

OUTPUT OF PROCESS WATCHER:

The script program that I wrote will show 4 options to choose from (namely user, percentage cpu usage, memory usage and check for a bad process). The screen shot below shows the initial output of the program:

```
AvikRnB:~ avikrb$ ./processwatcher

Enter your Choice of displaying Processes:
1.   User account
2.   Percentage of CPU
3.   Memory
4.   Look for bad processes
█
```

Figure 1 Initial display menu

USER ACCOUNT:

Choosing option “1” further asks the user to enter the USER NAME. In my case, I typed in “avikrb” as the username (which is my account, I could have typed in “root” to display processes from root).

```
AvikRnB:~ avikrb$ ./processwatcher

Enter your Choice of displaying Processes:
1.   User account
2.   Percentage of CPU
3.   Memory
4.   Look for bad processes
1
What is your user ID:
avikrb
```

Figure 2 Option 1

After entering the specified username the program shows only the list of processes run from user “avikrb”.

```
AvikRnB:~ avikrb$ ./processwatcher

Enter your Choice of displaying Processes:
1. User account
2. Percentage of CPU
3. Memory
4. Look for bad processes
1
What is your user ID:
avikrb
USER          PID  %CPU %MEM    RSS   VSZ
avikrb         40   0.0  0.2  10048 2538844
avikrb        326   0.0  0.0    920 2453148
avikrb        329   0.0  0.1   5560 2485748
avikrb        330   0.0  0.1   3992 2472836
avikrb        333   0.0  0.0   1968 2449672
avikrb        339   0.0  0.4  17212 2567324
avikrb        341   0.0  0.3  12864 2555516
avikrb        342   0.5  1.0  40872 2670940
avikrb        343   0.0  0.1   5288 2472284
avikrb        346   0.4  0.1   2816 2473932
avikrb        347   0.0  0.0    296 2446088
avikrb        348   0.0  0.1   4272 2474744
avikrb        349   0.0  0.1   3308 2479716
avikrb        350   0.0  0.1   3256 2456060
avikrb        352   0.0  0.1   3464 2506604
avikrb        355   0.0  0.0   2096 2469476
avikrb        356   0.0  0.1   2712 2469672
avikrb        357   0.0  0.1   5308 2484940
avikrb        358   0.0  0.1   4728 2485852
avikrb        361   0.0  0.3  11824 2523288
avikrb        363   0.0  0.1   2228 2482512
avikrb        365   0.0  0.1   2832 2472332
avikrb        366   0.0  0.1   4968 2474976
avikrb        369   0.0  0.0   1104 2447600
avikrb        370   0.0  0.2   8412 2479728
avikrb        373   0.1  0.0   1248  656656
avikrb        374   0.0  0.0   1852  662956
avikrb        375   0.0  0.1   4140  698100
avikrb        376   0.0  0.2   7256 2520320
avikrb        378   0.0  0.1   3484 2484644
avikrb        383   0.0  0.1   4628 2499076
avikrb        388   0.0  0.1   4332 2494472
avikrb        391   0.0  0.1   5276 2476940
avikrb        392   0.0  0.1   3312 2471164
avikrb        393   0.0  0.1   3832  673436
avikrb        394   0.0  0.2  10420 2468848
avikrb        401   0.0  0.0   1776 2471248
avikrb        402   0.0  0.1   5992 2516816
avikrb        403   0.0  0.0   1008 2446316
avikrb        407   0.0  0.1   5812 2496332
avikrb        413   0.0  0.2   6748 2448544
avikrb        414   0.0  0.2   9604 2478976
avikrb        423   0.0  2.3  96356 3865132
```

Figure 3 Result for Option 1

(Further output of the previous command is omitted)

PERCENTAGE OF CPU USAGE

The second option, option “2” directly displays ONLY the processes that are using significant percentage of the CPU along with the User and Process ID (PID). A sample out is shown below:

```
AvikRnB:~ avikrb$ ./processwatcher
Enter your Choice of displaying Processes:
1. User account
2. Percentage of CPU
3. Memory
4. Look for bad processes
2
USER          PID  %CPU
_windowserver 138   6.5
root          284   0.2
root          397   0.1
avikrb        425   0.1
avikrb        614   0.3
avikrb        619   0.2
avikrb        864   1.8
avikrb       1229   0.5
avikrb       1524   1.3
avikrb       1527   3.0
avikrb       1561   8.4
avikrb       1563   1.6
avikrb       1564   2.0
avikrb       1677   1.7
AvikRnB:~ avikrb$
```

