Getting process statistics and modifying system call XV6

Implemented in XV6-RISCV in DOCKER (in Mac OS - M1)

Code logic:

File: *test.c*
This file is created in the user directory of xv6. Here, a new process is created using *fork()*. Child process: *exec()* replaces the memory address space of the 'test' process with the program passed as its parameter. A new array *argv2[]* is used to store the command line arguments from index 1 (I,e. only arguments followed by test) from *argv* and passed to *exec()*. If any error occurs and *exec()* fails, it returns to this program and appropriate message is printed and program exits with an error.

/xv6-riscv/user/test.c

```c
#include "kernel/types.h"
#include "kernel/stat.h"
#include "user.h"

int main(int argc, char *argv[]) {
    int wt, rt, st;
    int pid;
    char *argv2[argc-1];

    //arguments to be passed in exec() from index 1
    for(int i=1; i<argc; i++){
      argv2[i-1] = argv[i];
    }

    pid = fork();
    if (pid == 0) {
      // Child process
      exec(argv[1], argv2);
      printf("Failed to execute %s\n", argv[1]);
      exit(1);
    } else if (pid > 0) {
      printf("%s pid %d \n", argv[1], pid);
      // Parent process
       waitext(&wt, &rt, &st);
    } else {
      printf("Failed to fork %s\n", argv[1]);
    }

  exit(0);
}
```

The parent process calls *waitext()* to wait for the child process to complete successfully. *waitext()* is extended code of standard *wait()* call, which is mentioned further in this document.

File: *defs.h*

```c
// proc.c
int             cpuid(void);
void            exit(int);
int             fork(void);
int             growproc(int);
void            proc_mapstacks(pagetable_t);
pagetable_t     proc_pagetable(struct proc *);
void            proc_freepagetable(pagetable_t, uint64);
int             kill(int);
int             killed(struct proc*);
void            setkilled(struct proc*);
struct cpu*     mycpu(void);
struct cpu*     getmycpu(void);
struct proc*    myproc();
void            procinit(void);
void            scheduler(void) __attribute__((noreturn));
void            sched(void);
void            sleep(void*, struct spinlock*);
void            userinit(void);
int             wait(uint64);
void            wakeup(void*);
void            yield(void);
int             either_copyout(int user_dst, uint64 dst, void *src, uint64 len);
int             either_copyin(void *dst, int user_src, uint64 src, uint64 len);
void            procdump(void);
int             waitext(uint64, uint64, uint64);
int             ps(void);
```

File: *proc.h*

Declared variables for process creation time (*ctime*), end time (*etime*), run time (*rtime*)and total time (*total_time*) in *proc* structure.

```
// Per-process state
struct proc {
  struct spinlock lock;

  // p->lock must be held when using these:
  enum procstate state;        // Process state
  void *chan;                  // If non-zero, sleeping on chan
  int killed;                  // If non-zero, have been killed
  int xstate;                  // Exit status to be returned to parent's wait
  int pid;                     // Process ID

  // wait_lock must be held when using this:
  struct proc *parent;         // Parent process

  // these are private to the process, so p->lock need not be held.
  uint64 kstack;               // Virtual address of kernel stack
  uint64 sz;                   // Size of process memory (bytes)
  pagetable_t pagetable;       // User page table
  struct trapframe *trapframe; // data page for trampoline.S
  struct context context;      // swtch() here to run process
  struct file *ofile[NOFILE];  // Open files
  struct inode *cwd;           // Current directory
  char name[16];               // Process name (debugging)

  int ctime, etime, rtime;
  uint total_time;
};
```

File *syscall.c*

```
c syscall 〉No Selection
102 extern uint64 sys_mkdir(void);
103 extern uint64 sys_close(void);
104 extern uint64 sys_uniq(void);
105 extern uint64 sys_head(void);
106 extern uint64 sys_waitext(void);
107 extern uint64 sys_ps(void);
108
109 // An array mapping syscall numbers from syscall
110 // to the function that handles the system call.
111 static uint64 (*syscalls[])(void) = {
112 [SYS_fork]    sys_fork,
113 [SYS_exit]    sys_exit,
114 [SYS_wait]    sys_wait,
115 [SYS_pipe]    sys_pipe,
116 [SYS_read]    sys_read,
117 [SYS_kill]    sys_kill,
118 [SYS_exec]    sys_exec,
119 [SYS_fstat]   sys_fstat,
120 [SYS_chdir]   sys_chdir,
121 [SYS_dup]     sys_dup,
122 [SYS_getpid]  sys_getpid,
123 [SYS_sbrk]    sys_sbrk,
124 [SYS_sleep]   sys_sleep,
125 [SYS_uptime]  sys_uptime,
126 [SYS_open]    sys_open,
127 [SYS_write]   sys_write,
128 [SYS_mknod]   sys_mknod,
129 [SYS_unlink]  sys_unlink,
130 [SYS_link]    sys_link,
131 [SYS_mkdir]   sys_mkdir,
132 [SYS_close]   sys_close,
133 [SYS_uniq]    sys_uniq,
134 [SYS_head]    sys_head,
135 [SYS_waitext]   sys_waitext,
136 [SYS_ps]   sys_ps,
137 };
138
```

