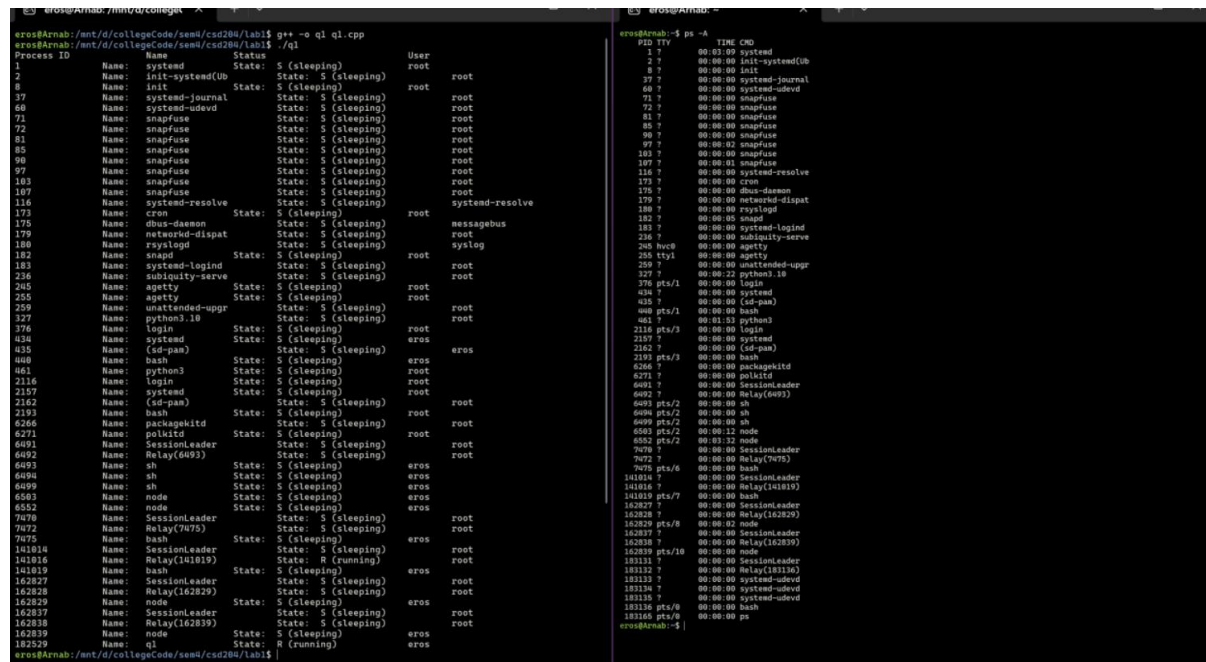


Q1.)

We observe that the written program gives us an additional output in the form of if the process is running or sleeping, and who is the user of the process (in this case, either the root or Eros). However, the “ps -A” command returns the time of the process as well.

Additionally, the PID of the processes may have changed between the command and the compilation of the process.

