Adam Cohen

CS 340

Professor Fluture

Homework #4

**PART 1**

**Execution of “date&; who; whoami; uname; echo Hello, World&” output**:

[1] 19003

Tue Mar 25 12:45:06 EDT 2014

yang pts/0 2014-03-23 03:00 (pool-100-2-73-208.nycmny.fios.verizon.net)

chan0371 pts/1 2014-03-25 11:24 (pool-98-116-90-126.nycmny.fios.verizon.net)

sale1252 pts/2 2014-03-25 12:41 (149.4.104.63)

shmo3478 pts/3 2014-03-25 12:16 (pool-108-27-61-7.nycmny.fios.verizon.net)

waan9368 pts/7 2014-03-25 12:01 (149.4.223.113)

desh1235 pts/9 2014-03-19 08:56 (149.4.223.91)

ghmu7764 pts/10 2014-03-25 12:02 (149.4.223.104)

rujo4209 pts/11 2014-03-25 12:23 (bsc3.qc.cuny.edu)

gaxi1808 pts/14 2014-03-25 12:03 (149.4.223.125)

xuxi1161 pts/15 2014-03-25 10:09 (cpe-67-243-132-202.nyc.res.rr.com)

wuyo0849 pts/16 2014-03-25 12:03 (149.4.223.112)

yust6570 pts/17 2014-03-25 12:09 (149.4.243.59)

fuvi1343 pts/18 2014-03-25 12:09 (149.4.223.115)

rasu4486 pts/19 2014-03-25 12:29 (149.4.223.101)

nahi2446 pts/20 2014-03-25 12:10 (149.4.223.28)

avmi6821 pts/21 2014-03-25 12:27 (bsc3.qc.cuny.edu)

hemo7628 pts/22 2014-03-25 12:11 (149.4.223.24)

ehra4809 pts/23 2014-03-23 21:17 (pool-96-224-251-48.nycmny.fios.verizon.net)

myje1040 pts/24 2014-03-25 12:14 (bsc3.qc.cuny.edu)

rued3426 pts/26 2014-03-25 12:17 (149.4.223.118)

pisa6440 pts/27 2014-03-25 12:18 (149.4.223.110)

brel0490 pts/29 2014-03-25 12:27 (bsc3.qc.cuny.edu)

yuqi1302 pts/30 2014-03-25 12:34 (149.4.223.126)

mido8334 pts/31 2014-03-25 12:30 (149.4.223.116)

quji5451 pts/32 2014-03-25 12:30 (149.4.223.120)

maaa3649 pts/34 2014-03-25 12:37 (149.4.110.21)

coad4837 pts/35 2014-03-25 12:37 (149.4.223.72)

[1]+ Done date

coad4837

Linux

[1] 19007

-sh-4.1$ Hello, World

**Nice man page**:

NAME

nice - run a program with modified scheduling priority

SYNOPSIS

nice [OPTION] [COMMAND [ARG]...]

DESCRIPTION

Run COMMAND with an adjusted niceness, which affects process schedul-

ing. With no COMMAND, print the current niceness. Nicenesses range

from -20 (most favorable scheduling) to 19 (least favorable).

-n, --adjustment=N

add integer N to the niceness (**default 10**)🡨default value for adj.

**PART 2**

**Windows**- Because MS-DOS was a non-multitasking OS it didn’t require a scheduler. In Windows 3.1x the scheduling algorithm used was non-preemptive so processes were not interrupted. This method is also known as cooperative multitasking. A basic preemptive scheduling algorithm started to be used in Windows 95. This algorithm let 16 bit applications run without preemption. Windows-NT used a multilevel feedback queue where there are 32 levels of priority. The Task Manager allowed the user to change between 5 of these priority levels. The kernel has the ability to modify a thread priority level based on I/O and CPU usage. The algorithm was modified in Windows Vista to utilize modern processors’ cycle counter register in order to keep track of the number of CPU cycles a thread executed vs the prior metric (interval-timer interrupt routine). In addition a priority scheduler is used in Vista for I/O queue so I/O programs don’t interfere with other operations in the foreground.

**Mac**- The older version of Mac, OS 9, used cooperative scheduling algorithm where one process controls multiple cooperative threads, and also provides preemptive scheduling for MP tasks. All process manager processes are scheduled cooperatively using a round robin algorithm. Each process on OS 9 has its own copy of the Thread Manager that schedules the threads belonging to a process cooperatively. The newer version of Mac OS is OS X which uses a multilevel feedback queue with four thread priority bands which are labeled: normal, system high priority, kernel mode only and real-time. The threads in OS X are scheduled preemptively but can also be cooperatively scheduled.

**Linux**- Linux uses two process scheduling algorithms: Time sharing algorithm and real time algorithm. A process’s scheduling class defines which scheduling algorithm to apply. For time sharing, Linux uses a prioritize credit based algorithm. The credit rule Credit = credit/2 + priority factors in both process history and its priority. The credit system also auto-prioritizes interactive or I/O bound processes. For real time scheduling classes, it uses FIFO and robin round system. The scheduler runs the process with the highest priority. If the priority is equal, it runs the process waiting the longest. FIFO process continues to run either they exit or block. The round robin process is pre-emptive after a while and places at the end of the scheduling queue.