

ASSIGNMENT 1

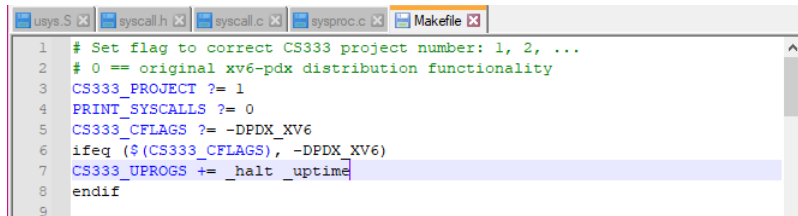
Nama : Atikah Khonsa Salsabila

NIM : 1313619002

Prodi : Ilmu Komputer 2019

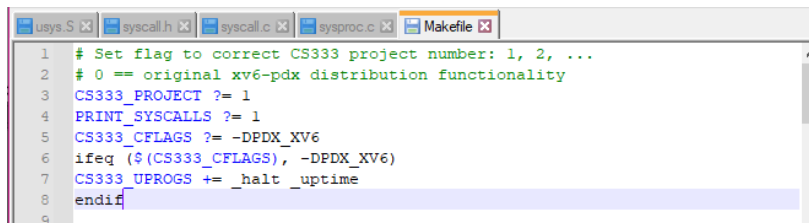
A. Makefile

- Line 3-4



```
1 # Set flag to correct CS333 project number: 1, 2, ...
2 # 0 == original xv6-pdx distribution functionality
3 CS333_PROJECT ?= 1
4 PRINT_SYSCALLS ?= 0
5 CS333_CFLAGS ?= -DPDX_XV6
6 ifeq ($(CS333_CFLAGS), -DPDX_XV6)
7 CS333_UPROGS += _halt _uptime
8 endif
9
```

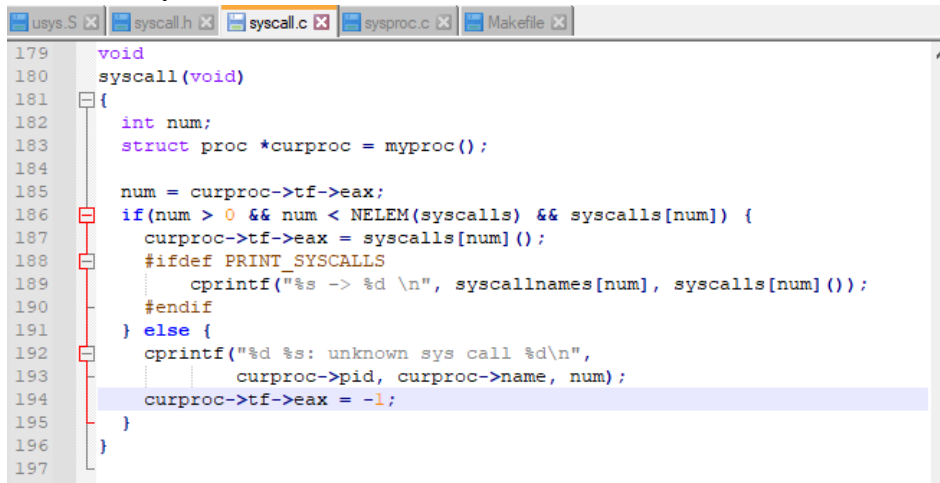
Untuk task 1, line 4 diberi nilai 1



```
1 # Set flag to correct CS333 project number: 1, 2, ...
2 # 0 == original xv6-pdx distribution functionality
3 CS333_PROJECT ?= 1
4 PRINT_SYSCALLS ?= 1
5 CS333_CFLAGS ?= -DPDX_XV6
6 ifeq ($(CS333_CFLAGS), -DPDX_XV6)
7 CS333_UPROGS += _halt _uptime
8 endif
9
```

