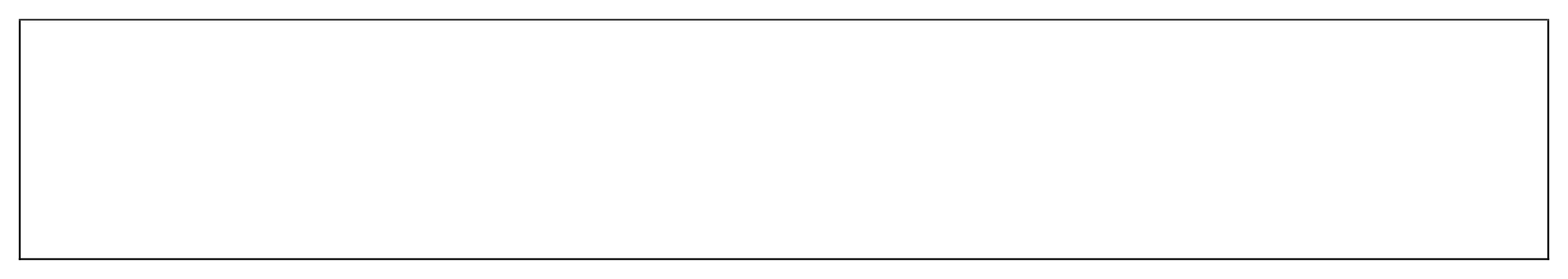
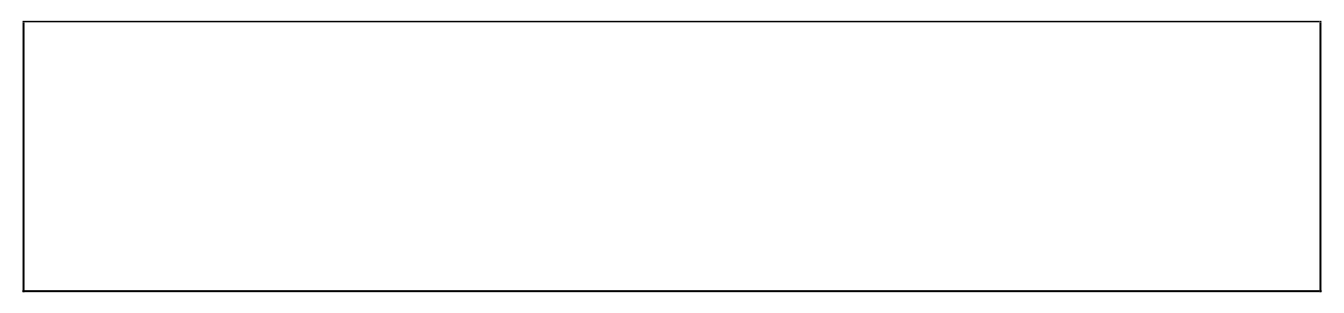
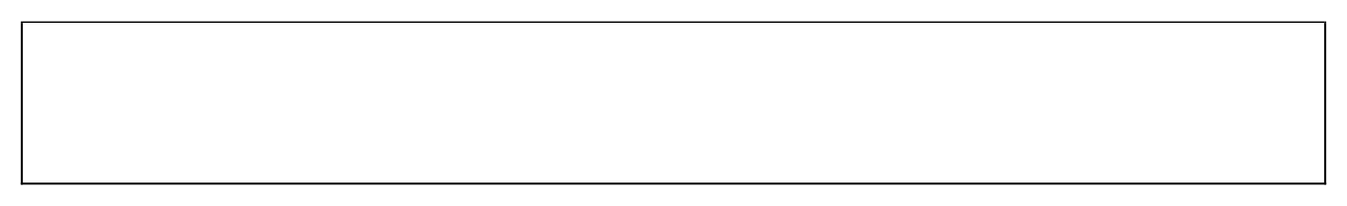
SCR2043 OPERATING SYSTEMS 

|  |  |  |  |
| --- | --- | --- | --- |
| Name | |  | | --- | | Aland Fryad | | Marks |
| Student ID : qiu23-0457 | |
| Section | : Software Engineering |

This lab assessment is designed to test your understanding and skills on some basic concepts

and tools related to process monitoring and management in operating system. Please follow the instructions carefully and submit your answers in this word document and rename the file as os-lab-assessment02-studentname-matricno.docx.

Essential Steps Before Starting Lab Assessment 2:

1. Download necessary source codes:

Use the wget command to retrieve the following source code files to your Linux (or WSL or MacOS) environment:

|  |  |
| --- | --- |
| |  | | --- | | wget -O mainprocess.c https://rebrand.ly/mainprocess\_c wget -O  subprocess1.c https://rebrand.ly/subprocess1\_c wget -O  subprocess2.c https://rebrand.ly/subprocess2\_c | |

