TO DO

Part1: Write a C program “hw1p1.c” to display the total of vrss sizes for all processes in the system using the ps command.

Part2: Write a C program “hw1p2.c” to display the total of vrss sizes for all processes in the system by extracting the information from the status file of each process.

SUNMISSION

Save the source code of two programs, hw1p1.c and hw1p2.c, in a folder with the name hw1_yourQUID. Compress this folder .tar.gz and submit it on the Blackboard system of the course.

HOW TO DO IT

hw1p1.c: you need your C program to do the following in order to achieve this desired output.

(1) navigate the subfolders of the proc folder looking for those folders holding information about processes in the system. Remember that proc folder has such folders, named after the process pid number, and that the proc folder has other folders hosting data for other purposes.

(2) each time you locate a process folder you need to create a constant string array to hold the parameters needed to execute the proper ps command to retrieve the rss information for a process with specific pid. The command to retrieve that with ps is "ps -p pid -o rss". The array should have these entries of the ps command as its elements and that last element in the array should be the NULL.

```
char *const parmList[] = {"ps","-p",pid,"-o","rss",NULL};
```

in this case pid is a string that you need to form based on the process pid which is the name of the folder.

(3) next you need to execute the ps command. To do so, you need to create a child process for this purpose who will then execute a proper exec system call that accepts two arguments, the path of the command to execute "/bin/ps" and the array you created earlier. This will be the execvp system call

```
execvp("/bin/ps", parmList);
```

The child needs to send its output to the parent then terminates hence, a pipe should be created before forking the child and a proper close/dup should be performed in the child code to get the result of execvp written to the pipe.

(4) the parent should wait to read the output of the child from the pipe and extract from it the value of rss without the heading then add that value to total.

(5) repeat the steps (2) to (4) for all folders holding information about processes in the system.

(6) display the total rss on the screen.

hw1p2.c: you need your C program to do the following in order to achieve this desired output.

(1) navigate the subfolders of the proc folder looking for those folders holding information about processes in the system. Remember that proc folder has such folders, named after the process pid number, and that the proc folder has other folders hosting data for other purposes.

(2) each time you locate a process folder you need to form a string hosting the path to the status file in that folder.

(3) use the path string to open the file.

(4) Remember that that the value of rss size is in the status file, open the file in gedit to understand the formats of it and to know how to locate the value of rss. Having done so, you can now code how to extract such information by read the file line by line. You can do that using a proper i/o method from those who are definded in stdio.h, for example, fscanf.

(5) each time you read a line, check if it contains the rss header which is "VmRSS:". You can do that with the use of the C string function strstr. If the line contains the substring "VmRSS:" then it is the desired line having the value needed hence, you need to read the value using proper i/o method from those who are defined in stdio.h, for example, sscanf.

(6) add the retrieved value of rss to the total.

(7) repeat the steps (2) to (6) for all folders holding information about processes in the system.

(8) display the total rss on the screen.