B. Task 1

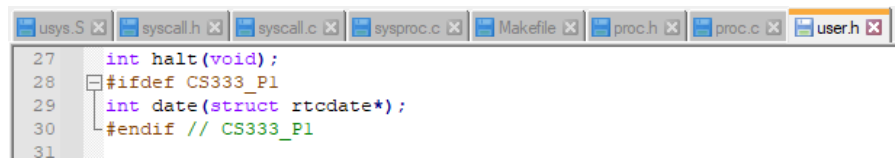
- Syscall.c : Line 188-190



```
179 void
180 syscall(void)
181 {
182     int num;
183     struct proc *curproc = myproc();
184
185     num = curproc->tf->eax;
186     if(num > 0 && num < NELEM(syscalls) && syscalls[num]) {
187         curproc->tf->eax = syscalls[num]();
188         #ifdef PRINT_SYSCALLS
189             cprintf("%s -> %d \n", syscallnames[num], syscalls[num]());
190         #endif
191     } else {
192         cprintf("%d %s: unknown sys call %d\n",
193             curproc->pid, curproc->name, num);
194         curproc->tf->eax = -1;
195     }
196 }
197
```

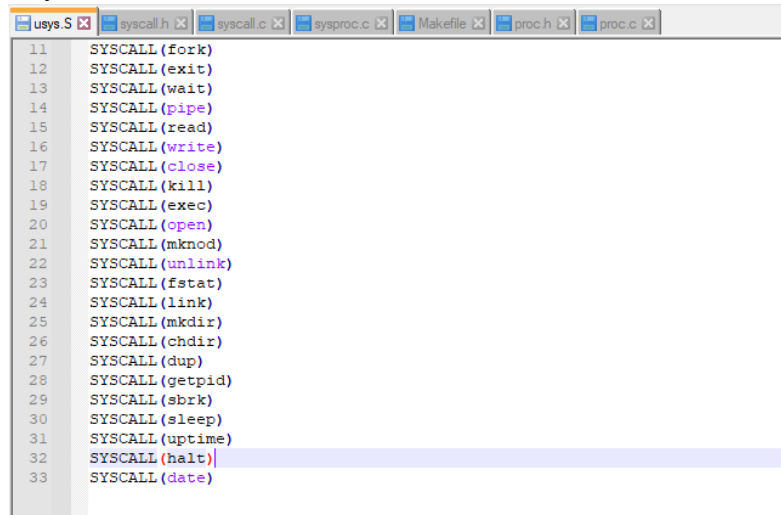
C. Task 2

- User.h : Line 28-30



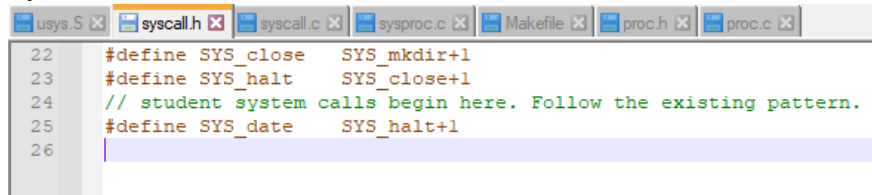
```
27 int halt(void);
28 #ifdef CS333_P1
29 int date(struct rtcdate*);
30 #endif // CS333_P1
31
```