The written program gives us a more comprehensive overview of the current programs.



```
eros@Arbab:/mnt/d/collegeCode/sem4/csd204/lab1$ g++ -o q1 q1.cpp
eros@Arbab:/mnt/d/collegeCode/sem4/csd204/lab1$ ./q1
Process ID      Name      State      User
1               systemd  S (sleeping) root
2               init-systemd(Up)  S (sleeping) root
37              systemd-journal  S (sleeping) root
66              systemd-udevdev  S (sleeping) root
71              snapfuse  S (sleeping) root
72              snapfuse  S (sleeping) root
81              snapfuse  S (sleeping) root
85              snapfuse  S (sleeping) root
98              snapfuse  S (sleeping) root
97              snapfuse  S (sleeping) root
103             snapfuse  S (sleeping) root
107             snapfuse  S (sleeping) root
116             snapfuse  S (sleeping) root
116             systemd-resolve  S (sleeping) systemd-resolve
173             cron       S (sleeping) root
175             dbus-daemon  S (sleeping) messagebus
179             networkd-dispat  S (sleeping) root
180             rsyslogd   S (sleeping) syslog
182             snapd      S (sleeping) root
183             systemd-logind  S (sleeping) root
236             subiquity-serve  S (sleeping) root
245             agetty     S (sleeping) root
285             agetty     S (sleeping) root
259             unattended-upgr  S (sleeping) root
327             python3.10  S (sleeping) root
376             login      S (sleeping) root
434             systemd   S (sleeping) root
435             (sd-pan)   S (sleeping) eros
448             bash       S (sleeping) eros
461             python3    S (sleeping) root
2116            login      S (sleeping) root
2157            systemd   S (sleeping) root
2162            (sd-pan)   S (sleeping) root
2193            pts/1     S (sleeping) eros
2193            pts/3     S (sleeping) eros
4266            packagekitd  S (sleeping) root
6091            polkitd   S (sleeping) root
6091            SessionLeader  S (sleeping) root
6091            Relay(6093)  S (sleeping) root
6493            sh         S (sleeping) eros
6494            sh         S (sleeping) eros
6499            sh         S (sleeping) eros
6503            node       S (sleeping) eros
6552            SessionLeader  S (sleeping) root
7470            Relay(7475)  S (sleeping) root
7472            bash       S (sleeping) eros
7475            SessionLeader  S (sleeping) eros
141014           hash       S (sleeping) root
141016           Relay(141019)  S (sleeping) eros
141019           SessionLeader  S (sleeping) root
162827           node       S (sleeping) root
162828           Relay(162829)  S (sleeping) eros
162829           SessionLeader  S (sleeping) root
162837           node       S (sleeping) eros
162838           Relay(162839)  S (sleeping) root
162839           SessionLeader  S (sleeping) root
182529           q1         R (running) eros

eros@Arbab:/mnt/d/collegeCode/sem4/csd204/lab1$ ps -A
PID TTY      TIME CMD
1 ?        00:01:09 systemd
2 ?        00:00:00 init-systemd(lib
8 ?        00:00:00 init
37 ?       00:00:00 systemd-journal
66 ?       00:00:00 systemd-udev
71 ?       00:00:00 snapfuse
72 ?       00:00:00 snapfuse
81 ?       00:00:00 snapfuse
85 ?       00:00:00 snapfuse
98 ?       00:00:00 snapfuse
97 ?       00:00:00 snapfuse
103 ?      00:00:00 snapfuse
107 ?      00:00:01 snapfuse
116 ?      00:00:00 snapfuse
116 ?      00:00:00 systemd-resolve
173 ?      00:00:00 cron
175 ?      00:00:00 dbus-daemon
179 ?      00:00:00 networkd-dispat
180 ?      00:00:00 rsyslogd
182 ?      00:00:00 snapd
183 ?      00:00:00 systemd-logind
236 ?      00:00:00 subiquity-serve
245 ?      00:00:00 agetty
285 ?      00:00:00 agetty
259 ?      00:00:00 unattended-upgr
327 ?      00:00:22 python3.10
376 ?      00:00:00 login
434 ?      00:00:00 systemd
435 ?      00:00:00 (sd-pan)
448 ?      00:00:00 bash
461 ?      00:01:53 python3
2116 ?     00:00:00 login
2157 ?     00:00:00 systemd
2162 ?     00:00:00 (sd-pan)
2193 ?     00:00:00 pts/1
2193 ?     00:00:00 pts/3
4266 ?     00:00:00 packagekitd
6091 ?     00:00:00 polkitd
6091 ?     00:00:00 SessionLeader
6091 ?     00:00:00 Relay(ev3)
6493 ?     00:00:00 sh
6494 ?     00:00:00 sh
6499 ?     00:00:00 sh
6503 ?     00:00:12 node
6552 ?     00:00:00 SessionLeader
7470 ?     00:00:00 Relay(7475)
7472 ?     00:00:00 bash
7475 ?     00:00:00 pts/6
141014 ?    00:00:00 SessionLeader
141016 ?    00:00:00 Relay(141019)
141019 ?    00:00:00 SessionLeader
162827 ?    00:00:00 node
162828 ?    00:00:00 Relay(162829)
162829 ?    00:00:00 SessionLeader
162837 ?    00:00:00 node
162838 ?    00:00:00 Relay(162839)
162839 ?    00:00:00 SessionLeader
182529 ?    00:00:00 q1
183115 ?    00:00:00 systemd-udev
183116 ?    00:00:00 bash
183116 ?    00:00:00 pts/8
eros@Arbab:~$
```

Q2.)

eros@Arbab:/mnt/d/collegeCode/sem4/csd204/lab1/Assgn1-ArnabMandal\$./q2

Config of the system:

OS: Linux

Release: 5.15.133.1-microsoft-standard-WSL2

Version: #1 SMP Thu Oct 5 21:02:42 UTC 2023

Machine: x86_64

eros@Arbab:/mnt/d/collegeCode/sem4/csd204/lab1/Assgn1-ArnabMandal\$

Q3.)

a) The processor or the CPU, is the overall chip that contains cores and threads, to execute tasks using its cores. Example:

On-line CPU(s) list: 0-19

Vendor ID: GenuineIntel

Model name: 13th Gen Intel(R) Core(TM) i9-13900H

CPU family: 6

Model: 186

Thread(s) per core: 2

Core(s) per socket: 10

The cores are present in the CPU, with each core being a processing unit within the processor that handles individual tasks independently and enables parallel task execution.

b) 10 core(s) per socket * 1 socket = 10 cores

c) 0-19 processors, a total of 20 CPU(s)

d) 2995.211 Mhz

e) Architecture: x86_64

CPU op-mode(s): 32-bit, 64-bit

Address sizes: 46 bits physical, 48 bits virtual

Byte Order: Little Endian

f) MemTotal: 7944564 kB

g) MemFree: 6390432 kB

h) Context switches: 1496748

Forks: 13641

Q4.)

