CSCI 3431.1 Fall, 2017 – Assignment 1 Operating Systems

0.1 Practice Exercises

1. [2 points] what is the difference between timesharing and multiprogramming systems?

Solution:

From the end user's view, the multiprogramming system can provide to just one user at the same time, one end user can provide a couple of tasks and bundle them into a batch, the computer can handle these tasks on sequence. Moreover, if one task needs some IO operation which will be a time-consuming task, the OS will start to handle the next task during waiting responses from IO devices.

The timesharing system can provide the ability to serve more than one end user at the same time. Although the computer handle the only one requirement at one moment, but the switch is so quickly, so each end user feel that the computer is controlled by themselves.

So, the big difference between these two types of operation system is that timesharing system become interactive.

2. [4 points] you would like to install and run both Windows XP and a REDHAT distribution of Linux on your computer. Discuss how your system can allow a choice of OS from which to boot?

Solution:

A specific program named bootstrap program will be loaded during power on or reboot which is typically stored in ROM or EPROM or known as firmware according to introduced in lecture. This program will make a choice to load which operation system's kernel and start execution.

According to https://en.wikipedia.org/wiki/Master_boot_record, the master boot record is always located at cylinder zero, head zero and section one on the hard disk or removable disk. It means that MBR is a "starting point" about how to proceed with boot process and load the operation system.

As a result, if the MBR is written with Windows XP' loader program, after power on or reboot, the computer will start XP system, otherwise, a REDHAT will be started.

3. [2 points] what is the main advantage for an OS designer of using a virtual-machine architecture? What is the main advantage for a user?

Solution:

For the OS designer, using a virtual-machine architecture will reduce the consideration about the hug differences among different bare hard wares. Generally, in order to design a new operation system, how to ensure it works well on difference hard ware will be a challenge. For example, there are 3 different processor architectures: ARM, Intel and MIPS in order to design a mobile phone operation system.

In the view of a user, the main advantage is that the interface will be unified by virtual-machine. For software developers, the difference about how to handle process, threads or others can be hidden behind virtual-machine. For the end user, their favourite software applications can be installed and used anywhere.

4. [3 points] what type of multiplexing (time, space, or both) can be used for sharing the following resources: CPU, RAM, Network card, Printer, Keyboard and display

Solution:

CPU can be used for time multiplexing because compare to IO devices, the speed of CPU is too high, it is possible to share CPU capability among different processes.

RAM can be used for space multiplexing because generally the RAM size is big enough to load different processes at the same time. But time multiplexing for RAM is impossible because CPU's speed is higher than RAM, so at the same time, only one process can be handled by CPU.

Network card can use time and space multiplexing because different applications need different communication band width through network card.

Printer can use time and space multiplexing because generally there is a cache in modern printer. The printer can print something and receiving another jobs at the same time.

Keyboard can use time multiplexing because most of operation system use interrupt mechanism to handle input from keyboard.

Display can use for time and space multiplexing because generally there is a cache in display which is like a printer.

5. [4 points] there are several design goals in building an operating system, for example, resource utilization, timelines, robustness, and so on.

a. Give an example of two design goals that may contradict one another. Explain.

Solution:

User friendly and performance is an example that may contradict one another.

In order to provide a user friendly interface is a key feature needs to be considered during design an operation system. A popular way is to provide a GUI system for end user with which end user can use mouse to complete most of the tasks. Mac OS and Windows is a good example for this.

On the other side, the performance is another key factor needs to be considered. With a high performance means that one OS should load less drives, for example, graphic chip drives. This is to say in order to provide a user friendly GUI interface, the operation system must load a couple of drives after booting which let the OS to be slower and slower.

This is why some batch tasks must run command line environment rather than GUI.

As a result, most of operation system provide several modes for end user: bash shell or GUI.

b. What are the trade-offs inherent in designing OS for handheld computers?

Solution:

Performance. Those handheld computers generally means that there is small memory or not so powerful CPU. So, the OS designer need to consider how to utilise these resources more efficiently.

Hiding hardware difference. Not like personal computer or servers, the industry standard about handheld computers can vary due to different manufactures. For example, some devices are designed to use ARM chips in order to save energy and long availability. Others use x86 architecture in order to have the most compatible application software.

User interface. Some handheld computers have a small display screen, but sometimes, a bigger font is a must. So, the OS must provide more display options for different scenarios.

6. [11 points] The Android operating system is not only the most popular mobile OS [Morra J., 2016] but also a widely used OS for the Internet of Things. Describe (in at most 1 page) the Android Software Architecture and highlight key features that contribute to the success of this system. Please make sure you cite your work and use at least 3 references.