- Usys.S : Line 33



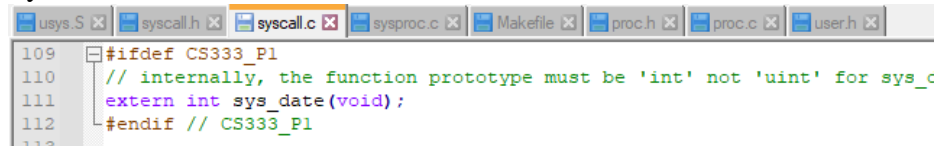
```
11 SYSCALL(fork)
12 SYSCALL(exit)
13 SYSCALL(wait)
14 SYSCALL(pipe)
15 SYSCALL(read)
16 SYSCALL(write)
17 SYSCALL(close)
18 SYSCALL(kill)
19 SYSCALL(exec)
20 SYSCALL(open)
21 SYSCALL(mknod)
22 SYSCALL(unlink)
23 SYSCALL(fstat)
24 SYSCALL(link)
25 SYSCALL(mkdir)
26 SYSCALL(chdir)
27 SYSCALL(dup)
28 SYSCALL(getpid)
29 SYSCALL(sbrk)
30 SYSCALL(sleep)
31 SYSCALL(uptime)
32 SYSCALL(halt)
33 SYSCALL(date)
```

- Syscall.h : Line 25



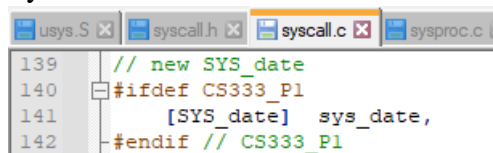
```
22 #define SYS_close SYS_mkdir+1
23 #define SYS_halt SYS_close+1
24 // student system calls begin here. Follow the existing pattern.
25 #define SYS_date SYS_halt+1
26
```

- Syscall.c : Line 109-111



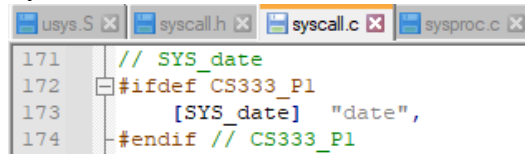
```
109 #ifdef CS333_P1
110 // internally, the function prototype must be 'int' not 'uint' for sys_c
111 extern int sys_date(void);
112 #endif // CS333_P1
113
```

- Syscall.c : Line 139-142



```
139 // new SYS_date
140 #ifdef CS333_P1
141 [SYS_date] sys_date,
142 #endif // CS333_P1
```

- Syscall.c : Line 171-176

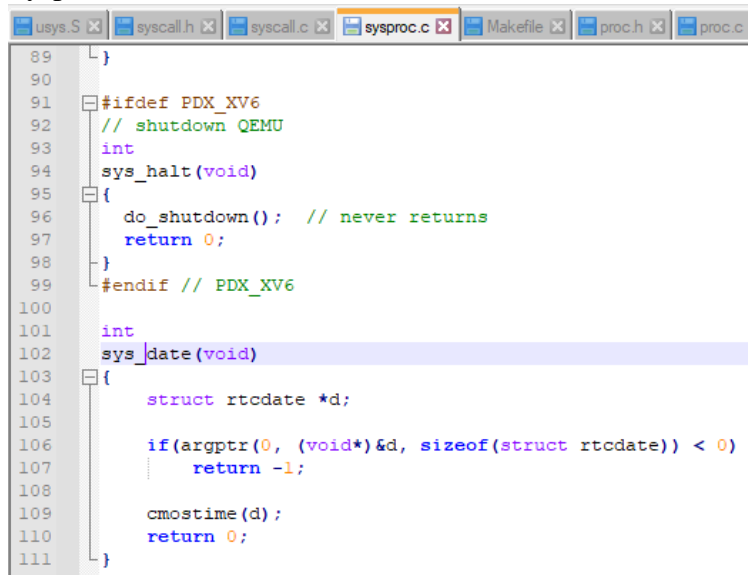


```

171 // SYS_date
172 #ifdef CS333_P1
173     [SYS_date] "date",
174 #endif // CS333_P1

