ECE 5725 HW1 Group work
yc2329 Yi Chen | my432 Mingda Yang

(Attached are our work for question 1 to 4 separately)
Q1 - 4 Mingda Yang
1.

2.

3. For testing I remove the read permission for test. Later on I changed it to 744 which only owner can read write and execute, others can only read.

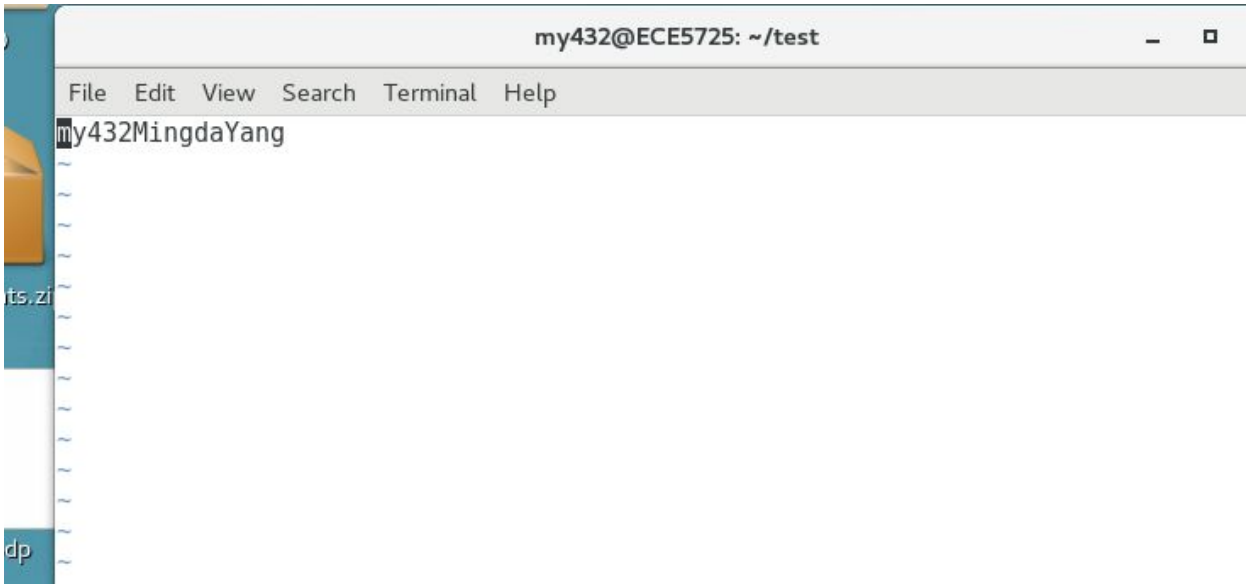
```
The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Mon Sep  4 16:01:21 2017 from cit-phillips-10.citlabs.cornell.edu
my432@ECE5725:~ $ ls
test
my432@ECE5725:~ $ ls -l
total 4
drwxr-xr-x 2 my432 students 4096 Sep  4 16:29 test
my432@ECE5725:~ $ chmod test -r
my432@ECE5725:~ $ ls -l
total 4
d-wx--x--x 2 my432 students 4096 Sep  4 16:29 test
my432@ECE5725:~ $ date
Thu Sep  7 11:39:20 EDT 2017
my432@ECE5725:~ $ whoiam
-bash: whoiam: command not found
my432@ECE5725:~ $ whoami
my432
```