2. Compile the source files:

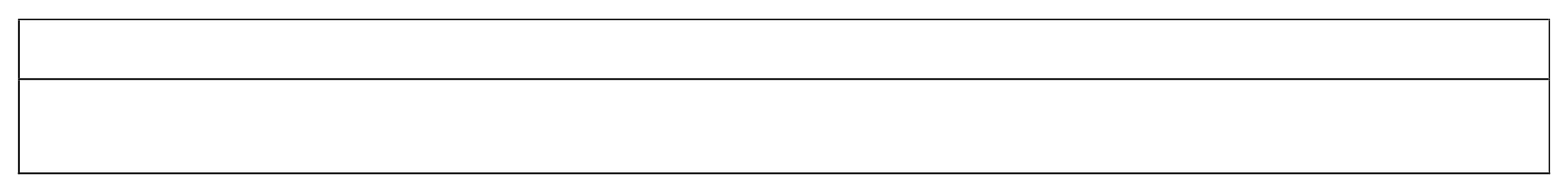
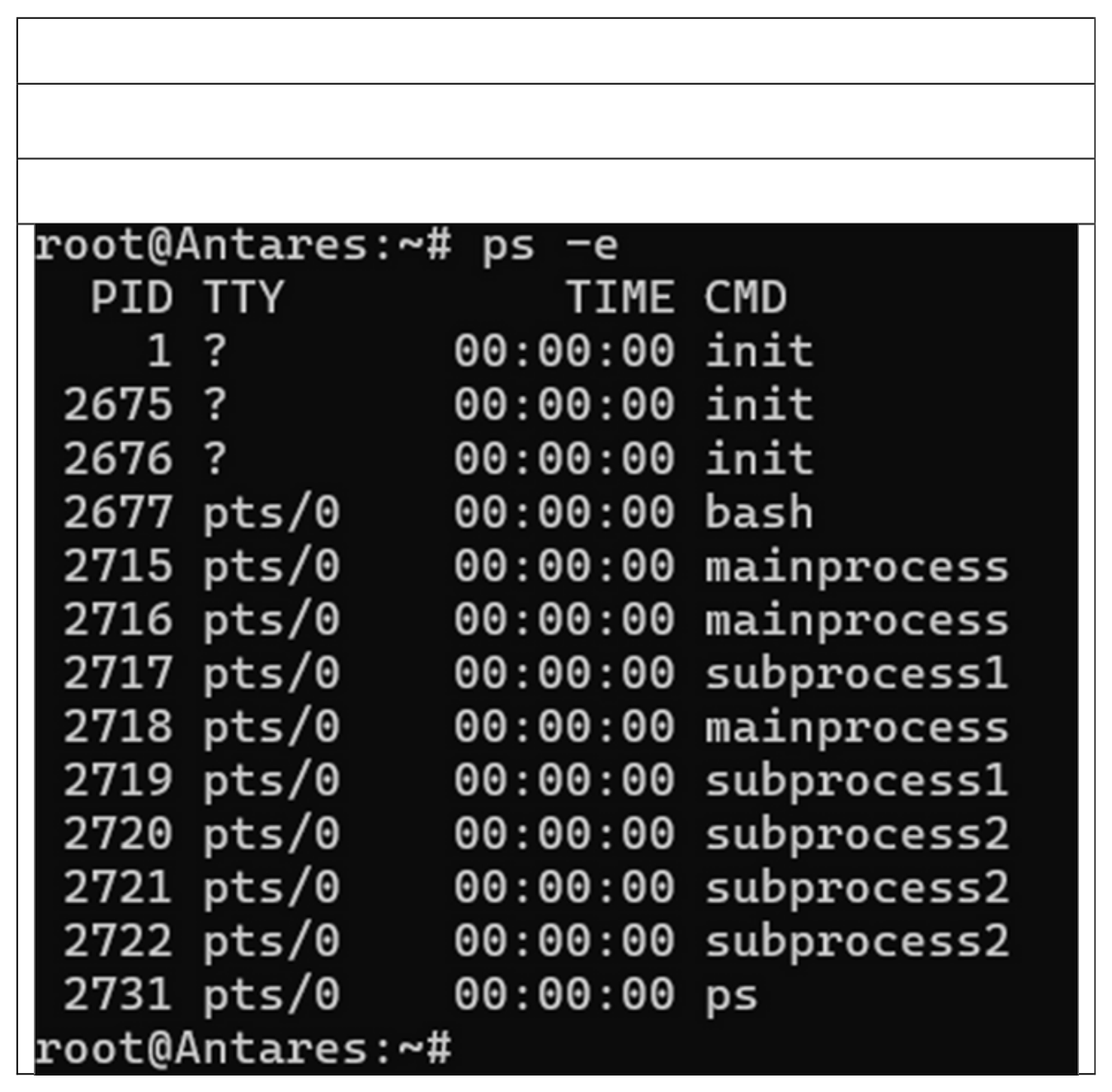
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Use the | gcc compiler to create executable files from the source code. | | | |
|  | |  | | --- | | gcc mainprocess.c -o mainprocess gcc | | | | | |
| subprocess1.c | | -o | subprocess1 | gcc |
| subprocess2.c -o subprocess2 | | | | |
| 3. | Execute the dummy processes: | | | | |

Run all the dummy processes

|  |
| --- |
| ./mainprocess & |

Press enter two times.

|  |  |
| --- | --- |
| 4. | The dummy processes are running for 2 hours. If you took longer than 2 hours on questions 1-9, please restart the main process with ./mainprocess &. |

Lab Assessment 2 : Linux Process Monitoring and Management 

Instructions:

1. Carefully execute each command as instructed in the questions.   
2. Write down the exact command used for each task.

3. Capture a screenshot of the command's output.

Question 1

Use the ps command with the appropriate option to display a complete list of all running processes within the Linux operating system.

