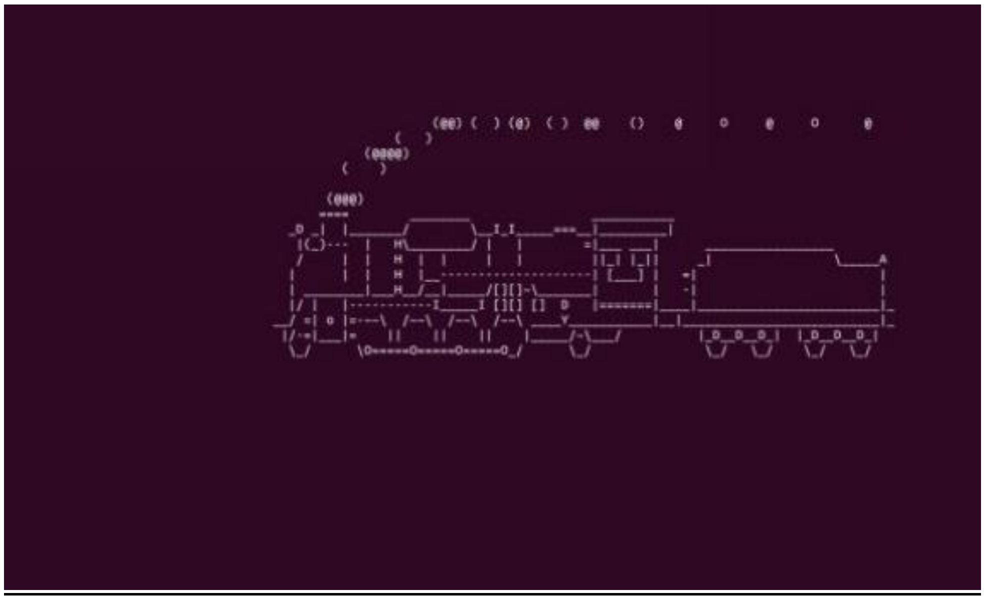
LINUX LAB

NAME: ANDIAPPAN V
REG NO:17MIS1143

1: how to use SL command?

Code:
From root user install: sudo apt install sl
And run the command:sl



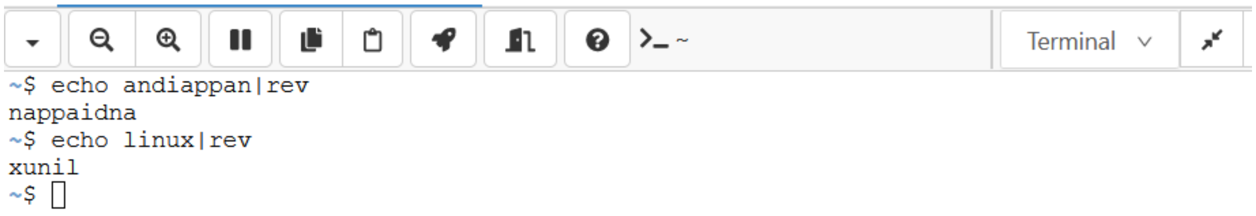
```
                      (@@) (  ) (@)  ( )  @@    ()    @     O     @     O      @
                 (   )
             (@@@@)
          (    )

        (@@@)
      ====        ________                ___________
  _D _|  |_______/        \__I_I_____===__|_________|
   |(_)---  |   H\________/ |   |        =|___ ___|      _________________
   /     |  |   H  |  |     |   |         ||_| |_||     _|                \_____A
  |      |  |   H  |__--------------------| [___] |   =|                        |
  | ________|___H__/__|_____/[][]~\_______|       |   -|                        |
  |/ |   |-----------I_____I [][] []  D   |=======|____|________________________|_
__/ =| o |=-~~\  /~~\  /~~\  /~~\ ____Y___________|__|__________________________|_
 |/-=|___|=    ||    ||    ||    |_____/~\___/          |_D__D__D_|  |_D__D__D_|
  \_/      \O=====O=====O=====O_/      \_/               \_/   \_/    \_/   \_/
```

2: reversing the string using re command

Code:

1)$ echo andiappan | rev
output: nappaidna

2) $ echo linux | rev
output:xunil

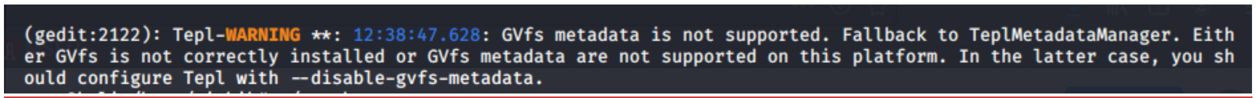


```
~$ echo andiappan|rev
nappaidna
~$ echo linux|rev
xunil
~$
```

3:vmstat :to check the virtual machine state and condition of cpu

code:

```
vmstat 1200 > vmstat1.data
filename= "/home/srihari/vmstat1.data"
tail -f $filename |
while read $line do
if [ (cat vmstat1.data | grep "swap")>0 ]
then
echo "some rogue process has consumed massive amounts of memory"> swap.txt
fi
if [ (cat vmstat1.data | grep "r")>1 ]
then
echo "some process are waiting to execute"> runqueue.txt
fi
if [ (cat vmstat1.data | grep "cpu")>1000 ]

then
echo "cpu usage is more"> cpu.txt
fi
End
```



```
(gedit:2122): Tepl-WARNING **: 12:38:47.628: GVfs metadata is not supported. Fallback to TeplMetadataManager. Either GVfs is not correctly installed or GVfs metadata are not supported on this platform. In the latter case, you should configure Tepl with --disable-gvfs-metadata.
```

Therefore my system is in good health ,or else there should be an alert message,popping up.

EXPLANATION:
the vmstat 1200 – monitors every 24 hours and puts the data into the vmstat1.data
grep “swap”- the swap should always be zero if its not then some process has consumed massive memory.
That will be monitored in this line
grep “r”- the running queue is constantly above process 1 it indicates the system is slow and some process

is waiting to be executed. That will be monitored here.
Grep “cpu”- it indicates the cpu usage of the system. If the cpu usage is more it will be monitored and will
alert in this line.

4)reader writer process using pipe

#c code

```
#include<stdio.h>
#include<unistd.h>

int main() {
   int pipefds1[2], pipefds2[2];
   int returnstatus1, returnstatus2;
   int pid;
   char pipe1writemessage[20] = "Hi";
   char pipe2writemessage[20] = "Hello";
   char readmessage[20];
   returnstatus1 = pipe(pipefds1);

   if (returnstatus1 == -1) {
      printf("Unable to create pipe 1 \n");
      return 1;
   }
   returnstatus2 = pipe(pipefds2);

   if (returnstatus2 == -1) {
      printf("Unable to create pipe 2 \n");
      return 1;
   }
   pid = fork();

   if (pid != 0) // Parent process {
      close(pipefds1[0]); // Close the unwanted pipe1 read side
      close(pipefds2[1]); // Close the unwanted pipe2 write side
      printf("In Parent: Writing to pipe 1 – Message is %s\n",
pipe1writemessage);
      write(pipefds1[1], pipe1writemessage, sizeof(pipe1writemessage));
      read(pipefds2[0], readmessage, sizeof(readmessage));
      printf("In Parent: Reading from pipe 2 – Message is %s\n",
readmessage);
   } else { //child process
      close(pipefds1[1]); // Close the unwanted pipe1 write side
      close(pipefds2[0]); // Close the unwanted pipe2 read side
      read(pipefds1[0], readmessage, sizeof(readmessage));
      printf("In Child: Reading from pipe 1 – Message is %s\n", readmessage);
      printf("In Child: Writing to pipe 2 – Message is %s\n",
pipe2writemessage);
      write(pipefds2[1], pipe2writemessage, sizeof(pipe2writemessage));
   }
   return 0;
}
```

Explanation

Step 1 − Create pipe1 for the parent process to write and the child process to read.

Step 2 − Create pipe2 for the child process to write and the parent process to read.

Step 3 − Close the unwanted ends of the pipe from the parent and child side.

Step 4 − Parent process to write a message and child process to read and display on the screen.

Step 5 − Child process to write a message and parent process to read and display on the screen.

```
In Parent: Writing to pipe 1 – Message is Hi
In Child: Reading from pipe 1 – Message is Hi
In Child: Writing to pipe 2 – Message is Hello
In Parent: Reading from pipe 2 – Message is Hello
```