Solution:

The Internet of Things (IoT) is the network of physical objects that contain embedded technology to communicate and sense or interact with their internal states or the external environment [1]. This is to say that everything has the ability to connect to network especially to internet. Considering a vehicle, for example, the heating system will start to work in order to warm the car when the owner sitting down before kitchen table for the breakfast. There is a sensor in the chair, according to the weight the smart chair can recognize who is sitting. For example, if the wife is recognized, the smart chair will send a signal to the car, the car will start to heating and begin to calculate the shortest path to send the baby to day care office.

In order to support the complex functions and features in the IoT, an operation system is a must. Now more and more solutions select Android as their choice.

Android system architecture contains several components: application framework, binder IPC proxies, system service, HAL and a Linux kernel [2]. According to the introduction about Linux kernel layer from Google, Android uses a specific version of the Linux kernel and add some specific additional features such as Low Memory Killer, IPC driver and others which are important for a mobile embedded platform [3]. The challenge for IoT devices manufactory to select an operation system is that the candidate OS must have the capability to cover the whole scopes and scales with different smart devices. For example, a smart chair will need less functions than a smart car. The android provides a Modular kernel solution for this requirement. In the Android 8.0, the device kernel splits into System-on-Chip, device and board-specific deliverables. As a result, different manufactures can provide their own solution based on the Android [4].

There are a couple of features in Android which can be utilised by IoT devices as below:

- Connectivity [5]. Android support both Wi-Fi and Bluetooth communication. Android 8.0 provides APIs that allow device providers to customize the paring request dialog when trying to pair with companion devices over Bluetooth, BLE and Wi-Fi [6]. This feature is very import function in order to deploy IoT devices because the use case of deployment are typically different with different devices. For example, some device do not have display, some devices have a limit and very small LED display, so we could not uniform the way how to connect to network. As a result, a customizing methodology become a must, Android support this future, so it will be easier to build the IoT devices based on Android.
- Android Enterprise [7]. The enterprise market is a huge market and it is very important for the domain of IoT. When developing and deploying the IoT solutions into an enterprise environment, a couple of things needs to be considered which is more complex than in personal market. For example, all of enterprise customers need a security and management solution, if the OS level can support this features, the cost will be down sharply. This is why Android is a good option as operation system for IoT industry.
- Updated Java language support [8]. The Java language become more popular in recent years, this means that the employers can find the Java developers easily in the market. In the view of IoT, the lifecycle is a very important factor, because there are so many competitors. On the other hand, there are a couple of open-source for Java solution in different domains, in order to build a product efficiently, selecting well-known and with good performance open-source frameworks or tools is a quick way for releasing the product. For example, Android support JDK8, it means that as a developer, the new API for handle date and time become easy with new java.tim package. Using Lambada expression makes the Java source code more efficient.

In conclusion, Android as an operation system has a couple of key features in order to support IoT devices and solutions because Android has a good architecture which supports a wide range of devices and solutions. On the other hand, Android has a couple features from kernel to application layer which is a key component for IoT devices, these features have been completed, tested and embedded in Android different layers.

[1] http://www.gartner.com/it-glossary/internet-of-things/

[2] https://source.android.com/devices/architecture/

[3] https://source.android.com/devices/architecture/kernel/

[4] https://source.android.com/devices/architecture/kernel/modular-kernels

[5] https://developer.android.com/about/versions/oreo/android-8.0.html

[6] https://developer.android.com/guide/topics/connectivity/companion-device-pairing.html

[7] https://developer.android.com/work/index.html

[8] https://developer.android.com/studio/write/java8-support.html

7. [4 points] a multiprocessor with eight processors has 20 attached tape drives. There is a large number of jobs submitted to the system that each require a maximum of four tape drives to complete

execution. Assume each job starts running with only three tape drives for a long period before requiring the fourth tape drive for a short period toward the end of its operation. Also assume an endless supply of such jobs.

a. Assume the scheduler in the OS will not start job unless there are four tape drives available. When a job is started, four drives are assigned immediately and are not released until the job finishes. What is the maximum number of jobs that can be in progress at once? What are the maximum and minimum number of tape drives that may be left idle as a result of this policy?

Solution:

The maximum number of jobs that can be in progress at once is 5. Because 20/4=5.

The maximum number of tape drives that may be left idle as a result of this policy is 15. This will occur during a short period toward the end of each job. All of the 5 jobs just use the fourth tape drive.

The minimum number is 5 which occur during the beginning stage. Each job utilise three tape drives.

b. Suggest an alternative policy to improve tape drive utilization and at the same time avoid system deadlock. What is the maximum number of jobs that can be in progress at once? What are the bounds on the number of idling tape drives?

Solution:

Another alternative policy may be designed to just start a job with three tape drives and add the fourth available tape drive when requires.

The maximum number of jobs that can be in progress at once is 6.