Command

ps -e

|  |
| --- |
| Output |

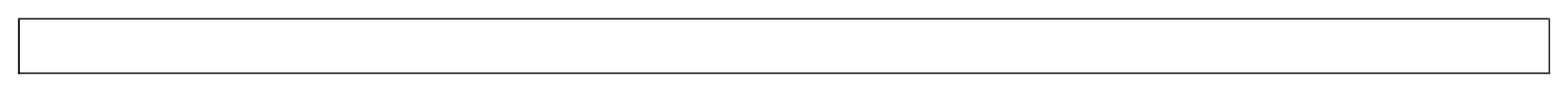
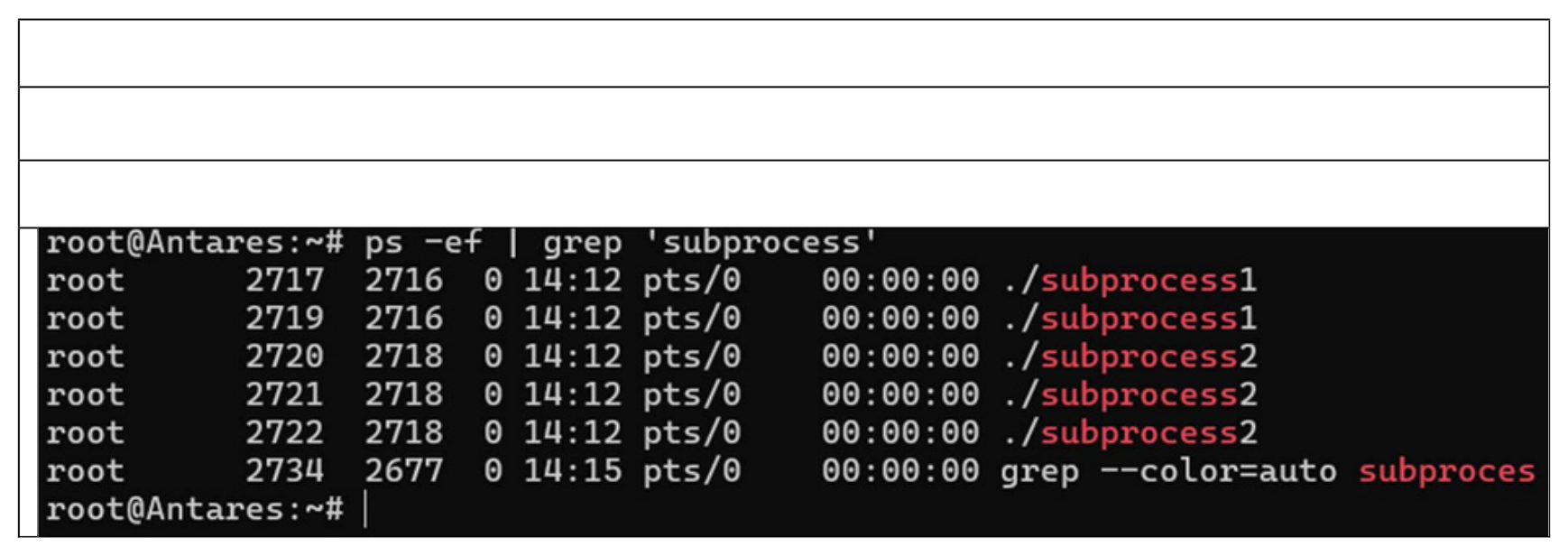
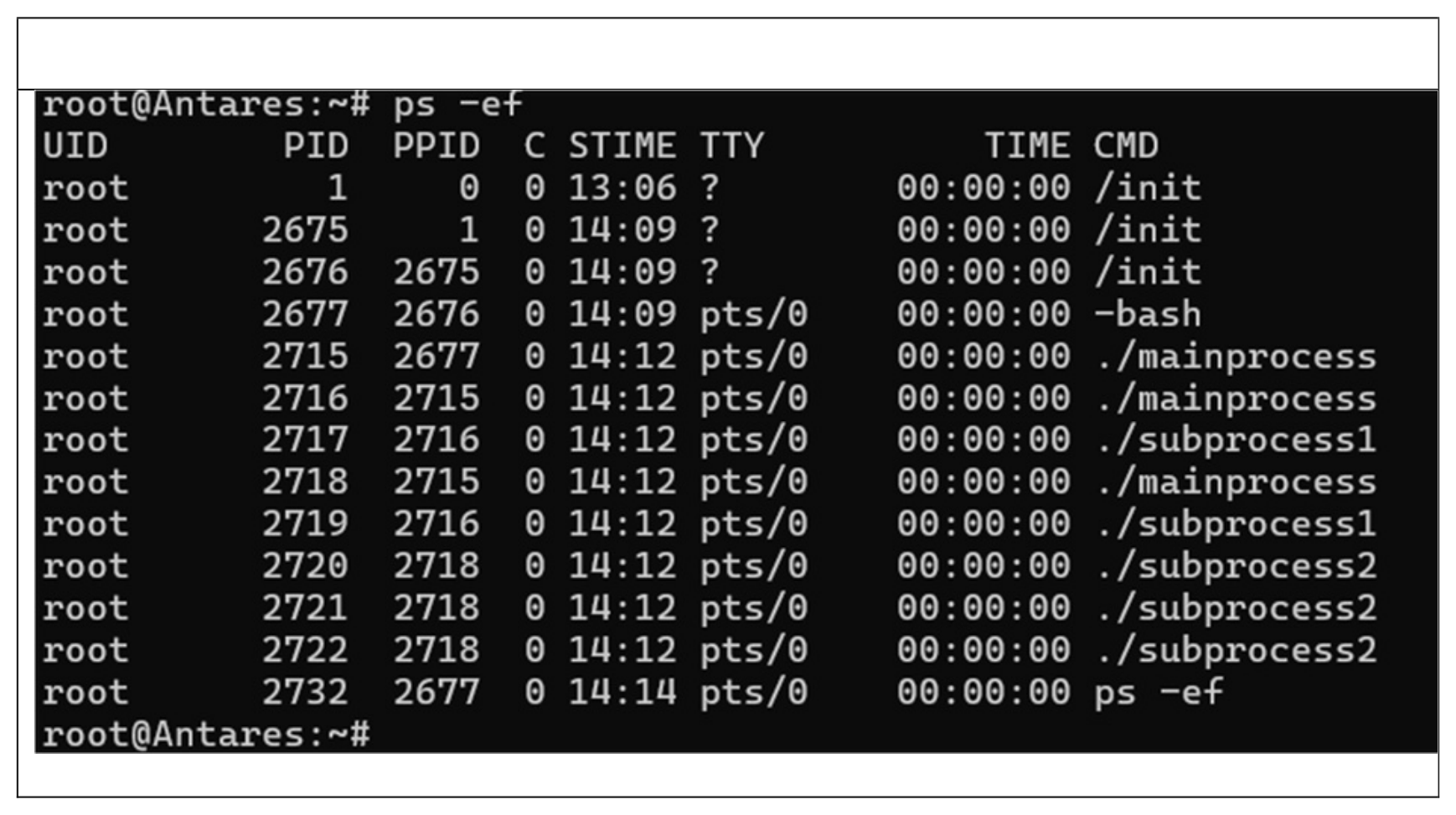
Question 2