File: *syscall.h*

```
h syscall 〉No Selection
1  // System call numbers
2  #define SYS_fork    1
3  #define SYS_exit    2
4  #define SYS_wait    3
5  #define SYS_pipe    4
6  #define SYS_read    5
7  #define SYS_kill    6
8  #define SYS_exec    7
9  #define SYS_fstat   8
10 #define SYS_chdir   9
11 #define SYS_dup    10
12 #define SYS_getpid 11
13 #define SYS_sbrk   12
14 #define SYS_sleep  13
15 #define SYS_uptime 14
16 #define SYS_open   15
17 #define SYS_write  16
18 #define SYS_mknod  17
19 #define SYS_unlink 18
20 #define SYS_link   19
21 #define SYS_mkdir  20
22 #define SYS_close  21
23 #define SYS_uniq   22
24 #define SYS_head   23
25 #define SYS_waitext 24
26 #define SYS_ps 25
27
```

File: *usys.pl*

```
usys 〉No Selection
25 entry("kill");
26 entry("exec");
27 entry("open");
28 entry("mknod");
29 entry("unlink");
30 entry("fstat");
31 entry("link");
32 entry("mkdir");
33 entry("chdir");
34 entry("dup");
35 entry("getpid");
36 entry("sbrk");
37 entry("sleep");
38 entry("uptime");
39 entry("uniq");
40 entry("head");
41 entry("waitext");
42 entry("ps");
```

File: *sysproc.c*
The parameters are passed in the user program test.c

```c
uint64
sys_waitext(void)
{
  uint64 wtime;
  uint64 rtime;
  uint64 p;

  argaddr(0, &wtime);

  argaddr(1, &rtime);

  argaddr(2, &p);

  return waitext(wtime, rtime, p);
}
```

File: *user.h*

```h
int unlink(const char*);
int fstat(int fd, struct stat*);
int link(const char*, const char*);
int mkdir(const char*);
int chdir(const char*);
int dup(int);
int getpid(void);
char* sbrk(int);
int sleep(int);
int uptime(void);
int uniq(int n, char *buffer, char var_icd);
int head(int n, char *buffer, int lines);
int waitext(int *wtime, int *rtime, int *addr);
int ps(void);
```

File: *trap.c*
In *kerneltrap()* function when a timer interrupt occurs when the process state is running or sleeping, the *rtime* is incremented.

```c
kerneltrap()

  // give up the CPU if this is a timer interrupt.
  if(which_dev == 2 && myproc() != 0 && myproc()->state == RUNNING){
    myproc()->rtime++;
    yield();
  }
  if(which_dev == 2 && myproc() != 0 && myproc()->state == SLEEPING){
    myproc()->rtime++;
  }

  // the yield() may have caused some traps to occur,
  // so restore trap registers for use by kernelvec.S's sepc instruction.
  w_sepc(sepc);
  w_sstatus(sstatus);
}
```

File: *proc.c*
All the declared time fields are initialized in *allocproc()* function. The creation time *ctime* is incremented with ticks, as the process starts. In *exit()*, process end time *etime* is incremented with the ticks.

```
C proc 〉No Selection
110  allocproc(void)
148
149    // time fields initializatio
150    p->ctime = ticks;          //
151    p->etime = 0;              //
152    p->rtime = 0;              //
153    p->total_time = 0;
154
155    return p;
156  }
```

```
C proc 〉No Selection
353  exit(int status)
385    p->state = ZOMBIE;
386
387    release(&wait_lock);
388
389    // updating end time
390     p->etime = ticks;
391
392    // Jump into the scheduler, never to return.
393    sched();
394    panic("zombie exit");
395  }
```

In *scheduler()* function, the total_time is incremented which stores the ticks whenever the process is running. In the *waitext()* function if a zombie process is found then the run time *rtime* is updated by difference of start and end time. The creation time, end time, the runtime incremented by timer interrupt, total scheduler time are printed.

```
C proc 〉f scheduler()
454  scheduler(void)
464    for(p = proc; p < &proc[NPROC]; p++) {
465      acquire(&p->lock);
466      if(p->state == RUNNABLE) {
467        //update total time
468        p->total_time += 1;
469        // Switch to chosen process.  It is the
470        // to release its lock and then reacquir
471        // before jumping back to us.
472        p->state = RUNNING;
473        c->proc = p;
474        swtch(&c->context, &p->context);
```

