Course COMP-8567
Assignment 02
Winter 2026
Due Date: Mar/4/2026, 11 PM
50 Marks
Plagiarism Detection Tool: MOSS
MOSS- Acceptable Similarity Percentage: <=6%

Associated Learning Outcomes:

- Apply OS concepts to design algorithms to solve systems programming problems in a variety of different systems, such as Unix/Linux/Android environments.
- Correctly define systems programming problems and identify and apply appropriate solutions approaches.
- Design and implement solutions that use the hardware and/or kernel services to solve systems programming problems involving the latest computing technologies.

Note: Please check the following link for the **complete list** of learning outcomes for COMP 8567
https://ctl2.uwindsor.ca/cuma/public/courses/pdf/ee1b450a-23a6-4635-b0c6-40a47a21331f

Please read these three points extra carefully:

1. Just a reminder that, like all labs/assignments/project, this assignment **must be implemented** on our CS Linux server using your official university login.
2. Since this assignment involves creating several processes to test your program, you might inadvertently create a **chain of processes** that might lead to what is known as a **"fork bomb"** which uses up a lot of system resources.
3. Regardless, it is your sole responsibility to execute the statement: **$killall -u *username*** periodically/mandatorily **when you are done working on this assignment on a given day,** failing which zero marks will be given to the assignment if there are any complaints from the system administrator

   Note: You can initially test your program using **a2sampletree.c,** which will be distributed on Brightspace. However, your program should run on any process tree created under a BASH terminal.

Write a C program ptree26w.c that searches for processes in the process tree (rooted at a specified process) and prints the requested information based on the input parameters.

Synopsis :

ptree26w [*root_process]* [*process_id*] [Option]

//Note: both *root_process* and *process_id* must be descendants of the same BASH terminal process

- **When [Option] is <u>not</u> provided:** Lists the PID, PPID of *process_id* **<u>if</u>** *process_id* belongs to the process subtree rooted at *root_process,* else print "Process *process_id* does not belong to the process subtree rooted at *root_process*"
- **When [Option] is provided:** Please **see the next section** for the **additional** action to be performed for each option.

Option:

Rules for Options

1. In any of the following options, if process_id does not belong to the process tree rooted at root_process , you need to print *" The process (list the process_id) does not belong to the process tree rooted at (list the root_process)"*
2. *The required action is to be performed only if the <u>pertinent user process is available and you have the permission to perform the required action</u>.*
    a. ***Ex:** if you try to kill the parent of a process using -kpp and the said process happens to be an orphan process, you (obviously) cannot kill **init**, and a suitable message must be displayed.*
3. *Any of the commands/options must not terminate BASH process/s, and an appropriate message must be displayed.*
4. *Use **SIGKILL** to perform all kill operations*

-cnt lists the count of all descendants of process_id

-oct List the total count of the orphan processes currently running in the process subtree rooted at process_id

-dtm Terminate all the descendant processes of process_id in the reverse chronological order of the creation time of the processes (Newly created processes must be terminated first)

-odt lists the process ID and the creation time of the oldest descendant of process_id

-ndt lists the process ID of the most recently created descendant of process_id

-dnd lists the count of all the **non-direct** descendants of process_id

-sst Send SIGSTOP to all the siblings of process_id

-sco All siblings of process_id currently stopped are continued using SIGCONT

-kgp Kills the grandparent of process_id

-kpp kills the parent of process_id

-ksp kills all the siblings of process_id

-kps kills all the siblings of process_id's parent

-kgc kills all the grandchildren of process_id

-kcp kills all the children of process_id

-krp root_process is killed using SIGKILL

-mmd List the descendant/s of process_id currently consuming the most memory (with the maximum VmRSS value), along with the VmRSS value. In the event of a tie, list all the processes

//VmRSS: Virtual Memory Resident Set Size

-mpd List the **descendant/s of process_id** that have consumed the **most cumulative CPU time** so far, and list the time consumed in clock ticks. In the event of a tie, list all the processes.

Note: *All of the above options must comply with the aforementioned* ***"Rules for Options"***