Employ the ps command with necessary options to unveil comprehensive details about each

running process.

|  |
| --- |
| Command |

|  |
| --- |
| ps -ef |



|  |  |
| --- | --- |
| |  | | --- | | Output | |

Question 3   
Use the ps command with some tools to only list processes named "subprocess" and show some info about them.

|  |
| --- |
| Command |

|  |
| --- |
| ps -ef | grep ‘subprocess’ |

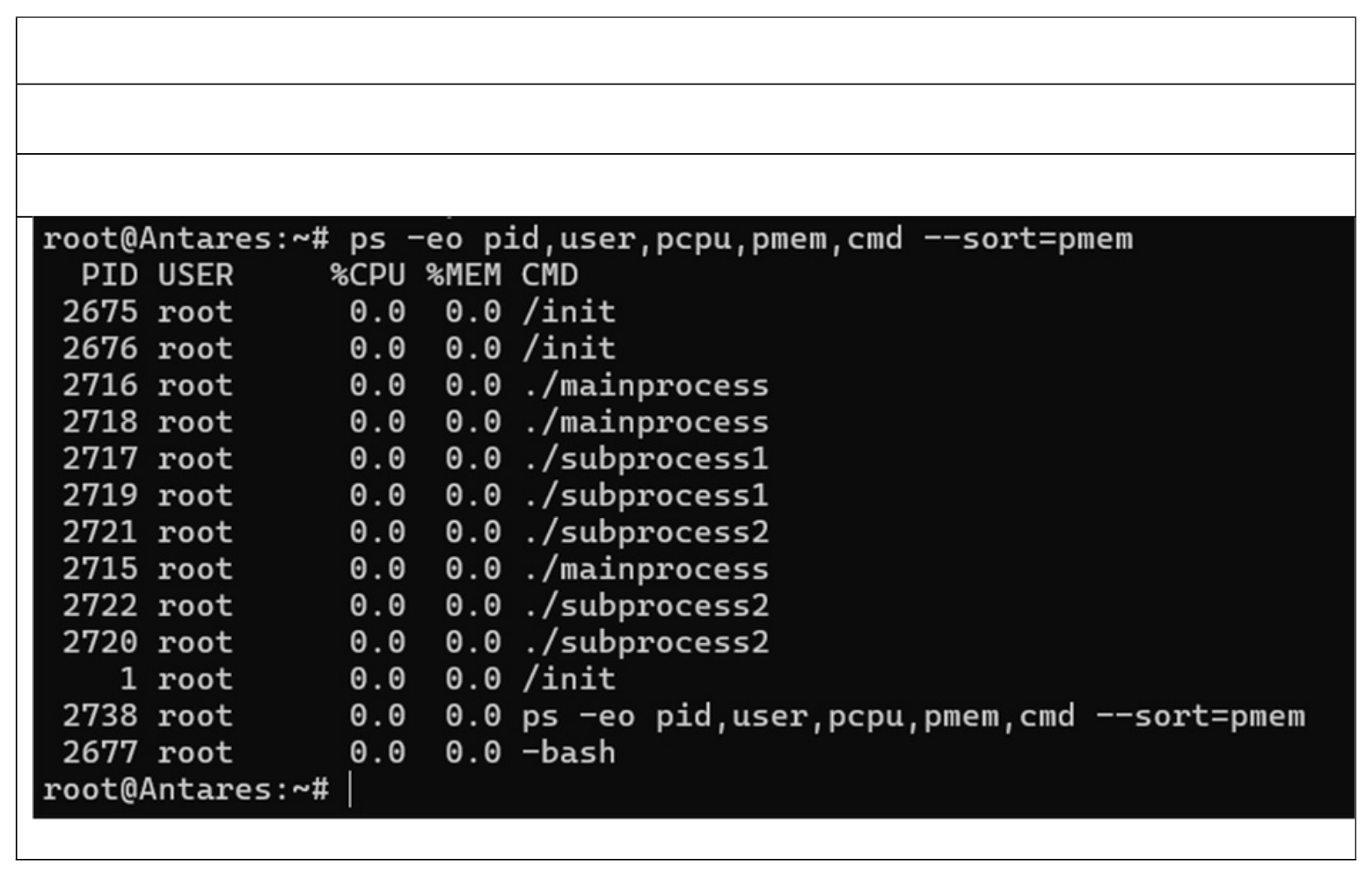
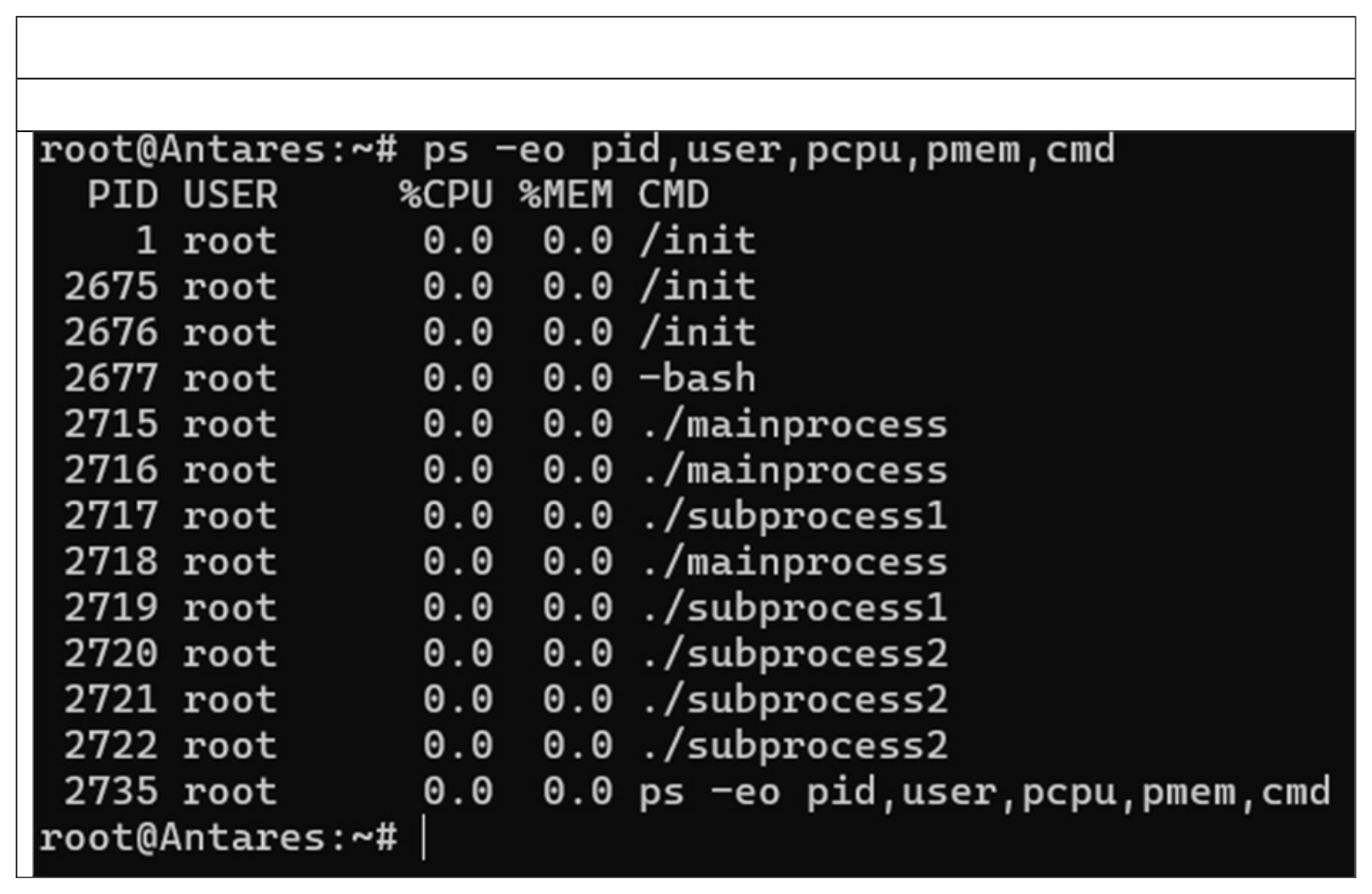
|  |
| --- |
| Output |

