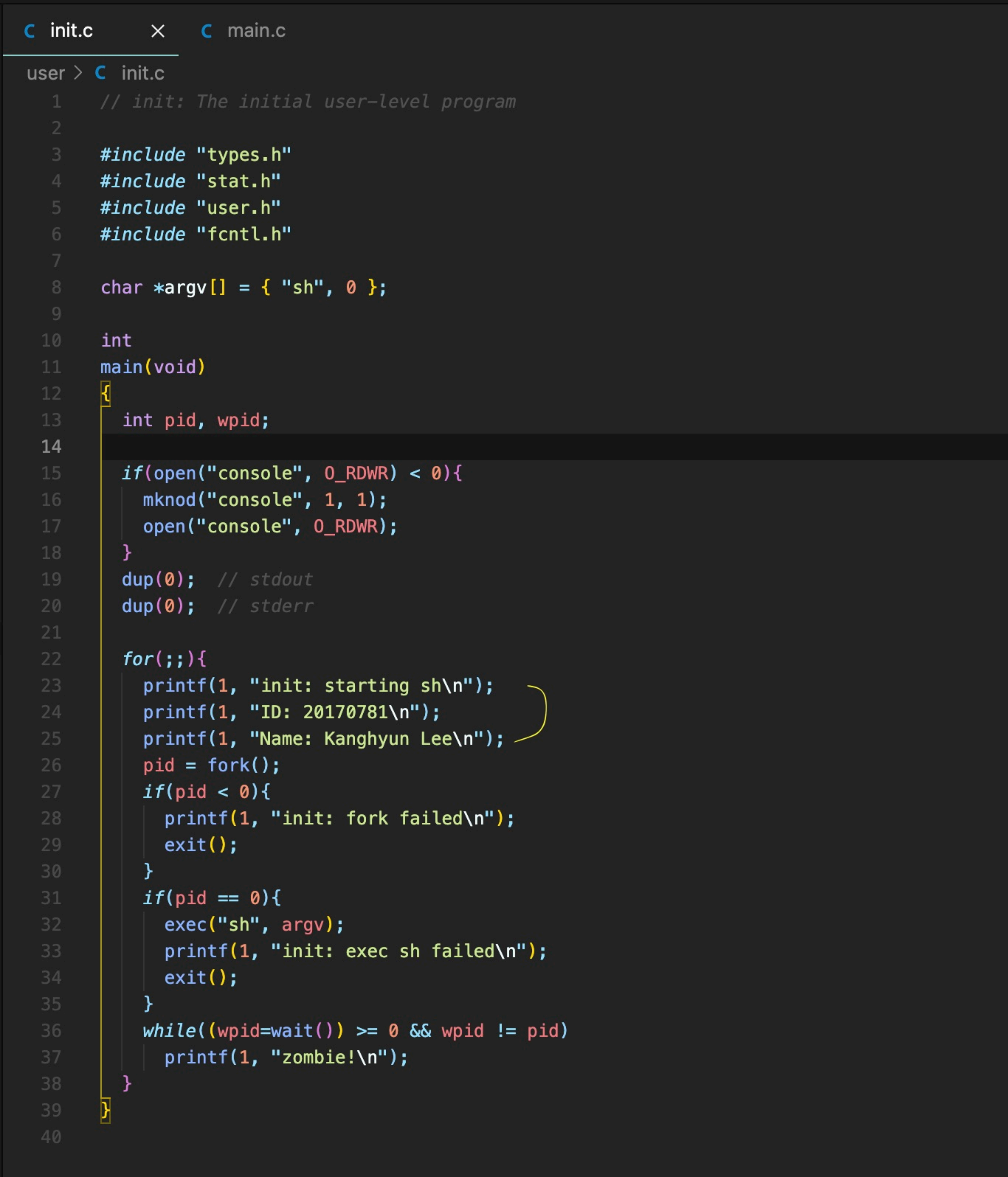


```
-Werror -fno-omit-frame-pointer -fno-stack-protector -I. -fno-pie -no-pie   -c -
o user/init.o user/init.c
ld -m    elf_i386 -N -e main -Ttext 0 -o user/_init user/init.o ulib.o usys.o pr
intf.o umalloc.o
objdump -S user/_init > user/init.asm
objdump -t user/_init | sed '1,/SYMBOL TABLE/d; s/ .* / /; /^$/d' > user/init.sy
m
./mkfs fs.img README user/_cat user/_echo user/_forktest user/_grep user/_init u
ser/_kill user/_ln user/_ls user/_mkdir user/_rm user/_sh user/_stressfs user/_u
sertests user/_wc user/_zombie
nmeta 59 (boot, super, log blocks 30 inode blocks 26, bitmap blocks 1) blocks 94
1 total 1000
balloc: first 565 blocks have been allocated
balloc: write bitmap block at sector 58
qemu-system-i386 -nographic -drive file=fs.img,index=1,media=disk,format=raw -dr
ive file=xv6.img,index=0,media=disk,format=raw -smp 2 -m 512
xv6...
cpu1: starting 1
cpu0: starting 0
sb: size 1000 nblocks 941 ninodes 200 nlog 30 logstart 2 inodestart 32 bmap star
t 58
init: starting sh
ID: 20170781
Name: Kanghyun Lee
$ 
```



```c
// init: The initial user-level program

#include "types.h"
#include "stat.h"
#include "user.h"
#include "fcntl.h"

char *argv[] = { "sh", 0 };

int
main(void)
{
  int pid, wpid;

  if(open("console", O_RDWR) < 0){
    mknod("console", 1, 1);
    open("console", O_RDWR);
  }
  dup(0);  // stdout
  dup(0);  // stderr

  for(;;){
    printf(1, "init: starting sh\n");
    printf(1, "ID: 20170781\n");
    printf(1, "Name: Kanghyun Lee\n");
    pid = fork();
    if(pid < 0){
      printf(1, "init: fork failed\n");
      exit();
    }
    if(pid == 0){
      exec("sh", argv);
      printf(1, "init: exec sh failed\n");
      exit();
    }
    while((wpid=wait()) >= 0 && wpid != pid)
      printf(1, "zombie!\n");
  }
}
```