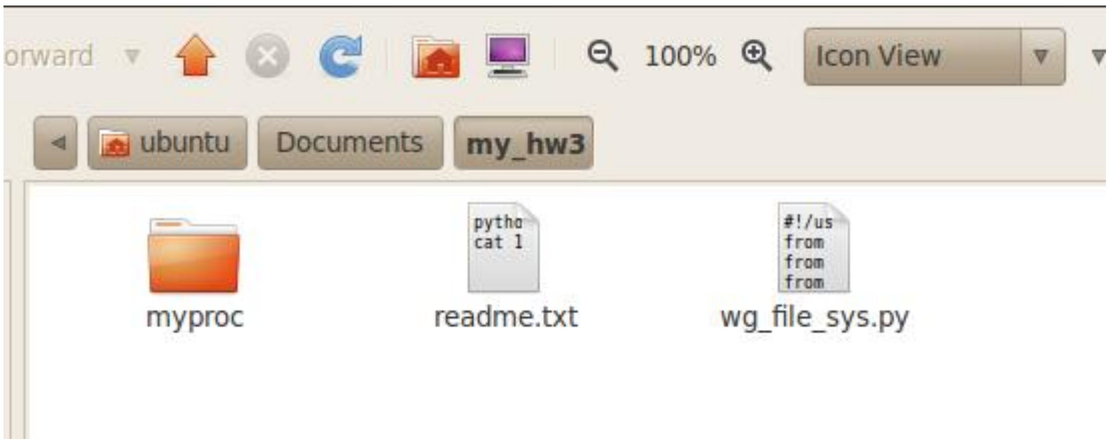
CS6223 - Introduction to Operating Systems.
Assignment 3

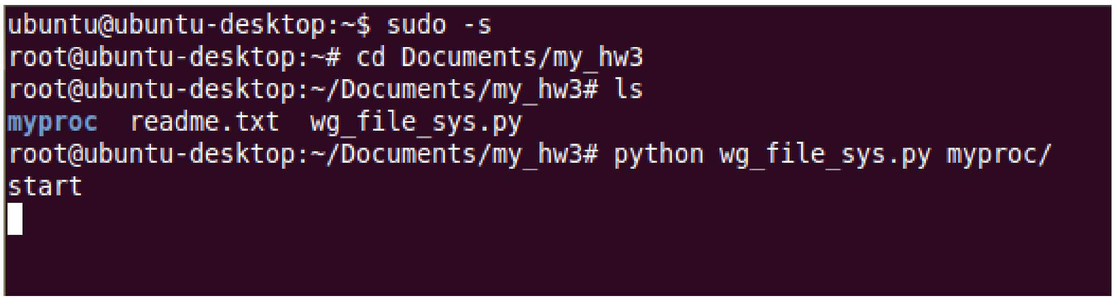
Name: Wei Gu   ID: N14490190

I used python to finish this assignment, and install fusepy in my linux system, so before run my code, you have to install this module on your machine.
fusepy is a Python module that provides a simple interface to FUSE and MacFUSE. It's just one file and is implemented using ctypes.
See details fusepy: https://code.google.com/p/fusepy/

1. - The root directory is myproc that stores basic information of all processes' in your system as files. Each file contain information of a process, filename is the same as process's id. You are free to use any file format to store this information, i.e plain text, json, xml. You would get bonus point if you could create cool feature for this. For example, using multi-levels directories to better organize the processes, or keeping track of both active processes and retired processes.



create a folder called myproc, and mount on the folder: myproc:

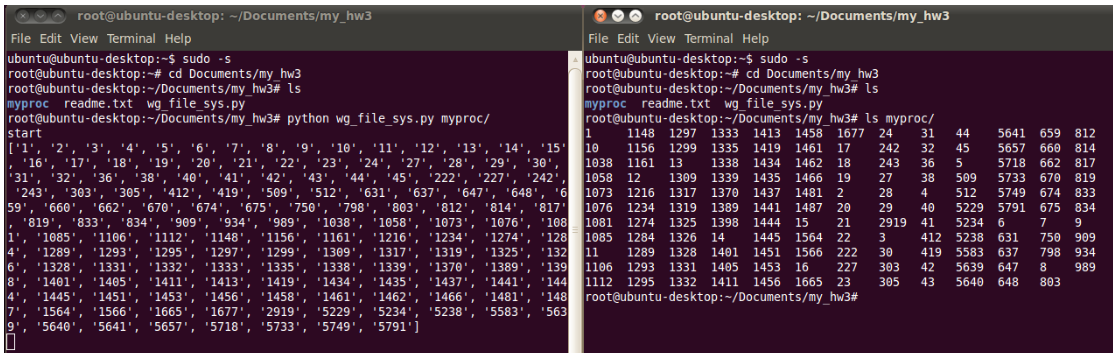


```
ubuntu@ubuntu-desktop:~$ sudo -s
root@ubuntu-desktop:~# cd Documents/my_hw3
root@ubuntu-desktop:~/Documents/my_hw3# ls
myproc  readme.txt  wg_file_sys.py
root@ubuntu-desktop:~/Documents/my_hw3# python wg_file_sys.py myproc/
start
```

2. - "ls myproc" command should return all processes' names.

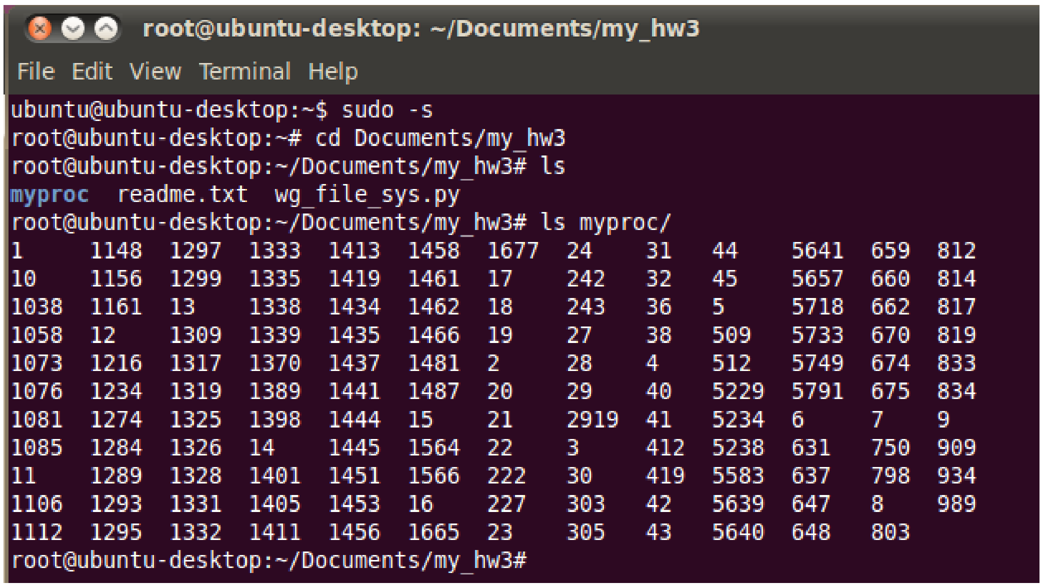
   open another terminal with super user right. My code fetch the process id from '/proc' and filter the useless imformation.