Question 4

Execute the ps command, specifying options that reveal only the following columns:

Process ID (pid)   
Owner of the process (user)   
CPU percentage (pcpu)   
 Memory percentage (pmem)   
 Command (cmd)

|  |  |
| --- | --- |
| |  | | --- | | Command | |



|  |
| --- |
| ps -eo pid,user,pcpu,pmem,cmd |

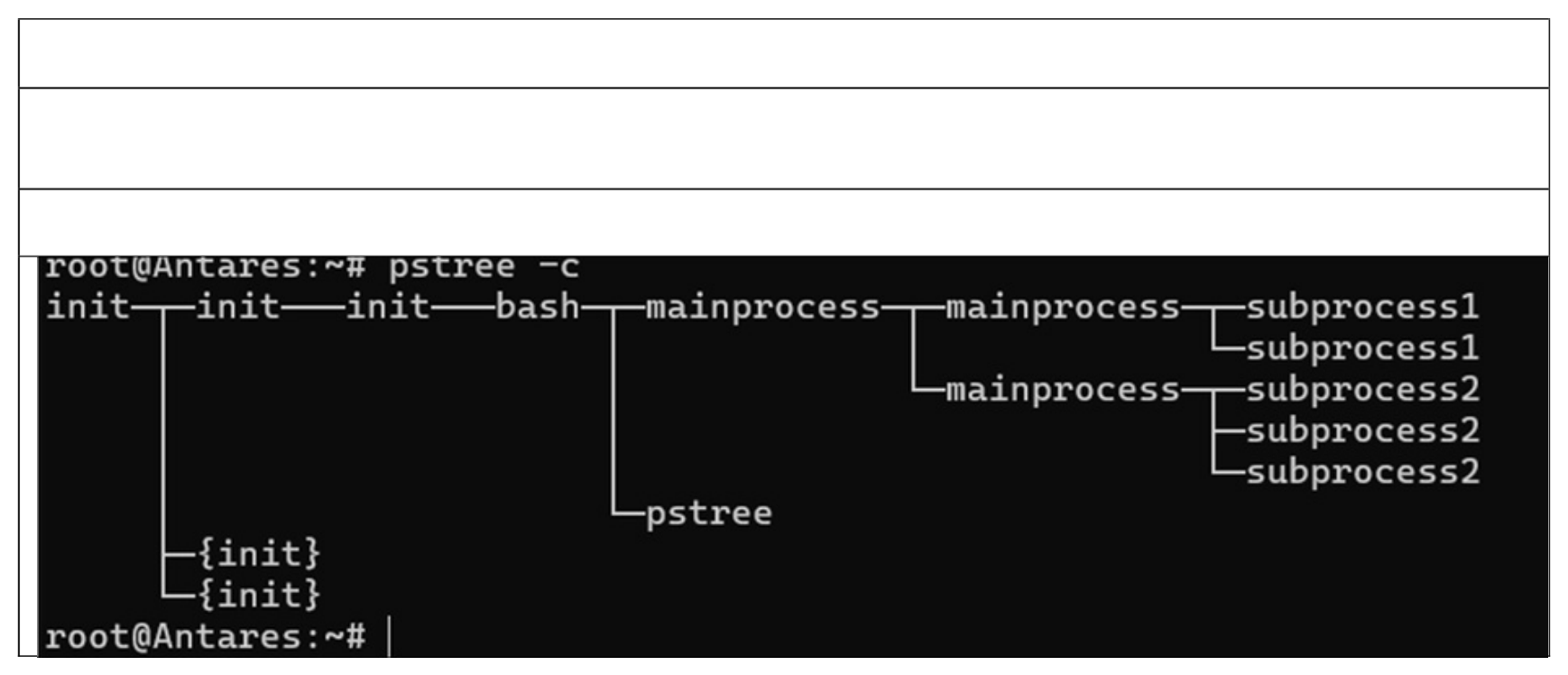
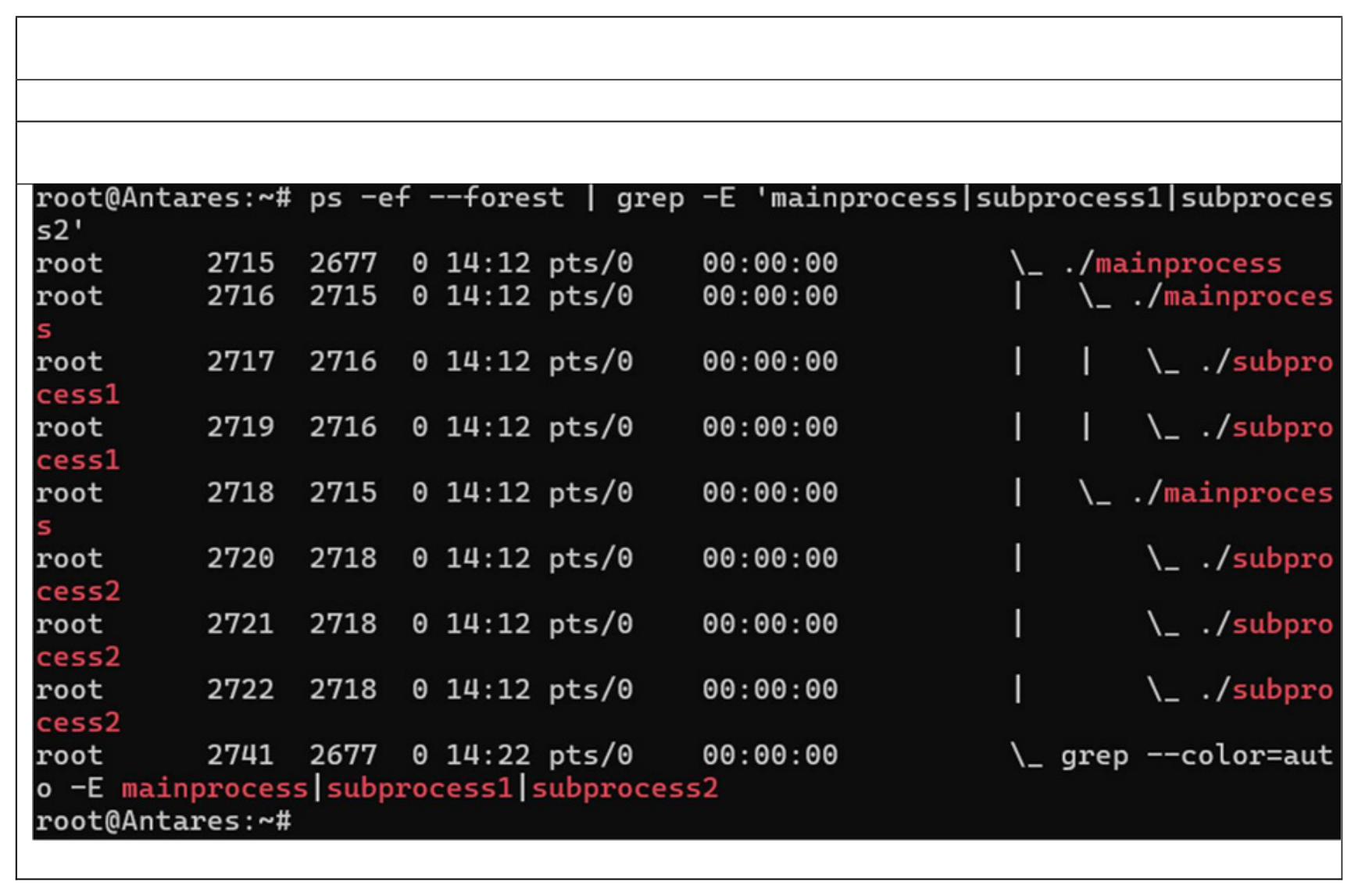
|  |
| --- |
| Output |

Question 5   
Building on the ps command used in Question 4, can you add an option to sort the listed processes by their memory usage (pmem)?

Command

|  |
| --- |
| ps -eo pid,user,pcpu,pmem,cmd –-sort=pmem |

|  |
| --- |
| Output |

Question 6   
Construct a command using ps, suitable options, and any additional tools to visualize the hierarchical structure (tree-like) of the following processes: 