The minimum number of idling tape drives is 2 which will occur each 6 jobs have three tape drives. As a result, there are 2 idling ones.

The maximum number is 12 which will occur, the first and second job get the fourth tape drives, the 3rd 4th 5th 6th jobs have to wait for their fourth tape drives.

Self-evaluation Please answer the following questions:

1. [1 points] Were you able to complete this assignment? What grade are you expecting? Please justify.

Yes, I did this assignment by myself on the time, and I feel my solution has reached the expectation from the teacher.

2. [2 points] Describe 2-3 challenges you faced while completing this assignment. How did you tackle those challenges?

Firstly, I have to learn more about IoT and Android in order to complete the question 6.

Secondly it’s about how to use ProcessBuilder and Process Java class. I got the description for these from https://docs.oracle.com/javase/7/docs/api/java/lang/ProcessBuilder.html and https://docs.oracle.com/javase/7/docs/api/java/lang/Process.html.

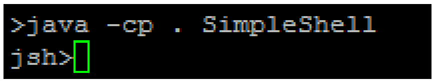
Finally, I need to learn about Linux shell commands from http://www.informit.com/blogs/blog.aspx?uk=The-10-Most-Important-Linux-Commands.

3. [2 points] provide a break down for the activities/milestones for this assignment. Give an estimate of hours spent on each activity. Try to be honest!

Date	Activities	Hours	Outcome
25/09	Revie the lecture handouts	3	Understanding OS/Process/Scheduler and others
26/09	0.1 Practice Exercises 1-5	2	Solutions
27/09	0.1 Practice Exercises 6-7	4	Solutions
27/09	0.2 Programming Exercise: common commands such as ls/cat/ps	2	Source code
28/09	cd history commands	2	Source code

The readme as below which contains a script showing all the test cases performed along with screenshots.

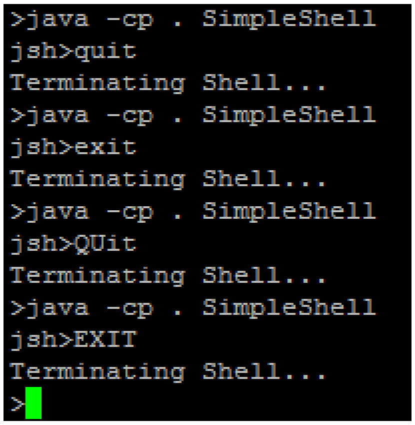
Step 1: This program has been compiled and run successfully.



```
>java -cp . SimpleShell
jsh>
```

Step 2: Exit and quit commands work properly.

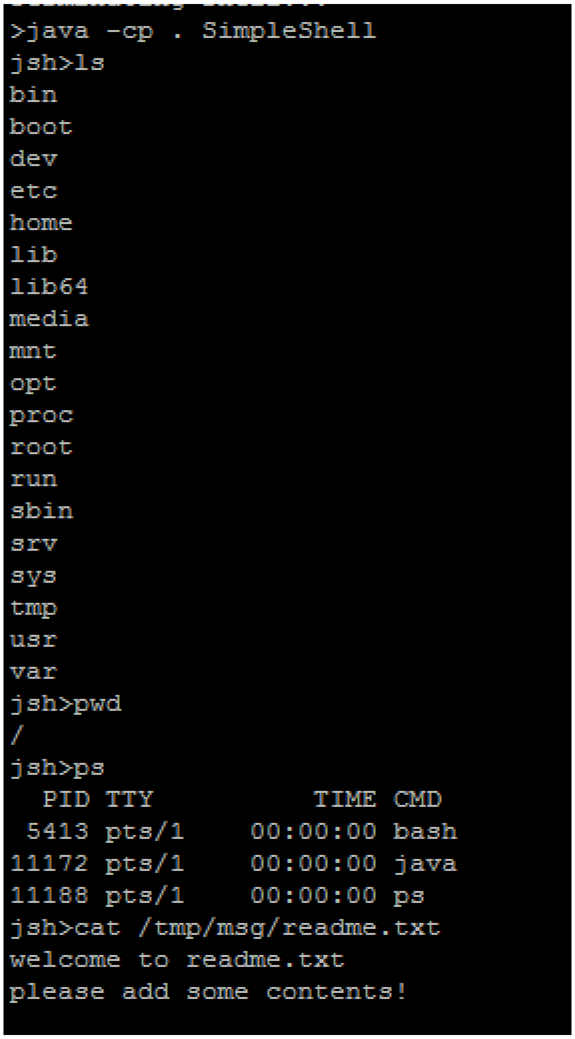
The program is not case sensitive to these commands.



```
>java -cp . SimpleShell
jsh>quit
Terminating Shell...
>java -cp . SimpleShell
jsh>exit
Terminating Shell...
>java -cp . SimpleShell
jsh>QUit
Terminating Shell...
>java -cp . SimpleShell
jsh>EXIT
Terminating Shell...
>
```

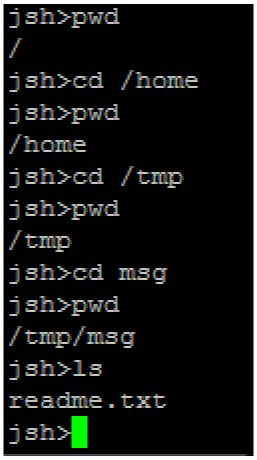
Step 3: The process builder is properly implemented.