   run ls myproc/:



```
ubuntu@ubuntu-desktop:~$ sudo -s
root@ubuntu-desktop:~# cd Documents/my_hw3
root@ubuntu-desktop:~/Documents/my_hw3# ls
myproc  readme.txt  wg_file_sys.py
root@ubuntu-desktop:~/Documents/my_hw3# python wg_file_sys.py myproc/
start
['1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '11', '12', '13', '14', '15'
, '16', '17', '18', '19', '20', '21', '22', '23', '24', '27', '28', '29', '30',
'31', '32', '36', '38', '40', '41', '42', '43', '44', '45', '222', '227', '242',
 '243', '303', '305', '412', '419', '509', '512', '631', '637', '647', '648', '6
59', '660', '662', '670', '674', '675', '750', '798', '803', '812', '814', '817'
, '819', '833', '834', '909', '934', '989', '1038', '1058', '1073', '1076', '108
1', '1085', '1106', '1112', '1148', '1156', '1161', '1216', '1234', '1274', '128
4', '1289', '1293', '1295', '1297', '1299', '1309', '1317', '1319', '1325', '132
6', '1328', '1331', '1332', '1333', '1335', '1338', '1339', '1370', '1389', '139
8', '1401', '1405', '1411', '1413', '1419', '1434', '1435', '1437', '1441', '144
4', '1445', '1451', '1453', '1456', '1458', '1461', '1462', '1466', '1481', '148
7', '1564', '1566', '1665', '1677', '2919', '5229', '5234', '5238', '5583', '563
9', '5640', '5641', '5657', '5718', '5733', '5749', '5791']
```

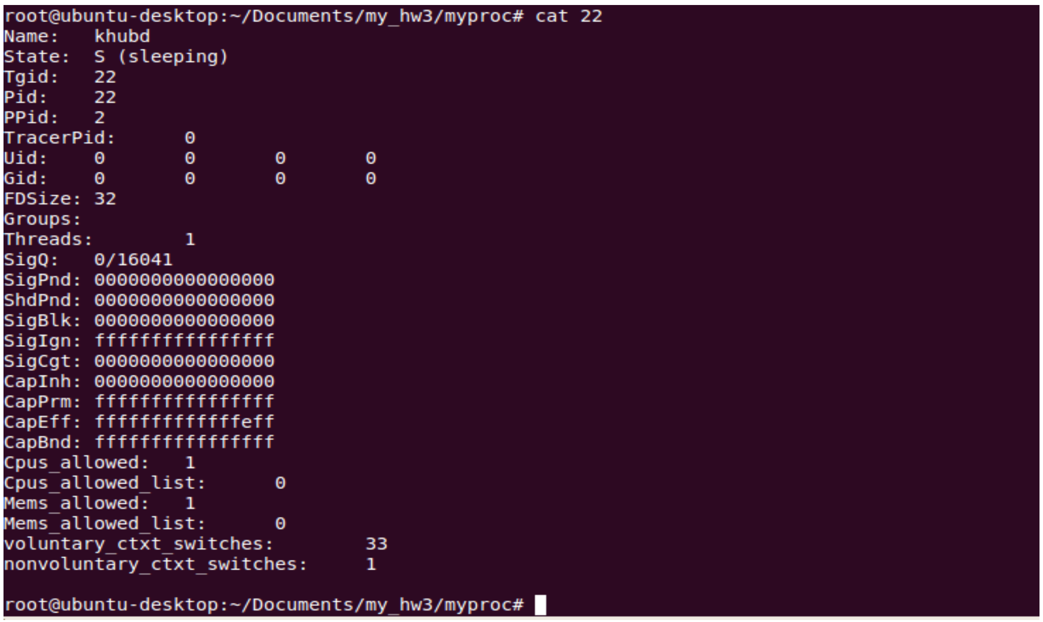
The right part is new terminal, which lists all the running process:



```
ubuntu@ubuntu-desktop:~$ sudo -s
root@ubuntu-desktop:~# cd Documents/my_hw3
root@ubuntu-desktop:~/Documents/my_hw3# ls
myproc  readme.txt  wg_file_sys.py
root@ubuntu-desktop:~/Documents/my_hw3# ls myproc/
1     1148  1297  1333  1413  1458  1677  24    31    44    5641  659  812
10    1156  1299  1335  1419  1461  17    242   32    45    5657  660  814
1038  1161  13    1338  1434  1462  18    243   36    5     5718  662  817
1058  12    1309  1339  1435  1466  19    27    38    509   5733  670  819
1073  1216  1317  1370  1437  1481  2     28    4     512   5749  674  833
1076  1234  1319  1389  1441  1487  20    29    40    5229  5791  675  834
1081  1274  1325  1398  1444  15    21    2919  41    5234  6     7    9
1085  1284  1326  14    1445  1564  22    3     412   5238  631   750  909
11    1289  1328  1401  1451  1566  222   30    419   5583  637   798  934
1106  1293  1331  1405  1453  16    227   303   42    5639  647   8    989
1112  1295  1332  1411  1456  1665  23    305   43    5640  648   803
root@ubuntu-desktop:~/Documents/my_hw3#
```

3. - Reading file would yield all information associated with the corresponding process.

