Linux Programming

Lab Exercise -June

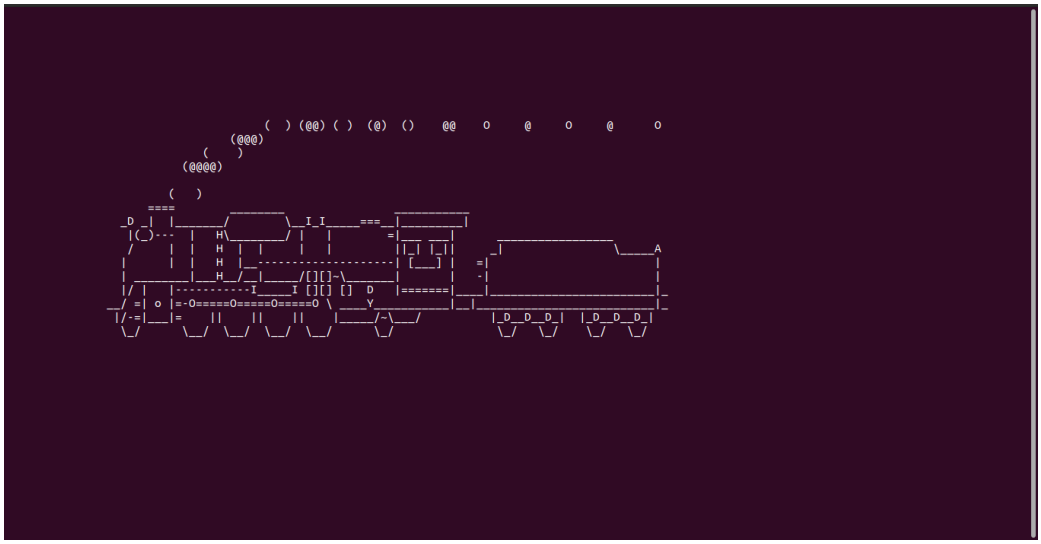
Prof.Harini S

K.SAIKALYAN

17MIS1102

1)sl –*funny train runs in terminal:*

Screenshots:



2) Rev Command in Linux:

Usage: *Rev [text] or[filename]*

rev -h ---Help
rev -V ---Version Number

Screenshots:

```
kalyan@kalyan-VirtualBox:~$ rev
yes
sey
kalyan
naylak
ajay
yaja
```

3) **Factor:** *The factor command in Linux is used to print the prime factors of the given numbers.*

Screenshots:

```
kalyan@kalyan-VirtualBox:~$ factor 20
20: 2 2 5
kalyan@kalyan-VirtualBox:~$
```

4)*yes:* ***y***es command in linux is used to print a continuous output stream of given *STRING*. If *STRING* is not mentioned then it prints ‘y’

Screenshots:

```
kalyan@kalyan-VirtualBox:~$ yes --version
yes (GNU coreutils) 8.30
Copyright (C) 2018 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <https://gnu.org/licenses/gpl.html>
.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

Written by David MacKenzie.
kalyan@kalyan-VirtualBox:~$
```

```
17mis1102_kande
17mis1102_kande
17mis1102_kande
17mis1102_kande
17mis1102_kande
```

Write a bash shell script to monitor the health of your system. Let the details be stored and archived in any folder of your choice.

Instructions:

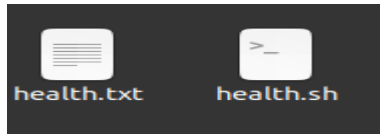
crontab -e --- to install the shell script for automation

Health monitor used in the scenario:

Top---process info
free---memory usage in the system
netstat---network info and socket connected info vmstat—virtual mem and cache info

Screenshots:

files: health.sh and health.txt




```
top -b -d 1 -n 5 >> health.txt
echo " "
echo "Free memory Statistics"
free -m  >> health.txt
echo " "
echo "Network Statistics"
netstat >> health.txt
echo "CPU load and VMstat"
vmstat -s >> health.txt
```