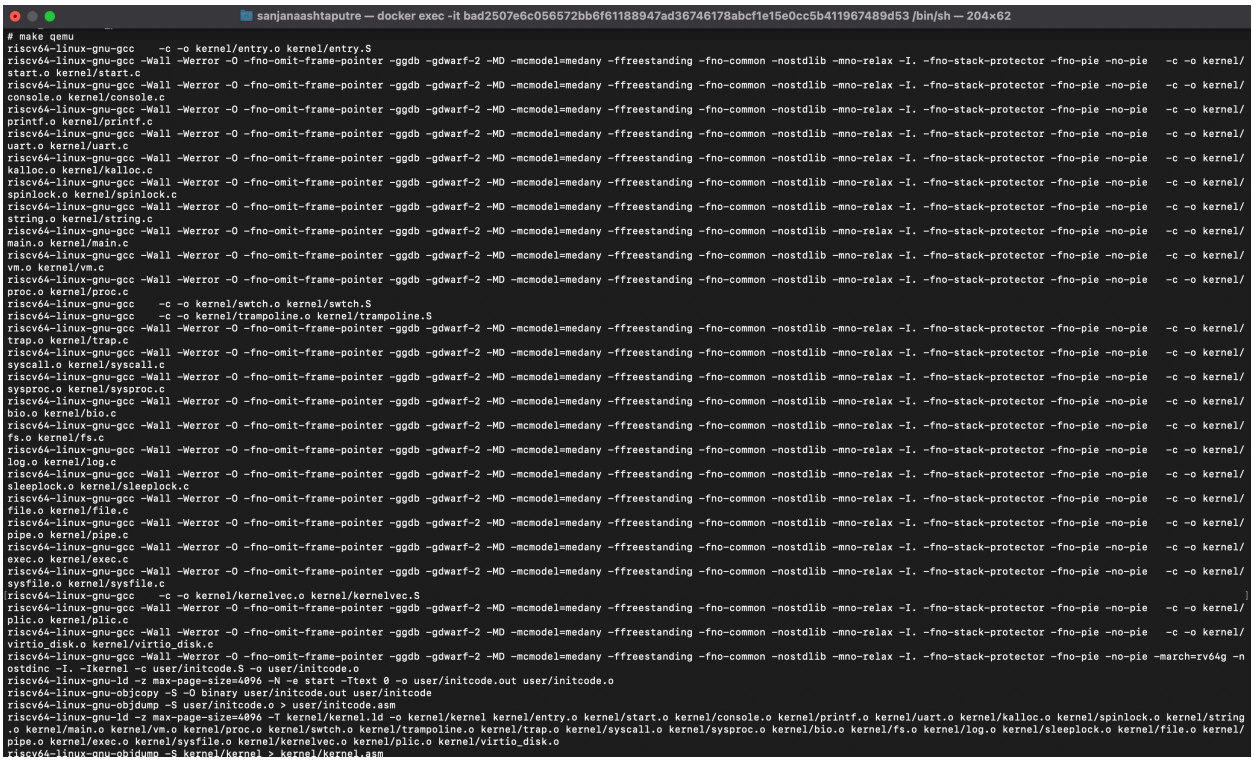
```
C proc 〉No Selection
697  waitext(uint64 wtime, uint64 rtime, uint64 addr)
714        if(pp->state == ZOMBIE){
715          // Found one.
716
717        //update run time
718        rtime = pp->etime - pp->ctime;
719
720        printf("\ncreation time : %d \nrun time : %d \nend time : %d \ntimer
               interrupt while running or sleeping: %d \n", pp->ctime, rtime,
               pp->etime, pp->rtime);
721        printf("process name : %s \ntotal scheduler time: %d \n",pp->name,
               pp->total_time);
```

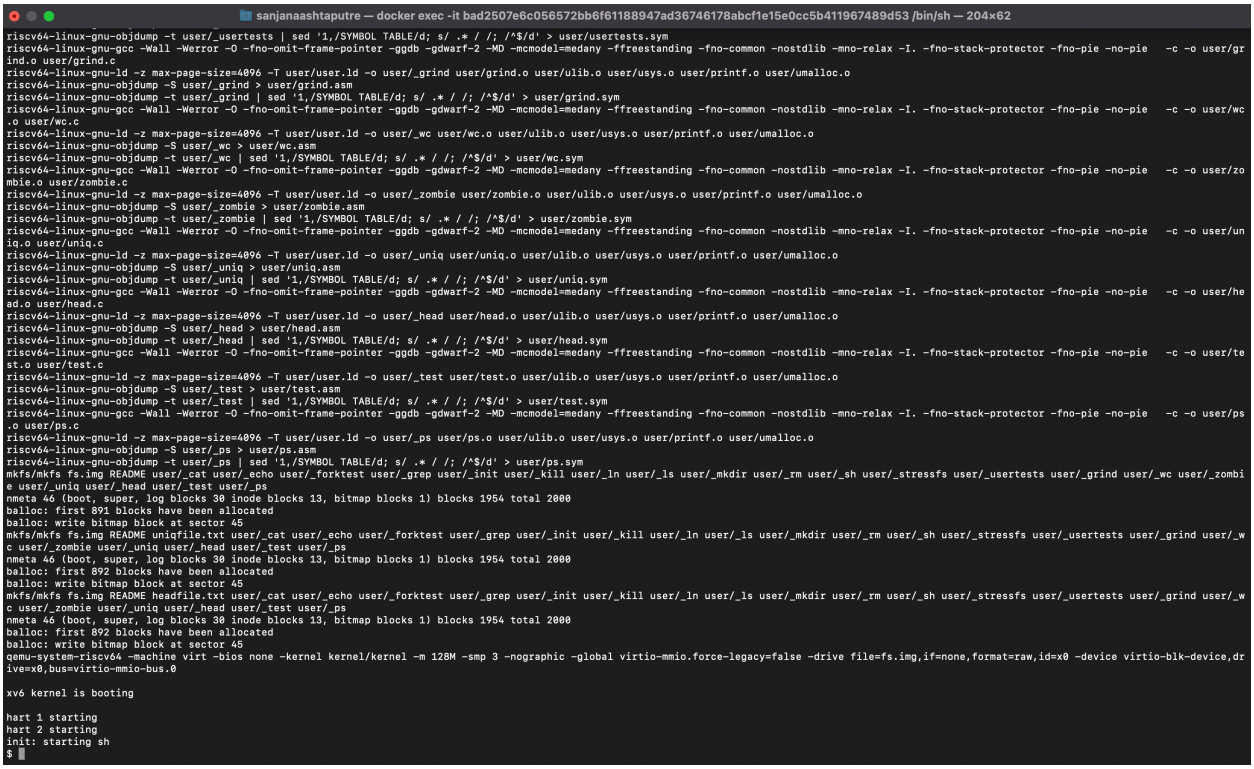
Steps to run the code and its output:

make clean

```
sanjanaashtaputre — docker exec -it bad2507e6c056572bb6f61188947a...
# make clean
rm -f *.tex *.dvi *.idx *.aux *.log *.ind *.ilg \
*/*.o */*.d */*.asm */*.sym \
user/initcode user/initcode.out kernel/kernel fs.img \
mkfs/mkfs .gdbinit \
        user/usys.S \
user/_cat user/_echo user/_forktest user/_grep user/_init user/_kill user/_ln us
er/_ls user/_mkdir user/_rm user/_sh user/_stressfs user/_usertests user/_grind
user/_wc user/_zombie user/_uniq user/_head user/_test user/_ps
#
```

make qemu

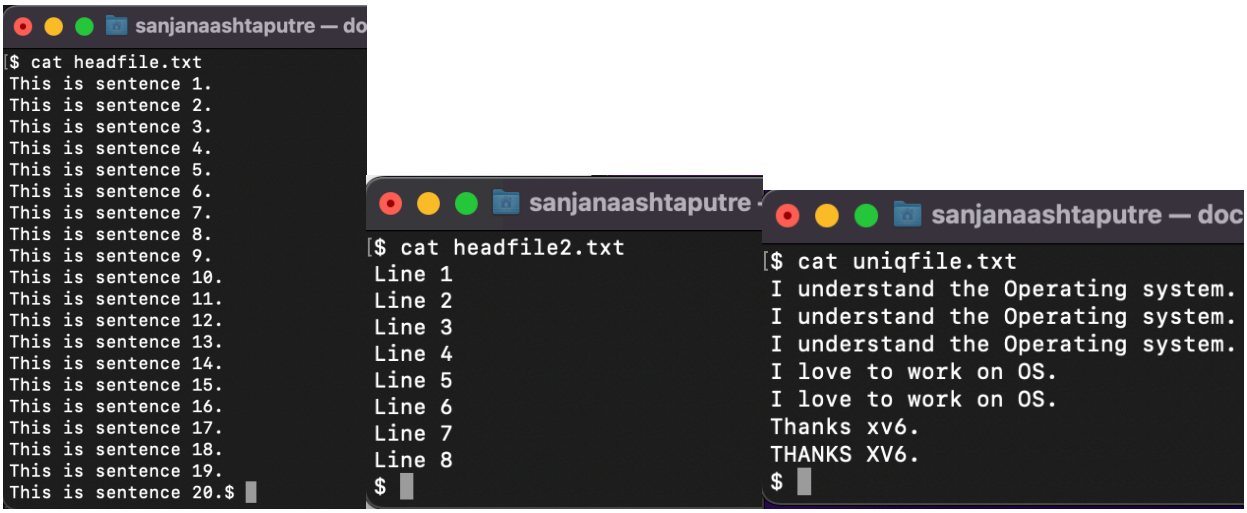

sanjanaashtaputre — docker exec -it bad2507e6c056572bb6f61188947ad36746178abcf1e15e0cc5b411967489d53 /bin/sh — 204×62
make qemu
riscv64-linux-gnu-gcc   -c -o kernel/entry.o kernel/entry.S
riscv64-linux-gnu-gcc -Wall -Werror -O -fno-omit-frame-pointer -ggdb -gdwarf-2 -MD -mcmodel=medany -ffreestanding -fno-common -nostdlib -mno-relax -I. -fno-stack-protector -fno-pie -no-pie   -c -o kernel/start.o kernel/start.c
riscv64-linux-gnu-gcc -Wall -Werror -O -fno-omit-frame-pointer -ggdb -gdwarf-2 -MD -mcmodel=medany -ffreestanding -fno-common -nostdlib -mno-relax -I. -fno-stack-protector -fno-pie -no-pie   -c -o kernel/console.o kernel/console.c
riscv64-linux-gnu-gcc -Wall -Werror -O -fno-omit-frame-pointer -ggdb -gdwarf-2 -MD -mcmodel=medany -ffreestanding -fno-common -nostdlib -mno-relax -I. -fno-stack-protector -fno-pie -no-pie   -c -o kernel/printf.o kernel/printf.c
riscv64-linux-gnu-gcc -Wall -Werror -O -fno-omit-frame-pointer -ggdb -gdwarf-2 -MD -mcmodel=medany -ffreestanding -fno-common -nostdlib -mno-relax -I. -fno-stack-protector -fno-pie -no-pie   -c -o kernel/uart.o kernel/uart.c
riscv64-linux-gnu-gcc -Wall -Werror -O -fno-omit-frame-pointer -ggdb -gdwarf-2 -MD -mcmodel=medany -ffreestanding -fno-common -nostdlib -mno-relax -I. -fno-stack-protector -fno-pie -no-pie   -c -o kernel/kalloc.o kernel/kalloc.c
riscv64-linux-gnu-gcc -Wall -Werror -O -fno-omit-frame-pointer -ggdb -gdwarf-2 -MD -mcmodel=medany -ffreestanding -fno-common -nostdlib -mno-relax -I. -fno-stack-protector -fno-pie -no-pie   -c -o kernel/spinlock.o kernel/spinlock.c
riscv64-linux-gnu-gcc -Wall -Werror -O -fno-omit-frame-pointer -ggdb -gdwarf-2 -MD -mcmodel=medany -ffreestanding -fno-common -nostdlib -mno-relax -I. -fno-stack-protector -fno-pie -no-pie   -c -o kernel/string.o kernel/string.c
riscv64-linux-gnu-gcc -Wall -Werror -O -fno-omit-frame-pointer -ggdb -gdwarf-2 -MD -mcmodel=medany -ffreestanding -fno-common -nostdlib -mno-relax -I. -fno-stack-protector -fno-pie -no-pie   -c -o kernel/main.o kernel/main.c
riscv64-linux-gnu-gcc -Wall -Werror -O -fno-omit-frame-pointer -ggdb -gdwarf-2 -MD -mcmodel=medany -ffreestanding -fno-common -nostdlib -mno-relax -I. -fno-stack-protector -fno-pie -no-pie   -c -o kernel/vm.o kernel/vm.c
riscv64-linux-gnu-gcc -Wall -Werror -O -fno-omit-frame-pointer -ggdb -gdwarf-2 -MD -mcmodel=medany -ffreestanding -fno-common -nostdlib -mno-relax -I. -fno-stack-protector -fno-pie -no-pie   -c -o kernel/proc.o kernel/proc.c
riscv64-linux-gnu-gcc   -c -o kernel/swtch.o kernel/swtch.S
riscv64-linux-gnu-gcc   -c -o kernel/trampoline.o kernel/trampoline.S
riscv64-linux-gnu-gcc -Wall -Werror -O -fno-omit-frame-pointer -ggdb -gdwarf-2 -MD -mcmodel=medany -ffreestanding -fno-common -nostdlib -mno-relax -I. -fno-stack-protector -fno-pie -no-pie   -c -o kernel/trap.o kernel/trap.c
riscv64-linux-gnu-gcc -Wall -Werror -O -fno-omit-frame-pointer -ggdb -gdwarf-2 -MD -mcmodel=medany -ffreestanding -fno-common -nostdlib -mno-relax -I. -fno-stack-protector -fno-pie -no-pie   -c -o kernel/syscall.o kernel/syscall.c