Figure 4 Result for Option 2

MEMORY USAGE:

Option “3” in the display menu gives the list of processes using significant memory. The fields that are shown are User, Process ID (PID), Percentage of Memory used (%mem) and RSS. A sample out is shown below:

```
AvikRnB:~ avikrb$ ./processwatcher

Enter your Choice of displaying Processes:
1. User account
2. Percentage of CPU
3. Memory
4. Look for bad processes
3
USER          PID %MEM  RSS
root           1  0.1   3808
root          11  0.2   6476
root          12  0.1   3100
root          13  0.1   2628
root          18  0.1   2444
root          19  0.1   3328
root          22  0.1   2500
root          29  0.1   3004
root          33  0.1   6132
root          36  0.4  18576
_mdnsresponder 37  0.1   3260
avikrb         40  0.2   9956
_locationd     42  0.1   3372
root           46  0.1   2128
root           47  0.1   3896
root           52  0.1   4136
root           55  0.1   2800
root           61  0.1   3296
root           62  0.1   5868
root           64  0.1   2972
root           65  0.8  32648
root           73  0.1   4572
root           77  0.1   4736
_windowserver  138 1.8  76192
root           171 0.6  26800
root           176 0.1   3664
root           272 0.1   3392
root           284 0.2   7816
_softwareupdate 287 2.7 112840
root           290 0.1   4156
_coreaudiod    291 0.1   5488
_coreaudiod    293 0.1   2636
avikrb         329 0.1   5560
avikrb         330 0.1   4116
avikrb         339 0.4  17168
avikrb         341 0.3  12824
avikrb         342 1.1  45716
avikrb         343 0.1   5288
avikrb         346 0.1   2968
```

Figure 5 Result for Option 3

(Further output is omitted)

BAD PROCESSES:

Option “4” provides an option for the users to check if a certain process is a “bad process”. These bad processes can either be spam or viruses. For the program Process Watcher, I input random names in a file called “badprocess.txt”. This file is

later opened within the program to check if it contains the process name of the process user wants to check. Screenshot below shows the file “badprocess.txt”.

```
AvikRnB:~ avikrb$ cat badprocess.txt
x
bad
dirty
virus
trojan
srojan
loopvirus
casino
bigmoney

AvikRnB:~ avikrb$
```

Figure 6 badprocess.txt file

When the user enters option “4” in the initial display menu, he/she is asked to enter a process name to check whether it is a bad process or not.

```
AvikRnB:~ avikrb$ ./processwatcher

Enter your Choice of displaying Processes:
1. User account
2. Percentage of CPU
3. Memory
4. Look for bad processes
4
Enter Process name to check if it is a bad Process:
```

Figure 7 Option 4

For illustration, I am entering “x” as the process name. “X” refers to XQuartz program that runs on Apple computers. Entering “x” on terminal opens the XQuartz automatically. Here, I am actually running XQuartz behind the terminal, and I have listed “x” as a bad process name in “badprocess.txt” file. So when I am prompted to enter a process name to check whether it is a bad process or not I will be entering “x”. This should tell me that “x” is a bad process and it should further provide me with an option to kill the process. Screenshot below explains this.

```
AvikRnB:~ avikrb$ ./processwatcher

Enter your Choice of displaying Processes:
1. User account
2. Percentage of CPU
3. Memory
4. Look for bad processes
4
Enter Process name to check if it is a bad Process:
x
x is a bad process
Do you want to stop x [Y/N]??
```

Figure 8 Kill a bad process

If I now choose “y” as my choice, then XQuartz should be terminated from my system.

```
AvikRnB:~ avikrb$ ./processwatcher

Enter your Choice of displaying Processes:
1. User account
2. Percentage of CPU
3. Memory
4. Look for bad processes
4
Enter Process name to check if it is a bad Process:
x
x is a bad process
Do you want to stop x [Y/N]??
y
[3]+ Killed: 9
AvikRnB:~ avikrb$
```

Figure 9 Bad process killed

And that is what exactly happens. XQuartz is terminated. The program contains a command to kill the bad process if the user wants to.

However if the user does not want to kill the bad process then choosing “n” keeps the process running.

```
AvikRnB:~ avikrb$ ./processwatcher
Enter your Choice of displaying Processes:
1. User account
2. Percentage of CPU
3. Memory
4. Look for bad processes
4
Enter Process name to check if it is a bad Process: x
x
x is a bad process
Do you want to stop x [Y/N]??
n
x Process is still running!!!!
AvikRnB:~ avikrb$
```

Figure 10 Not terminating a bad process