Open health.txt Save

```
top - 00:19:02 up  1:38,  1 user,  load average: 0.17, 0.27, 0.33
Tasks: 248 total,   1 running, 229 sleeping,  18 stopped,   0 zombie
%Cpu(s): 17.6 us,  2.7 sy,  0.0 ni, 78.4 id,  1.4 wa,  0.0 hi,  0.0 si,  0.0 st
MiB Mem :   8173.2 total,   3687.3 free,   2263.5 used,   2222.5 buff/cache
MiB Swap:   2048.0 total,   2048.0 free,      0.0 used.   5485.4 avail Mem

   PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
  2650 kalyan    20   0 3686064 432968 134804 S  52.9   5.2  11:16.32 gnome-she+
 26833 kalyan    20   0 1221480  67328  42624 S  17.6   0.8   0:13.08 nautilus
  2338 kalyan    20   0  790544 317308 106484 S  11.8   3.8   2:52.10 Xorg
 28381 kalyan    20   0   20396   3648   3004 R  11.8   0.0   0:00.02 top
     1 root      20   0  166688  11252   7780 S   0.0   0.1   0:09.45 systemd
     2 root      20   0       0      0      0 S   0.0   0.0   0:00.00 kthreadd
     3 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 rcu_gp
     4 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 rcu_par_gp
     6 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 kworker/0+
     8 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 mm_percpu+
     9 root      20   0       0      0      0 S   0.0   0.0   0:00.15 ksoftirqd+
    10 root      20   0       0      0      0 I   0.0   0.0   0:06.23 rcu_sched
    11 root      rt   0       0      0      0 S   0.0   0.0   0:00.07 migration+
    12 root     -51   0       0      0      0 S   0.0   0.0   0:00.00 idle_inje+
    14 root      20   0       0      0      0 S   0.0   0.0   0:00.00 cpuhp/0
    15 root      20   0       0      0      0 S   0.0   0.0   0:00.00 cpuhp/1
    16 root     -51   0       0      0      0 S   0.0   0.0   0:00.00 idle_inje+
    17 root      rt   0       0      0      0 S   0.0   0.0   0:00.06 migration+
    18 root      20   0       0      0      0 S   0.0   0.0   0:00.11 ksoftirqd+
    20 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 kworker/1+
    21 root      20   0       0      0      0 S   0.0   0.0   0:00.00 cpuhp/2
    22 root     -51   0       0      0      0 S   0.0   0.0   0:00.00 idle_inje+
    23 root      rt   0       0      0      0 S   0.0   0.0   0:00.03 migration+
    24 root      20   0       0      0      0 S   0.0   0.0   0:00.14 ksoftirqd+
    26 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 kworker/2+
    27 root      20   0       0      0      0 S   0.0   0.0   0:00.00 cpuhp/3
    28 root     -51   0       0      0      0 S   0.0   0.0   0:00.00 idle_inje+
    29 root      rt   0       0      0      0 S   0.0   0.0   0:00.06 migration+
    30 root      20   0       0      0      0 S   0.0   0.0   0:01.13 ksoftirqd+
    32 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 kworker/3+
```

Plain Text Tab Width: 8 Ln 1, Col 1 INS

Crontab :

```
  GNU nano 3.2              /tmp/crontab.3rgutj/crontab

# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h  dom mon dow   command
* * * * * /home/kalyan/health.sh

^G Get Help    ^O Write Out   ^W Where Is    ^K Cut Text    ^J Justify
^X Exit        ^R Read File   ^\ Replace     ^U Uncut Text  ^T To Spell
```

Write a C program to implement Simple reader – writer algorithm using shared memory segment with semaphore

A)

```
kalyan@kalyan-VirtualBox:~$ gedit writer.cpp &
[7] 29043
kalyan@kalyan-VirtualBox:~$ g++ writer.cpp
[7]   Done                    gedit writer.cpp
kalyan@kalyan-VirtualBox:~$ ./a.out
Write Data : Hi Kalyan ...Dammnnnn
Data written in memory: Hi Kalyan ...Dammnn
kalyan@kalyan-VirtualBox:~$ g++ reader.cpp
kalyan@kalyan-VirtualBox:~$ ./a.out
Data read from memory: Hi Kalyan ...Dammnn
kalyan@kalyan-VirtualBox:~$
```