Take pid 22 for an example, cd myproc and run cat 22 in new terminal, it will print the status of this process, it get the information from '/proc/22/status':



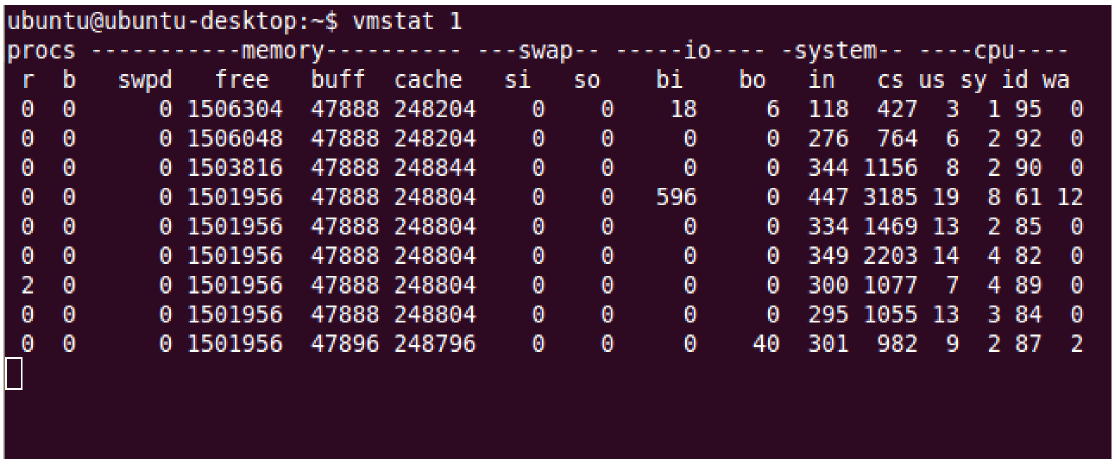
```
root@ubuntu-desktop:~/Documents/my_hw3/myproc# cat 22
Name:	khubd
State:	S (sleeping)
Tgid:	22
Pid:	22
PPid:	2
TracerPid:	0
Uid:	0	0	0	0
Gid:	0	0	0	0
FDSize:	32
Groups:
Threads:	1
SigQ:	0/16041
SigPnd:	0000000000000000
ShdPnd:	0000000000000000
SigBlk:	0000000000000000
SigIgn:	ffffffffffffffff
SigCgt:	0000000000000000
CapInh:	0000000000000000
CapPrm:	ffffffffffffffff
CapEff:	fffffffffffffeff
CapBnd:	ffffffffffffffff
Cpus_allowed:	1
Cpus_allowed_list:	0
Mems_allowed:	1
Mems_allowed_list:	0
voluntary_ctxt_switches:	33
nonvoluntary_ctxt_switches:	1

root@ubuntu-desktop:~/Documents/my_hw3/myproc#
```

4. -Cool feature:

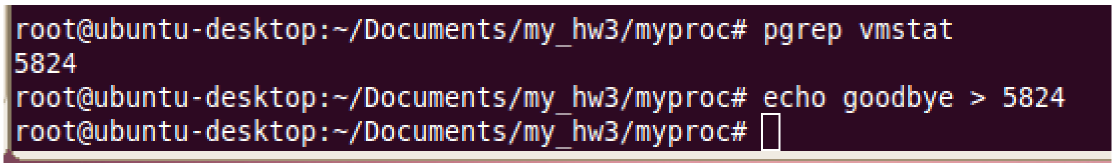
   Add a new function which can kill process by pid.

   (1) Open a new terminal, and run vmstat 1, a new process started.