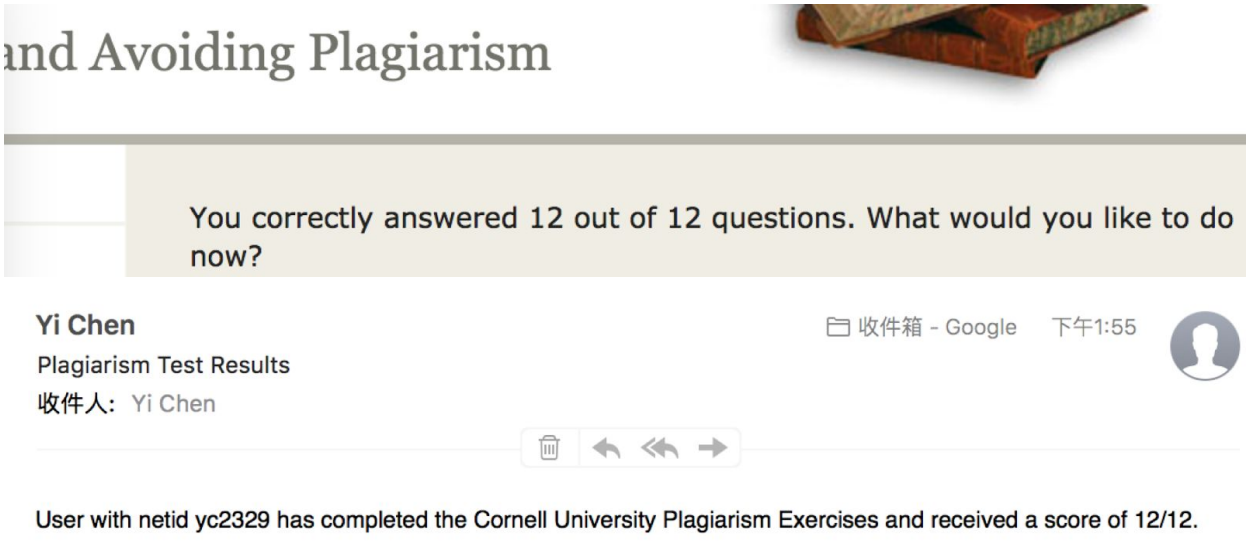
```
my432@ECE5725:~ $ ls -l
total 4
drwxr--r-- 2 my432 students 4096 Sep  7 12:13 test
```

4.

```
my432@ECE5725:~/test $ chmod 744 HW1.txt
my432@ECE5725:~/test $ vi HW1.txt
my432@ECE5725:~/test $ ls -l
total 4
-rwxr--r-- 1 my432 students 16 Sep  7 11:44 HW1.txt
```



//////////
Q1 - 4 Yi Chen
1.



2.

3.

```
[yc2329@ECE5725:~ $ whoami
 yc2329
[yc2329@ECE5725:~ $ date
 Mon Sep  4 14:13:40 EDT 2017
[yc2329@ECE5725:~ $ mkdir test
[yc2329@ECE5725:~ $ ls
 test
[yc2329@ECE5725:~ $ cd test/
[yc2329@ECE5725:~/test $ ls
[yc2329@ECE5725:~/test $ cd
[yc2329@ECE5725:~ $ cd ../
[yc2329@ECE5725:/home $ ls
 abc248  cb674   dy85    jfs9    ks763       pi      rx65    ts755   yl2684
 abh222  cjs342  dys27   jh2635  lost+found  pl557   sl2928  vmt28   ys775
 aif33   cl2445  fjm83   jl3449  lx238       pq32    sl2947  wj236   yw883
 aj373   csw73   hc937   jm2424  lz455       ql65    ssd56   wm226   yy757
 amb633  cz382   hl887   jmw483  mh2387      qq39    ssw74   xc374   zl579
 arb392  dg566   hw622   jw2299  mny8        rd542   sw889   xx257   zz488
 bt346   dh468   ijh6    jz863   my432       rs2364  tc464   xy363
 btj28   djk289  jc2649  jzw8    nm594       rt446   tmb233  yc2329
 bx64    dk562   jcc384  ks2263  nz248       rw564   trd44   yh772
[yc2329@ECE5725:/home $ cd y
 yc2329/ yh772/  yl2684/ ys775/  yw883/  yy757/
[yc2329@ECE5725:/home $ chmod 700 yc2329/
[yc2329@ECE5725:/home $ ls -la /home |grep yc2329/
[yc2329@ECE5725:/home $ ls -la /home |grep yc2329
 drwx------  4 yc2329 students  4096 Sep  4 14:17 yc2329
[yc2329@ECE5725:/home $ cd yc2329/
[yc2329@ECE5725:~ $ passwd
 Changing password for yc2329.
[(current) UNIX password:
[Enter new UNIX password:
[Retype new UNIX password:
 Bad: new and old password are too similar
[Enter new UNIX password:
[Retype new UNIX password:
 passwd: password updated successfully
```