 "mainprocess"

|  |  |
| --- | --- |
|    | "subprocess1"  "subprocess2" |

Command   
ps -ef --forest | grep -E 'mainprocess|subprocess1|subprocess2'

|  |
| --- |
| Output |

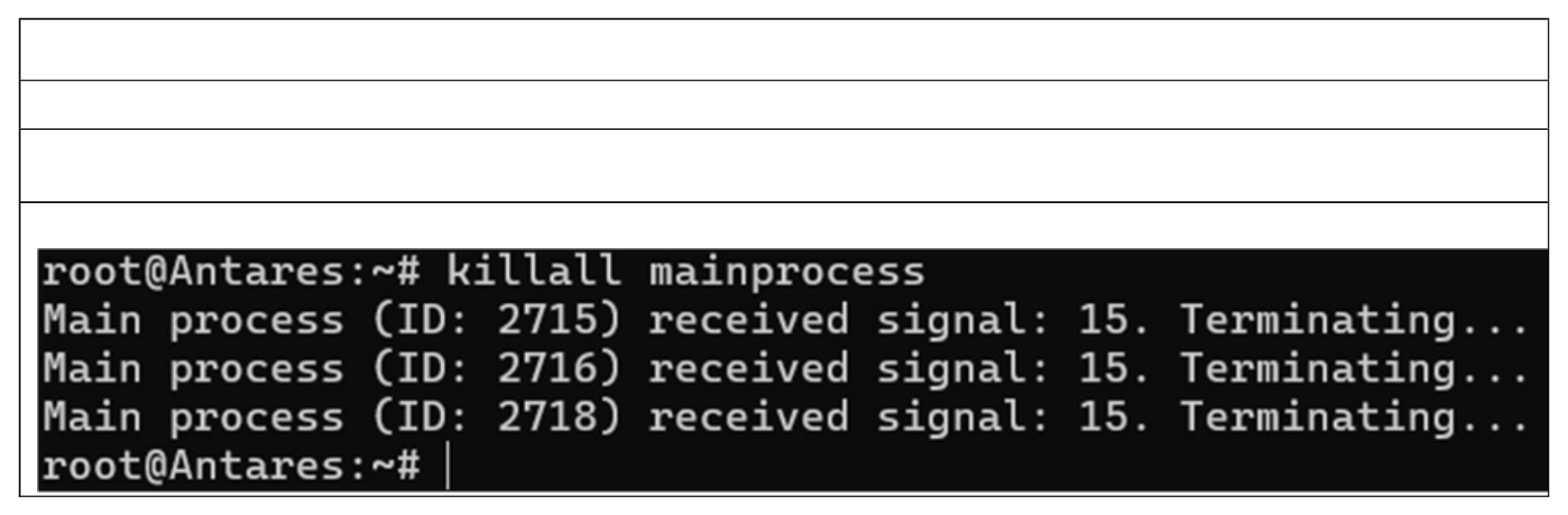
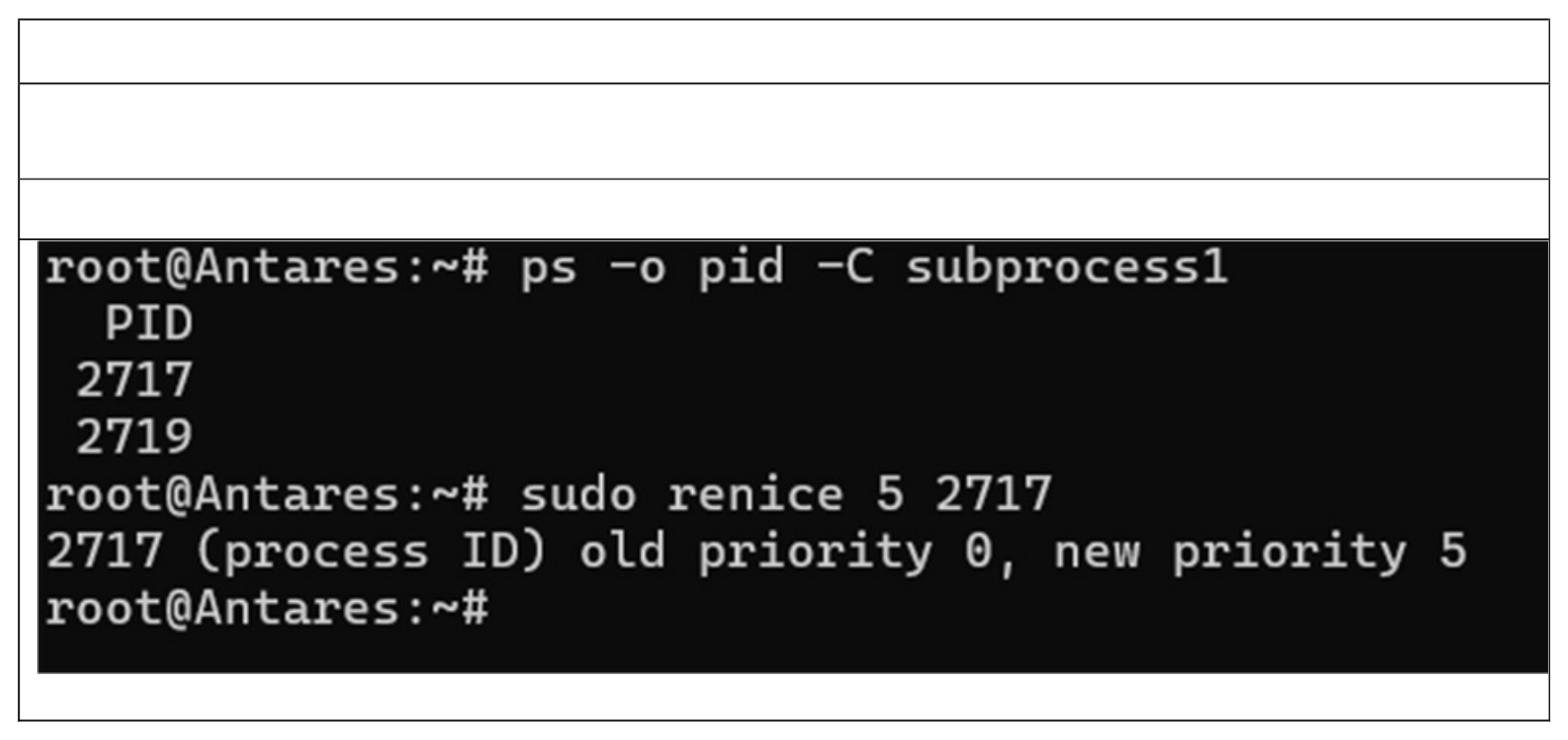
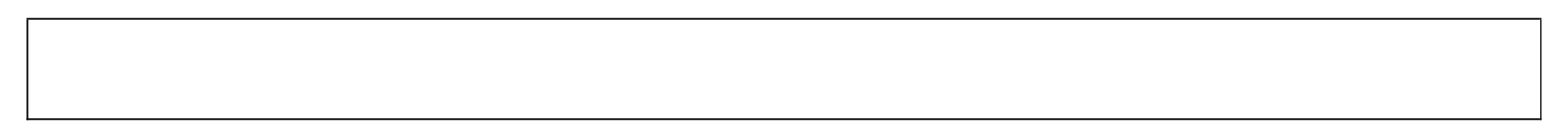
Question 7