riscv64-linux-gnu-gcc -Wall -Werror -O -fno-omit-frame-pointer -ggdb -gdwarf-2 -MD -mcmodel=medany -ffreestanding -fno-common -nostdlib -mno-relax -I. -fno-stack-protector -fno-pie -no-pie   -c -o kernel/sysproc.o kernel/sysproc.c
riscv64-linux-gnu-gcc -Wall -Werror -O -fno-omit-frame-pointer -ggdb -gdwarf-2 -MD -mcmodel=medany -ffreestanding -fno-common -nostdlib -mno-relax -I. -fno-stack-protector -fno-pie -no-pie   -c -o kernel/bio.o kernel/bio.c
riscv64-linux-gnu-gcc -Wall -Werror -O -fno-omit-frame-pointer -ggdb -gdwarf-2 -MD -mcmodel=medany -ffreestanding -fno-common -nostdlib -mno-relax -I. -fno-stack-protector -fno-pie -no-pie   -c -o kernel/fs.o kernel/fs.c
riscv64-linux-gnu-gcc -Wall -Werror -O -fno-omit-frame-pointer -ggdb -gdwarf-2 -MD -mcmodel=medany -ffreestanding -fno-common -nostdlib -mno-relax -I. -fno-stack-protector -fno-pie -no-pie   -c -o kernel/log.o kernel/log.c
riscv64-linux-gnu-gcc -Wall -Werror -O -fno-omit-frame-pointer -ggdb -gdwarf-2 -MD -mcmodel=medany -ffreestanding -fno-common -nostdlib -mno-relax -I. -fno-stack-protector -fno-pie -no-pie   -c -o kernel/sleeplock.o kernel/sleeplock.c
riscv64-linux-gnu-gcc -Wall -Werror -O -fno-omit-frame-pointer -ggdb -gdwarf-2 -MD -mcmodel=medany -ffreestanding -fno-common -nostdlib -mno-relax -I. -fno-stack-protector -fno-pie -no-pie   -c -o kernel/file.o kernel/file.c
riscv64-linux-gnu-gcc -Wall -Werror -O -fno-omit-frame-pointer -ggdb -gdwarf-2 -MD -mcmodel=medany -ffreestanding -fno-common -nostdlib -mno-relax -I. -fno-stack-protector -fno-pie -no-pie   -c -o kernel/pipe.o kernel/pipe.c
riscv64-linux-gnu-gcc -Wall -Werror -O -fno-omit-frame-pointer -ggdb -gdwarf-2 -MD -mcmodel=medany -ffreestanding -fno-common -nostdlib -mno-relax -I. -fno-stack-protector -fno-pie -no-pie   -c -o kernel/exec.o kernel/exec.c
riscv64-linux-gnu-gcc -Wall -Werror -O -fno-omit-frame-pointer -ggdb -gdwarf-2 -MD -mcmodel=medany -ffreestanding -fno-common -nostdlib -mno-relax -I. -fno-stack-protector -fno-pie -no-pie   -c -o kernel/sysfile.o kernel/sysfile.c
riscv64-linux-gnu-gcc   -c -o kernel/kernelvec.o kernel/kernelvec.S
riscv64-linux-gnu-gcc -Wall -Werror -O -fno-omit-frame-pointer -ggdb -gdwarf-2 -MD -mcmodel=medany -ffreestanding -fno-common -nostdlib -mno-relax -I. -fno-stack-protector -fno-pie -no-pie   -c -o kernel/plic.o kernel/plic.c
riscv64-linux-gnu-gcc -Wall -Werror -O -fno-omit-frame-pointer -ggdb -gdwarf-2 -MD -mcmodel=medany -ffreestanding -fno-common -nostdlib -mno-relax -I. -fno-stack-protector -fno-pie -no-pie   -c -o kernel/virtio_disk.o kernel/virtio_disk.c
riscv64-linux-gnu-gcc -Wall -Werror -O -fno-omit-frame-pointer -ggdb -gdwarf-2 -MD -mcmodel=medany -ffreestanding -fno-common -nostdlib -mno-relax -I. -fno-stack-protector -fno-pie -no-pie -march=rv64g -nostdinc -I. -Ikernel -c user/initcode.S -o user/initcode.o
riscv64-linux-gnu-ld -z max-page-size=4096 -N -e start -Ttext 0 -o user/initcode.out user/initcode.o
riscv64-linux-gnu-objcopy -S -O binary user/initcode.out user/initcode
riscv64-linux-gnu-objdump -S user/initcode.o > user/initcode.asm
riscv64-linux-gnu-ld -z max-page-size=4096 -T kernel/kernel.ld -o kernel/kernel kernel/entry.o kernel/start.o kernel/console.o kernel/printf.o kernel/uart.o kernel/kalloc.o kernel/spinlock.o kernel/string.o kernel/main.o kernel/vm.o kernel/proc.o kernel/swtch.o kernel/trampoline.o kernel/trap.o kernel/syscall.o kernel/sysproc.o kernel/bio.o kernel/fs.o kernel/log.o kernel/sleeplock.o kernel/file.o kernel/pipe.o kernel/exec.o kernel/sysfile.o kernel/kernelvec.o kernel/plic.o kernel/virtio_disk.o
riscv64-linux-gnu-objdump -S kernel/kernel > kernel/kernel.asm