4.

```
[yc2329@ECE5725:~ $ cd test/
[yc2329@ECE5725:~/test $ touch HW1.txt
[yc2329@ECE5725:~/test $ ls
 HW1.txt
[yc2329@ECE5725:~/test $ cat HW1.txt
[yc2329@ECE5725:~/test $ vi HW1.txt
[yc2329@ECE5725:~/test $ cat HW1.txt
 yc2329 Yi Chen
[yc2329@ECE5725:~/test $ chmod 640 HW1.txt
[yc2329@ECE5725:~/test $ ls -l
 total 4
 -rw-r----- 1 yc2329 students 15 Sep  4 14:34 HW1.txt
```

//////////

5.

Permission 777*(rwxrwxrwx)*:

Allow everyone including owner, group and others to read, write and execute. It is very dangerous because anyone can write malicious code to it.

Permission 644*(rw-r--r--)*:

The owner can read and write a file, while others can only read the file.

Permission 700*(rwx------)*:

Only the owner can read, write and execute a file. No one else has access.

6. What were two key events that led to the proliferation of early Unix systems and paved the way for the eventual development of Linux?

1) In 1983, Sep 27, Richard Stallman initiated the GNU project, which is the first key event. The GNU project led to a free Unix like operating system.
2) Minix created by Andrew S. Tanebaum is the 2nd key event. It was first released in 1987, initially target as educational purpose.

7. Explain what the 'df' command does. Using the ECE5725 server, show the output of this command and explain the size settings for the /home entry. Use the appropriate flags on the df command to show the data in a readable format.

df command displays the amount of disk space available on the file system containing each file name argument. If no file name is given, the space available on all currently mounted file systems is shown.

8. Give definitions for Hard Real Time (HRT) and Soft Real Time (SRT) systems. Give an example of an application requiring HRT. Give an example of a system requiring SRT.

(Source from: WhatIs.com

https://stackoverflow.com/questions/17308956/differences-between-hard-real-time-soft-real-time-and-firm-real-time
https://www.adi.com/technology/tech-apps/what-are-soft-and-hard-real-time-applications/)

Hard Real Time: A hard real-time system (also known as an immediatereal-time system) is hardware or software that must operate within the confines of a stringent deadline.

An application requiring HRT: A Full Authority Digital Engine Controller (FADEC) controls the activities of an aircraft jet engine. The FADEC design mandates particular timing requirements. For example, if the FADEC senses that a turbine drive shaft has broken, then the FADEC must respond with a damage mitigating action in a predetermined time.

Soft Real Time: Firm/soft real time systems can miss some deadlines, but eventually performance will degrade if too many are missed.

A system requiring SRT: airline reservation systems.

9. Can Linux be used as a RTOS? Give a possible application where Linux would work as an RTOS? Give a second application where Linux might NOT work as an RTOS?
Linux itself is not a real time operating system but a timesharing operating system, however there are some real-time applications that linux can run, with some real time extension added. There are several main difference that identify Linux as non RTOS. The most importatnt one is the default scheduling policy in Linux is time shared. Other include different memory models, while RTOS has all memory models in one memory space, at the same time Linux have user space and kernel space and applications run in user space, hardware interaction is in kernel space.
There are lots of works have been done to add real time features to linux. And now with some proper extension, some of the real time applications can be run in Linux
RTOS applications: There are three ways to implement RTOS 1) Microkernel approach, 2)Monolithic Approach, 3) Decouple Approach. Applications like LynxOS or RTLinux are all RTOS examples. And an example of RT extension is RTAI(Real time application interface)
Non RTOS applications: Linux's scheduling on kernel(time sharing)

10. In class, we discussed how the controllers for the SpaceX Falcon 9 rocket engines run Linux. How did the SpaceX researchers, engineers, and developers insure that their designs for these motors would be fault tolerant

Each of the 9 Merlin rocket motors is controlled by 3 linux embedded controllers, and the whole rocket stacks is controlled by 3 overload processors. The reason is by using triple modulo redundancy. They run the same code to control the rocket and vote on every steps, if all agreed on the decision, then process can proceed. Otherwise, if only one is vote no, then warning will show, and process will still run.