a) 1602 is the PID

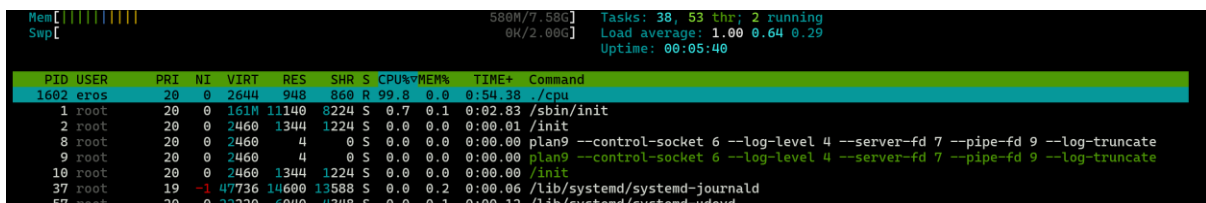
b) PID: 1602 CMD

%CPU: 102

%MEM: 0.0

CMD: ./cpu

Or



The screenshot shows a terminal window with system status information at the top: Mem[|||||] 588M/7.58G, Swap[] 0K/2.00G, Tasks: 38, 63 thr; 2 running, Load average: 1.00 0.64 0.29, Uptime: 00:05:40. Below this is the output of the 'top' command, showing a list of processes. The process with PID 1602, user eros, is highlighted in blue, showing it is running with 102% CPU usage and 0.0% memory usage. The command is './cpu'.

PID	USER	PRI	NI	VIRT	RES	SHR	S	CPU%	MEM%	TIME+	Command
1602	eros	20	0	2644	948	860	R	99.8	0.0	0:54.38	./cpu
1	root	20	0	161M	11140	8224	S	0.7	0.1	0:02.83	/sbin/init
2	root	20	0	2460	1344	1224	S	0.0	0.0	0:00.01	/init
8	root	20	0	2460	4	0	S	0.0	0.0	0:00.00	plan9 --control-socket 6 --log-level 4 --server-fd 7 --pipe-fd 9 --log-truncate
9	root	20	0	2460	4	0	S	0.0	0.0	0:00.00	plan9 --control-socket 6 --log-level 4 --server-fd 7 --pipe-fd 9 --log-truncate
10	root	20	0	2460	1344	1224	S	0.0	0.0	0:00.00	/init
37	root	19	-1	47736	14600	13588	S	0.0	0.2	0:00.06	/lib/systemd/systemd-journald
57	root	20	0	22220	6040	4348	S	0.0	0.1	0:00.12	/lib/systemd/systemd-jdevd

c)

```
eros@Arnab:~$ ps -o pid,state,cmd -C cpu
PID S CMD
1602 R ./cpu
eros@Arnab:~$
```

It is in the running state

```
top - 13:42:33 up 6 min, 1 user, load average: 1.00, 0.72, 0.34
Tasks: 1 total, 1 running, 0 sleeping, 0 stopped, 0 zombie
%Cpu(s): 5.0 us, 0.0 sy, 0.0 ni, 95.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
MiB Mem : 7758.4 total, 6740.4 free, 587.2 used, 430.8 buff/cache
MiB Swap: 2048.0 total, 2048.0 free, 0.0 used, 6941.6 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
1602	eros	20	0	2644	948	860	R	100.0	0.0	2:04.11	cpu

Q5.)

a) 16558

b) 15522

Parent: 15520

ParentL15519

Parent: 2

Parent: 1

Parent: 0

c) If the file descriptor is 0, its either disconnected or redirected to /dev/null to be in the background

If it is 1, it points to /tmp/tmp.txt where output will be captured

If it is 2, it stays connected to terminal to be debugged

d) eros@Arnab:/mnt/d/collegeCode/sem4/csd204/lab1/Assgn1-ArnabMandal\$ ls -l /proc/16433/fd

total 0

lrwx----- 1 eros eros 64 Jan 26 09:44 0 -> /dev/pts/0

lrwx----- 1 eros eros 64 Jan 26 09:44 1 -> /dev/pts/0

lrwx----- 1 eros eros 64 Jan 26 09:44 2 -> /dev/pts/0

eros@Arnab:/mnt/d/collegeCode/sem4/csd204/lab1/Assgn1-ArnabMandal\$

Basically when the shell creates a pipe, the shell redirects the standard output of the first process aka cpu-print into the pipe, and redirects the second process aka grep hello to read from the pipe, enabling processes to communicate with each other.

e) Cd and history are build in commands

Ls and ps are external executables

Q6.)

PID VSZ RSS CMD

1534 6560 4732 ./mem1

PID VSZ RSS CMD

1794 6560 4884 ./mem2

Expected was that mem1 would have higher RSS due to array being actively accessed. The

difference is due to lazy allocation, i.e. the system allots ram before it even used.

Q7.) One of the programs randomly opens files among the ones created, while the other repeatedly opens foo0 In a loop. The disk utilization is highest during the first run, but then it reduces because it is cached into memory. Disk1 has a higher disk throughput while disk has random access causing frequent disk seeks.