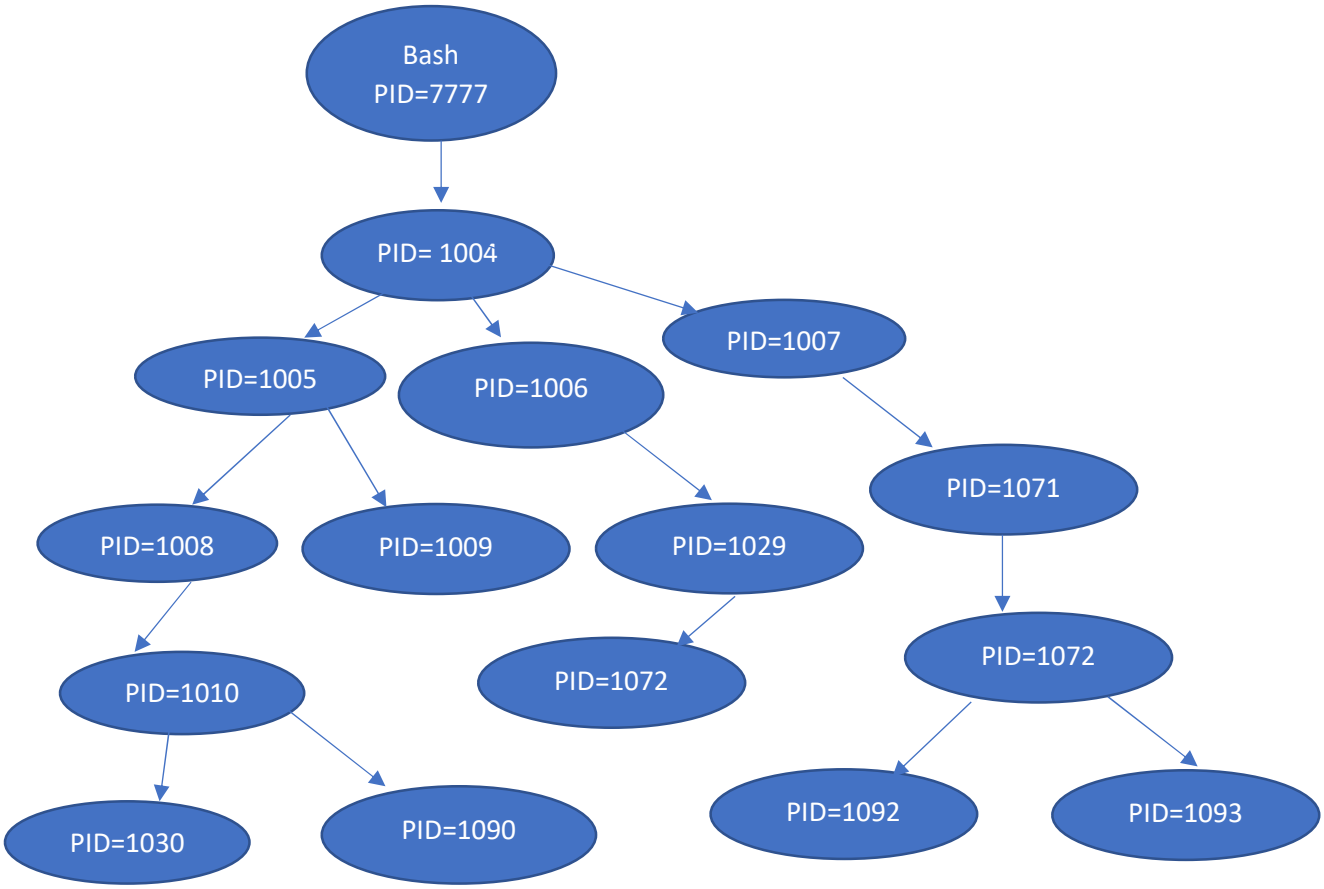
Additional Commands

ptree26w -bcp (No other arguments) *count of the number of processes started under the current bash //Includes nested children, background processes, etc*

ptree26w -bop (No other arguments) *count the overall number of processes in all open bash terminals (excluding the bash processes in the count)*

Note: This is an example only. Your assignment must work for all valid user process trees as per the requirement.

Sample Runs



$ ptree26w 1004 1008
1008 1005
 	
$ ptree26w 1004 1010 -cnt
1010 1008
2
 	
$ ptree26w 1004 1010 -kgp
1010 1008
1005 is terminated
 	
$ ptree26w 1004 1005 -kpp
1005 1004
Parent is BASH and will not be terminated
 	
$ ptree26w 1071 1090
Process 1090 does not belong to the process subtree rooted at 1071
 	
$ ptree26w 1006 1072 -kpp
Process 1072 does not belong to the process subtree rooted at 1006
 	
$ ptree26w 1004 1008 -mmd
1090 is the descendant of 1008 consuming the most memory.
VmRSS: 5232000 bytes
 	
$ ptree26w 1004 1010 -dtm
//Processes 1090 and 1030 must be terminated
	$ ptree26w 1005 1010 -odt
	1010 1008
	Oldest descendant of 1010 is 1030, whose creation time is Wed 15 Oct 2025 06:26:08 AM EDT
	
	$ ptree26w 1004 1004 -kgp
	1004 7777
	Grandparent is not a user process and will not be terminated
	
	$ ptree26w 1004 1010 -kpp
	1010 1008
	SIGKILL was sent to process 1008
	
	$ ptree26w 1004 1072 -kcp
	1072 1071
	SIGKILL was sent to the following child process(s)
	1092 1093
	
	$ptree26w -bcp
	15
	
	$ ptree26w 1004 1008 -ksp
	1008 1005
	SIGKILL was sent to the following sibling process/s of 1008
	1009
	
	$ ptree26w 1004 1007 -dnd
	1007 1004
	3
	
	$ ptree26w 1004 1007 -mpd
	1072 is the descendant of 1007 that has used the most CPU time.
	Total CPU time: 521 clock ticks

Explanation of the program

-You are required to include adequate and appropriate comments to explain the working of the program.
-Please see the assignment rubrics for more information

Submission Instructions:

Plagiarism Detection Tool: MOSS
MOSS- Acceptable Similarity Percentage: <=6%

You need to submit the following:
1. A2_firstname_lastname_SID.c
3. Zoom/Google Drive recording link explaining the following (not more than 15 minutes)

- Your camera must be switched on.
- Overall working of the code and various modules (around 8-9 minutes)
- Execution of the code under various inputs/conditions as per the requirements of the assignment (around 6-7 minutes)
- Other form of links/MP4 files will NOT be acceptable.
- Include the link in the COMMENTS section.