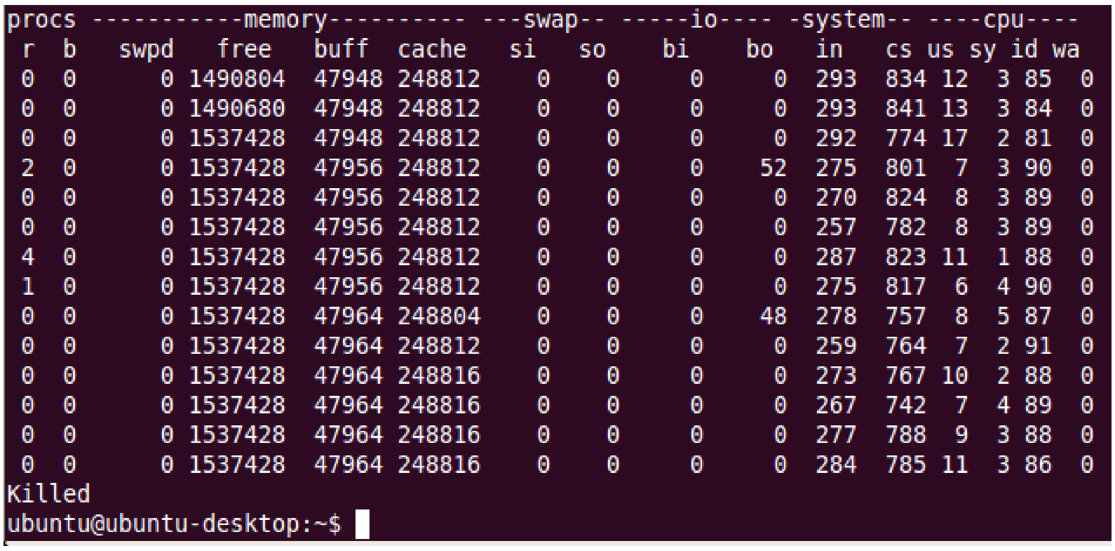
```
ubuntu@ubuntu-desktop:~$ vmstat 1
procs -----------memory---------- ---swap-- -----io---- -system-- ----cpu----
 r  b   swpd   free   buff  cache   si   so    bi    bo   in   cs us sy id wa
 0  0      0 1506304  47888 248204    0    0    18     6  118  427  3  1 95  0
 0  0      0 1506048  47888 248204    0    0     0     0  276  764  6  2 92  0
 0  0      0 1503816  47888 248844    0    0     0     0  344 1156  8  2 90  0
 0  0      0 1501956  47888 248804    0    0   596     0  447 3185 19  8 61 12
 0  0      0 1501956  47888 248804    0    0     0     0  334 1469 13  2 85  0
 0  0      0 1501956  47888 248804    0    0     0     0  349 2203 14  4 82  0
 2  0      0 1501956  47888 248804    0    0     0     0  300 1077  7  4 89  0
 0  0      0 1501956  47888 248804    0    0     0     0  295 1055 13  3 84  0
 0  0      0 1501956  47896 248796    0    0     0    40  301  982  9  2 87  2
```

   (2) Get the pid and kill it, use my defined command: goodbye.



```
root@ubuntu-desktop:~/Documents/my_hw3/myproc# pgrep vmstat
5824
root@ubuntu-desktop:~/Documents/my_hw3/myproc# echo goodbye > 5824
root@ubuntu-desktop:~/Documents/my_hw3/myproc# 
```

