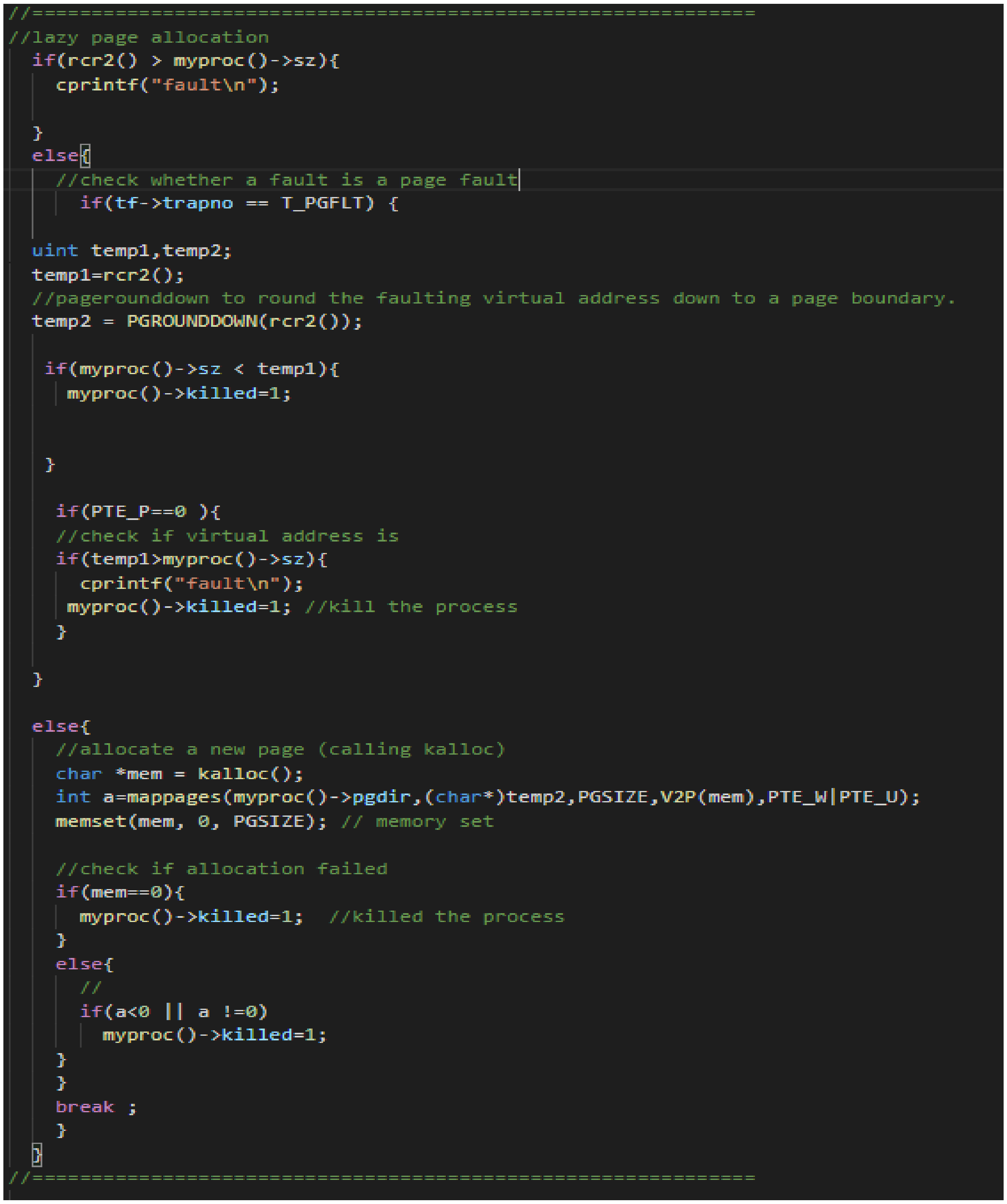
Assignment #5

OPERATING SYSTEM

Group_04

Team members: Tejendra Khatri, Sushil Pandey, Ankur Lamichhane, Solomon Christian



```
//============================================================
//lazy page allocation
  if(rcr2() > myproc()->sz){
    cprintf("fault\n");

  }
  else{
    //check whether a fault is a page fault
      if(tf->trapno == T_PGFLT) {

  uint temp1,temp2;
  temp1=rcr2();
  //pagerounddown to round the faulting virtual address down to a page boundary.
  temp2 = PGROUNDDOWN(rcr2());

   if(myproc()->sz < temp1){
     myproc()->killed=1;


   }

    if(PTE_P==0 ){
    //check if virtual address is
    if(temp1>myproc()->sz){
      cprintf("fault\n");
     myproc()->killed=1; //kill the process
    }

  }

  else{
    //allocate a new page (calling kalloc)
    char *mem = kalloc();
    int a=mappages(myproc()->pgdir,(char*)temp2,PGSIZE,V2P(mem),PTE_W|PTE_U);
    memset(mem, 0, PGSIZE); // memory set

    //check if allocation failed
    if(mem==0){
      myproc()->killed=1;  //killed the process
    }
    else{
      //
      if(a<0 || a !=0)
        myproc()->killed=1;
    }
    }
    break ;
    }
  }
//============================================================
```