Basic commands (ps, ls, cat, …) run successfully.



```
>java -cp . SimpleShell
jsh>ls
bin
boot
dev
etc
home
lib
lib64
media
mnt
opt
proc
root
run
sbin
srv
sys
tmp
usr
var
jsh>pwd
/
jsh>ps
  PID TTY          TIME CMD
 5413 pts/1    00:00:00 bash
11172 pts/1    00:00:00 java
11188 pts/1    00:00:00 ps
jsh>cat /tmp/msg/readme.txt
welcome to readme.txt
please add some contents!
```

Step 4: Changing Directories

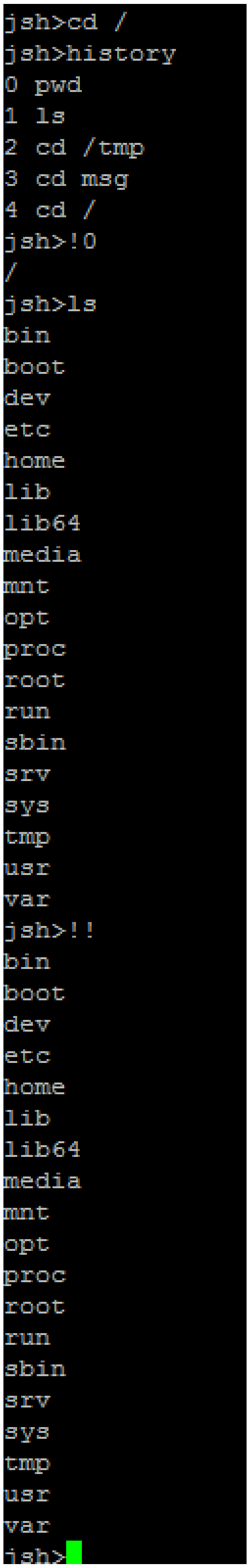


```
jsh>pwd
/
jsh>cd /home
jsh>pwd
/home
jsh>cd /tmp
jsh>pwd
/tmp
jsh>cd msg
jsh>pwd
/tmp/msg
jsh>ls
readme.txt
jsh>
```

Step 5: history command works properly.

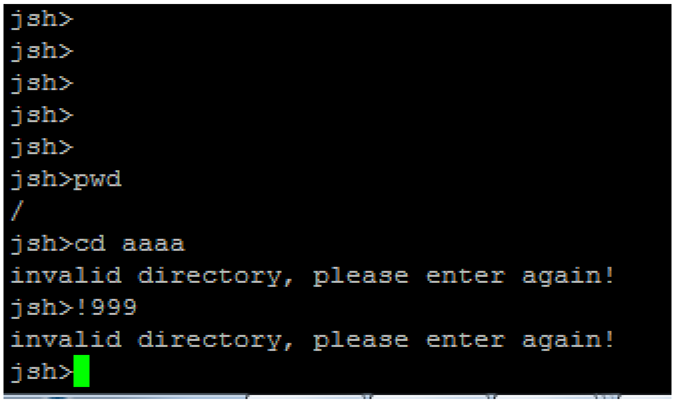
!! command executes the last command in from the history.

!# executes the command # from the history or returns an error message if # is not a valid history command index.



```
jsh>cd /
jsh>history
0 pwd
1 ls
2 cd /tmp
3 cd msg
4 cd /
jsh>!0
/
jsh>ls
bin
boot
dev
etc
home
lib
lib64
media
mnt
opt
proc
root
run
sbin
srv
sys
tmp
usr
var
jsh>!!
bin
boot
dev
etc
home
lib
lib64
media
mnt
opt
proc
root
run
sbin
srv
sys
tmp
usr
var
jsh>
```

Step 6: Code deals with exceptions in an appropriate manner. For example, exceptions such as attempting to change directory to an invalid directory should result in a message to the user and the continuation of the program



```
jsh>
jsh>
jsh>
jsh>
jsh>
jsh>pwd
/
jsh>cd aaaa
invalid directory, please enter again!
jsh>!999
invalid directory, please enter again!
jsh>
```