sanjanaashtaputre — docker exec -it bad2507e6c056572bb6f61188947ad36746178abcf1e15e0cc5b411967489d53 /bin/sh — 204×62
riscv64-linux-gnu-objdump -t user/_usertests | sed '1,/SYMBOL TABLE/d; s/ .* / /; /^$/d' > user/usertests.sym
riscv64-linux-gnu-gcc -Wall -Werror -O -fno-omit-frame-pointer -ggdb -gdwarf-2 -MD -mcmodel=medany -ffreestanding -fno-common -nostdlib -mno-relax -I. -fno-stack-protector -fno-pie -no-pie   -c -o user/grind.o user/grind.c
riscv64-linux-gnu-ld -z max-page-size=4096 -T user/user.ld -o user/_grind user/grind.o user/ulib.o user/usys.o user/printf.o user/umalloc.o
riscv64-linux-gnu-objdump -S user/_grind > user/grind.asm
riscv64-linux-gnu-objdump -t user/_grind | sed '1,/SYMBOL TABLE/d; s/ .* / /; /^$/d' > user/grind.sym
riscv64-linux-gnu-gcc -Wall -Werror -O -fno-omit-frame-pointer -ggdb -gdwarf-2 -MD -mcmodel=medany -ffreestanding -fno-common -nostdlib -mno-relax -I. -fno-stack-protector -fno-pie -no-pie   -c -o user/wc.o user/wc.c
riscv64-linux-gnu-ld -z max-page-size=4096 -T user/user.ld -o user/_wc user/wc.o user/ulib.o user/usys.o user/printf.o user/umalloc.o
riscv64-linux-gnu-objdump -S user/_wc > user/wc.asm
riscv64-linux-gnu-objdump -t user/_wc | sed '1,/SYMBOL TABLE/d; s/ .* / /; /^$/d' > user/wc.sym
riscv64-linux-gnu-gcc -Wall -Werror -O -fno-omit-frame-pointer -ggdb -gdwarf-2 -MD -mcmodel=medany -ffreestanding -fno-common -nostdlib -mno-relax -I. -fno-stack-protector -fno-pie -no-pie   -c -o user/zombie.o user/zombie.c
riscv64-linux-gnu-ld -z max-page-size=4096 -T user/user.ld -o user/_zombie user/zombie.o user/ulib.o user/usys.o user/printf.o user/umalloc.o
riscv64-linux-gnu-objdump -S user/_zombie > user/zombie.asm
riscv64-linux-gnu-objdump -t user/_zombie | sed '1,/SYMBOL TABLE/d; s/ .* / /; /^$/d' > user/zombie.sym
riscv64-linux-gnu-gcc -Wall -Werror -O -fno-omit-frame-pointer -ggdb -gdwarf-2 -MD -mcmodel=medany -ffreestanding -fno-common -nostdlib -mno-relax -I. -fno-stack-protector -fno-pie -no-pie   -c -o user/uniq.o user/uniq.c
riscv64-linux-gnu-ld -z max-page-size=4096 -T user/user.ld -o user/_uniq user/uniq.o user/ulib.o user/usys.o user/printf.o user/umalloc.o
riscv64-linux-gnu-objdump -S user/_uniq > user/uniq.asm
riscv64-linux-gnu-objdump -t user/_uniq | sed '1,/SYMBOL TABLE/d; s/ .* / /; /^$/d' > user/uniq.sym
riscv64-linux-gnu-gcc -Wall -Werror -O -fno-omit-frame-pointer -ggdb -gdwarf-2 -MD -mcmodel=medany -ffreestanding -fno-common -nostdlib -mno-relax -I. -fno-stack-protector -fno-pie -no-pie   -c -o user/head.o user/head.c
riscv64-linux-gnu-ld -z max-page-size=4096 -T user/user.ld -o user/_head user/head.o user/ulib.o user/usys.o user/printf.o user/umalloc.o
riscv64-linux-gnu-objdump -S user/_head > user/head.asm
riscv64-linux-gnu-objdump -t user/_head | sed '1,/SYMBOL TABLE/d; s/ .* / /; /^$/d' > user/head.sym
riscv64-linux-gnu-gcc -Wall -Werror -O -fno-omit-frame-pointer -ggdb -gdwarf-2 -MD -mcmodel=medany -ffreestanding -fno-common -nostdlib -mno-relax -I. -fno-stack-protector -fno-pie -no-pie   -c -o user/test.o user/test.c
riscv64-linux-gnu-ld -z max-page-size=4096 -T user/user.ld -o user/_test user/test.o user/ulib.o user/usys.o user/printf.o user/umalloc.o
riscv64-linux-gnu-objdump -S user/_test > user/test.asm
riscv64-linux-gnu-objdump -t user/_test | sed '1,/SYMBOL TABLE/d; s/ .* / /; /^$/d' > user/test.sym
riscv64-linux-gnu-gcc -Wall -Werror -O -fno-omit-frame-pointer -ggdb -gdwarf-2 -MD -mcmodel=medany -ffreestanding -fno-common -nostdlib -mno-relax -I. -fno-stack-protector -fno-pie -no-pie   -c -o user/ps.o user/ps.c
riscv64-linux-gnu-ld -z max-page-size=4096 -T user/user.ld -o user/_ps user/ps.o user/ulib.o user/usys.o user/printf.o user/umalloc.o
riscv64-linux-gnu-objdump -S user/_ps > user/ps.asm
riscv64-linux-gnu-objdump -t user/_ps | sed '1,/SYMBOL TABLE/d; s/ .* / /; /^$/d' > user/ps.sym
mkfs/mkfs fs.img README user/_cat user/_echo user/_forktest user/_grep user/_init user/_kill user/_ln user/_ls user/_mkdir user/_rm user/_sh user/_stressfs user/_usertests user/_grind user/_wc user/_zombie user/_uniq user/_head user/_test user/_ps
nmeta 46 (boot, super, log blocks 30 inode blocks 13, bitmap blocks 1) blocks 1954 total 2000
balloc: first 891 blocks have been allocated
balloc: write bitmap block at sector 45
mkfs/mkfs fs.img README uniqfile.txt user/_cat user/_echo user/_forktest user/_grep user/_init user/_kill user/_ln user/_ls user/_mkdir user/_rm user/_sh user/_stressfs user/_usertests user/_grind user/_wc user/_zombie user/_uniq user/_head user/_test user/_ps
nmeta 46 (boot, super, log blocks 30 inode blocks 13, bitmap blocks 1) blocks 1954 total 2000
balloc: first 892 blocks have been allocated
balloc: write bitmap block at sector 45
mkfs/mkfs fs.img README headfile.txt user/_cat user/_echo user/_forktest user/_grep user/_init user/_kill user/_ln user/_ls user/_mkdir user/_rm user/_sh user/_stressfs user/_usertests user/_grind user/_wc user/_zombie user/_uniq user/_head user/_test user/_ps
nmeta 46 (boot, super, log blocks 30 inode blocks 13, bitmap blocks 1) blocks 1954 total 2000
balloc: first 892 blocks have been allocated
balloc: write bitmap block at sector 45
qemu-system-riscv64 -machine virt -bios none -kernel kernel/kernel -m 128M -smp 3 -nographic -global virtio-mmio.force-legacy=false -drive file=fs.img,if=none,format=raw,id=x0 -device virtio-blk-device,drive=x0,bus=virtio-mmio-bus.0