   (3) It was killed.



```
procs -----------memory---------- ---swap-- -----io---- -system-- ----cpu----
 r  b   swpd   free   buff  cache   si   so    bi    bo   in   cs us sy id wa
 0  0      0 1490804  47948 248812    0    0     0     0  293  834 12  3 85  0
 0  0      0 1490680  47948 248812    0    0     0     0  293  841 13  3 84  0
 0  0      0 1537428  47948 248812    0    0     0     0  292  774 17  2 81  0
 2  0      0 1537428  47956 248812    0    0     0    52  275  801  7  3 90  0
 0  0      0 1537428  47956 248812    0    0     0     0  270  824  8  3 89  0
 0  0      0 1537428  47956 248812    0    0     0     0  257  782  8  3 89  0
 4  0      0 1537428  47956 248812    0    0     0     0  287  823 11  1 88  0
 1  0      0 1537428  47956 248812    0    0     0     0  275  817  6  4 90  0
 0  0      0 1537428  47964 248804    0    0     0    48  278  757  8  5 87  0
 0  0      0 1537428  47964 248812    0    0     0     0  259  764  7  2 91  0
 0  0      0 1537428  47964 248816    0    0     0     0  273  767 10  2 88  0
 0  0      0 1537428  47964 248816    0    0     0     0  267  742  7  4 89  0
 0  0      0 1537428  47964 248816    0    0     0     0  277  788  9  3 88  0
 0  0      0 1537428  47964 248816    0    0     0     0  284  785 11  3 86  0
Killed
ubuntu@ubuntu-desktop:~$ 
```

Source Code:

```
#!/usr/bin/env python

from __future__ import with_statement

from errno import ENOENT

from stat import S_IFDIR, S_IFREG

from sys import argv, exit

from time import time

import os


from fuse import FUSE, FuseOSError, Operations, LoggingMixIn, fuse_get_context


def check(s):    # check if only made up of digit

    return s[1:].isdigit()


class my_file_sys(LoggingMixIn, Operations):

        def __init__(self):    # start mounting

                print 'start'


        def access(self, path, mode):

                print 'access',path

                full_path = '/proc'+path

                if not os.access(full_path, mode):

                        raise FuseOSError(errno.EACCES)


        def chmod(self, path, mode):

                full_path = '/proc'+path

                return os.chmod(full_path, mode)


        def chown(self, path, uid, gid):
```

```
        full_path = '/proc'+path
        return os.chown(full_path, uid, gid)

    def getattr(self, path, fh=None):
        uid, gid, pid = fuse_get_context()
        if path == '/':
            st = dict(st_mode=(S_IFDIR | 0755), st_nlink=2)
        elif check(path):
            size = 10000
            st = dict(st_mode=(S_IFREG | 0666), st_size=size)
        else:
            raise FuseOSError(ENOENT)
        st['st_ctime'] = st['st_mtime'] = st['st_atime'] = time()
        return st

    def read(self, path, size, offset, fh):
        encoded = lambda x: ('%s\n' % x).encode('utf-8')
        if check(path):
            print 'read....',path
            fh=os.open('/proc'+path+'/status',os.O_RDWR)    # get the status
            offset=0
            length=10000
            os.lseek(fh,offset,os.SEEK_SET)
            tmp_read = os.read(fh,length)
            return encoded(tmp_read)
        raise RuntimeError('unexpected path: %r' % path)

    def readdir(self, path, fh):    # for ls myproc
        return self.process_list()
```

```
def process_list(self):    # get the process id
    pids = []
    for subdir in os.listdir('/proc'):
            if(subdir.isdigit()):
                pids.append(subdir)
    print pids
    return ['.','..']+pids

def open(self, path, flags):
            full_path = '/proc'+path
            print 'open.......',full_path,flags
            return os.open(full_path+'/status', flags)

def create(self, path, mode, fi=None):
        print 'create...'
        full_path = '/proc'+path
        return os.open(full_path, os.O_WRONLY | os.O_CREAT, mode)

def write(self, path, data, offset, fh):
        print path,data
        if data[:7]=='goodbye':            # kill by pid
            os.kill(int(path[1:]),9)
        return 10

def truncate(self, path, length, fh=None):
        print 'truncate',path
        length=1000
        with open('/proc'+path+'/status', 'r+') as f:
            f.truncate(length)
    # Disable unused operations:
    access = None
    getxattr = None
    listxattr = None
    opendir = None
    releasedir = None
    statfs = None


if __name__ == '__main__':
    if len(argv) != 2:
        print('usage: %s <mountpoint>' % argv[0])
        exit(1)

    fuse = FUSE(my_file_sys(), argv[1], foreground=True)
```