# Lazy page allocation Policy and implementation:

xv6 applications ask the kernel for heap memory using sys_sbrk() system call. So, in this assignment we eliminate the page allocation from sys_sbrk() and we should process size by n in sysproc.c. Also, to respond to page fault from user request, we modified code on trap.c. i.e we implemented lazy page allocation as mention above.

# Modified and Created:

1. trap.c:

In trap.c, we allocated the lazy page for which the code is mention above in screenshot. Also, as we eliminated static from mappages in vm.c, we call it in trap.c to have access to it.

2. vm.c:

We eliminate static from int static mappages and called it in trap.c to access it.

3. usertests.c:

As discussed in class and as Jonathan mentioned in class that memtest could not be completed with deferred paging, I commented that part in usertest.c.

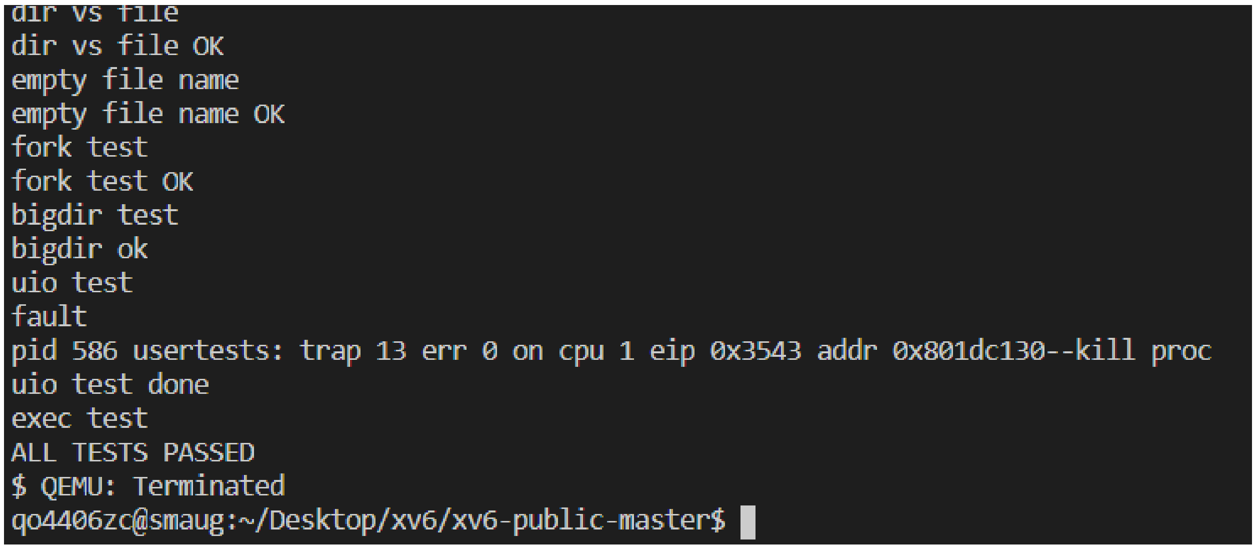
4. sysproc.c:

We eliminated the page allocation and handled allocation and deallocation cases.

# Team Member's Roles:

Each member showed their significant role in the completion of the assignment.

# result of output test:



```
dir vs file
dir vs file OK
empty file name
empty file name OK
fork test
fork test OK
bigdir test
bigdir ok
uio test
fault
pid 586 usertests: trap 13 err 0 on cpu 1 eip 0x3543 addr 0x801dc130--kill proc
uio test done
exec test
ALL TESTS PASSED
$ QEMU: Terminated
qo4406zc@smaug:~/Desktop/xv6/xv6-public-master$
```