xv6 kernel is booting

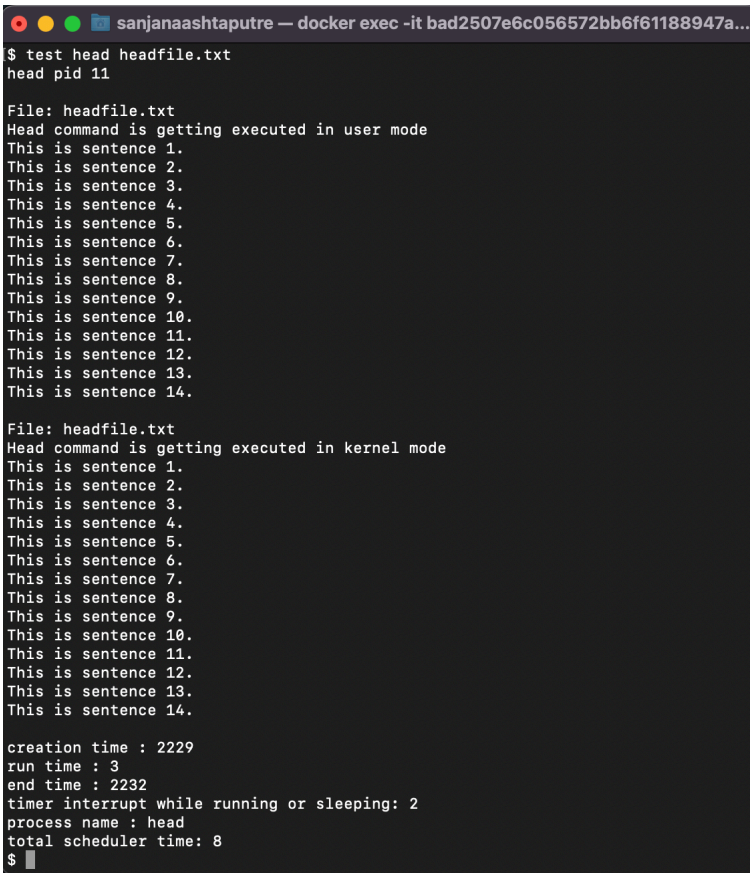
hart 1 starting
hart 2 starting
init: starting sh
$

Input files:

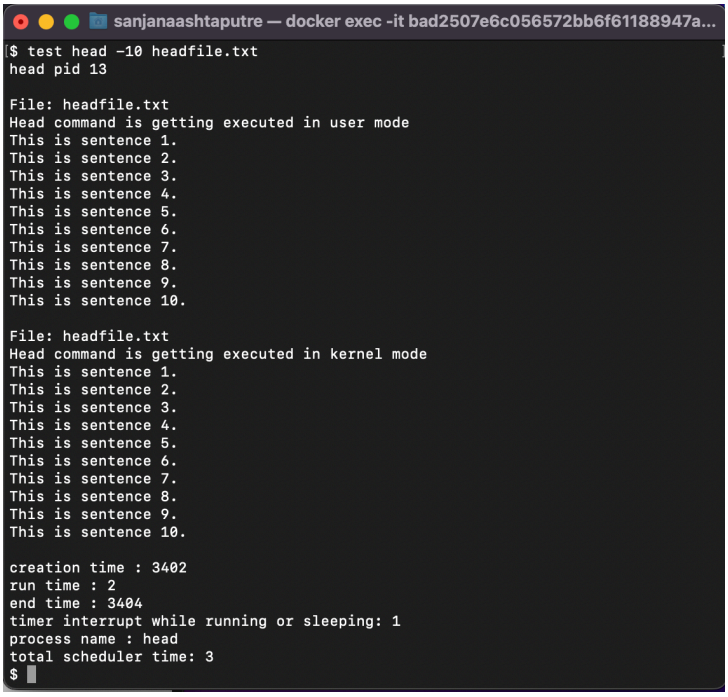


Output: head

test head head file.txt



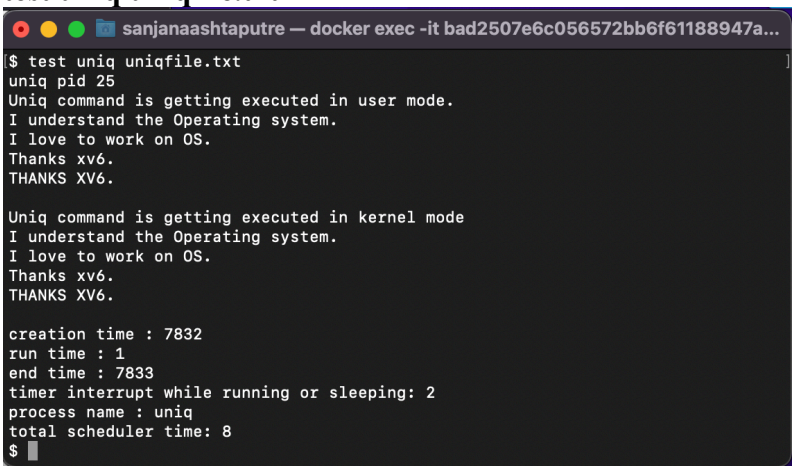
test head -10 headfile.txt



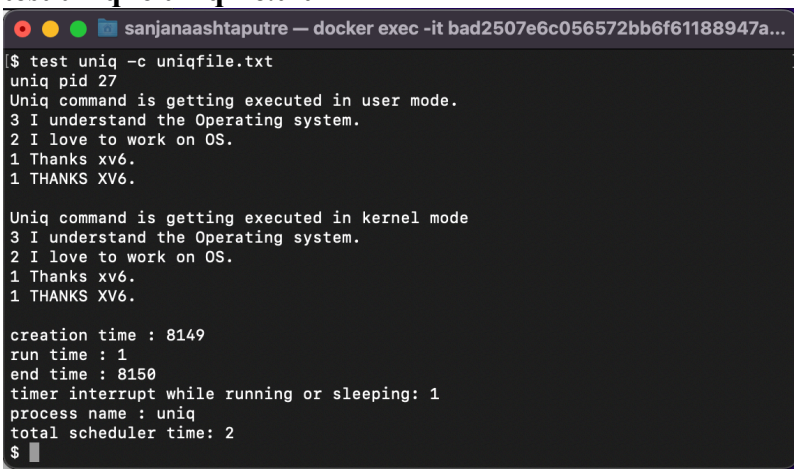
test head -5 headfile.txt headfile2.txt

Output: uniq

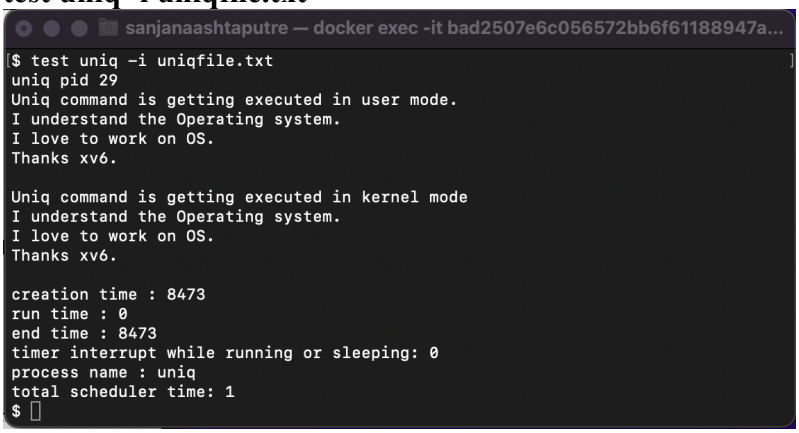
test uniq uniqfile.txt



test uniq -c uniqfile.txt



test uniq -i uniqfile.txt



test uniq -d uniqfile.txt