```

- Sysproc.c : Line 101-111

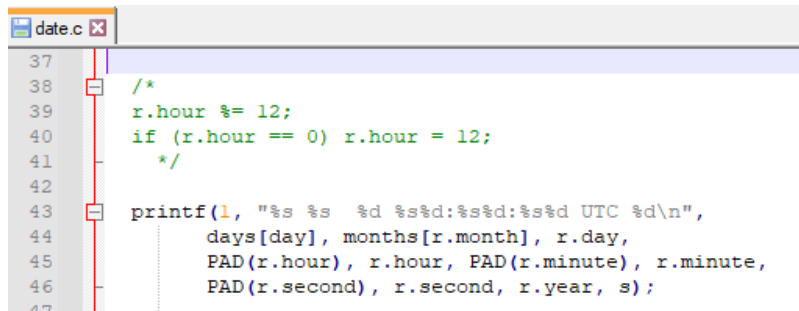


```

89 }
90
91 #ifdef PDX_XV6
92     // shutdown QEMU
93     int
94     sys_halt(void)
95     {
96         do_shutdown(); // never returns
97         return 0;
98     }
99 #endif // PDX_XV6
100
101 int
102 sys_date(void)
103 {
104     struct rtcdate *d;
105
106     if(argptr(0, (void*)&d, sizeof(struct rtcdate)) < 0)
107         return -1;
108
109     cmostime(d);
110     return 0;
111 }

```

- Date.c : Line 38-46



```

37
38 /*
39  r.hour %= 12;
40  if (r.hour == 0) r.hour = 12;
41  */
42
43 printf(1, "%s %s %d %s%d:%s%d:%s%d UTC %d\n",
44         days[day], months[r.month], r.day,
45         PAD(r.hour), r.hour, PAD(r.minute), r.minute,
46         PAD(r.second), r.second, r.year, s);
47

```

D. Task 3

- Proc.h : Line 52

```
usys.S x syscall.h x syscall.c x sysproc.c x Makefile x proc.h x
31     uint ebp;
32     uint eip;
33 };
34
35 enum procstate { UNUSED, EMBRYO, SLEEPING, RUNNABLE, RUNNING, ZOMBIE };
36
37 // Per-process state
38 struct proc {
39     uint sz; // Size of process memory (bytes)
40     pde_t* pgdir; // Page table
41     char *kstack; // Bottom of kernel stack for this process
42     enum procstate state; // Process state
43     uint pid; // Process ID
44     struct proc *parent; // Parent process. NULL indicates no parent
45     struct trapframe *tf; // Trap frame for current syscall
46     struct context *context; // switch() here to run process
47     void *chan; // If non-zero, sleeping on chan
48     int killed; // If non-zero, have been killed
49     struct file *ofile[NOFILE]; // Open files
50     struct inode *cwd; // Current directory
51     char name[16]; // Process name (debugging)
52     uint start_ticks;
53 };
54
```

- Proc.c : Line 152

```
usys.S x syscall.h x syscall.c x sysproc.c x Makefile x proc.h x proc.c x user.h x
147     sp -= sizeof *p->context;
148     p->context = (struct context*)sp;
149     memset(p->context, 0, sizeof *p->context);
150     p->context->eip = (uint)forkret;
151
152     p->start_ticks = ticks;
153     return p;
154 }
```

- Proc.c : Line 567-583

```
date.c x Makefile x proc.c x
563 #elif defined(CS333_P1)
564 void
565 procdumpP1(struct proc *p, char *state_string)
566 {
567     int berlalu_s;
568     int berlalu_ms;
569
570     berlalu_ms = ticks - p->start_ticks;
571     berlalu_s = berlalu_ms / 1000;
572     berlalu_ms = berlalu_ms % 1000;
573
574     char* mulai = "";
575     if(berlalu_ms < 100 && berlalu_ms >= 10)
576         mulai = "0";
577     if(berlalu_ms < 10)
578         mulai = "00";
579
580     cprintf("%d\t%s\t%s%d.%s%d\t%s\t%d\t",
581             p->pid, p->name, " ",
582             berlalu_s, mulai, berlalu_ms,
583             states[p->state], p->sz);
584     return;
585 }
586 #endif
```