|  |  |  |
| --- | --- | --- |
| Use | pstree | command with option that show the number of threads to each process. |

|  |
| --- |
| Command |

|  |
| --- |
| pstree -c |

|  |
| --- |
| Output |



Question 8

|  |  |  |
| --- | --- | --- |
| Use | renice | command to change priority level of one of process “subprocess1”. |

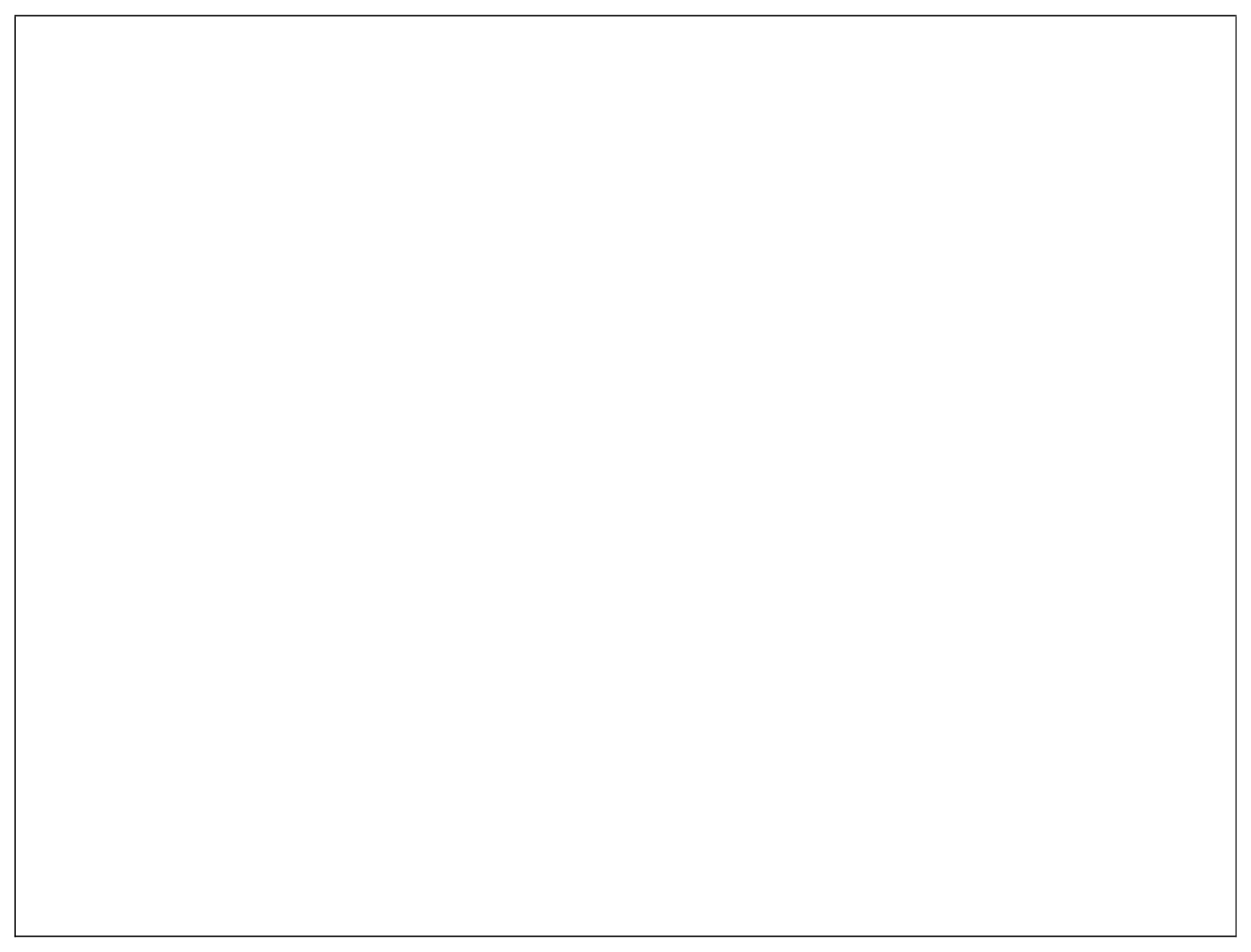
|  |
| --- |
| Command |

|  |
| --- |
| sudo renice 5 2717 |

|  |
| --- |
| Output |

Question 9   
Terminate all running processes with the name “mainprocess”.

|  |  |  |
| --- | --- | --- |
| |  | | --- | | killall mainprocess | | Command  Output |

Question 10   
Write a short C or Python code (choose only one language) demonstrating multiprocessing with fork() and wait(). Compile and/or run the code. Show the output. 

Source Code:

Nano process.py

import multiprocessing   
import os   
def child\_process():

|  |
| --- |
| print(f"Hello from the child process! PID: {os.getpid()}") |

|  |
| --- |
| if \_\_name\_\_ == "\_\_main\_\_":  # Create a new process  process = multiprocessing.Process(target=child\_process) process.start() # Start the child process  process.join() # Wait for the child process to finish |

|  |
| --- |
| print(f"Hello from the parent process! PID: {os.getpid()}") |

|  |
| --- |
| python3 process.py -o process |

Output:

