**Subiecte de pregătire către lucrarea de laborator 5**

**Explicaţi noţiunea de:**

Process - Un proces reprezintă un program în execuție și are ataşate o serie de informaţii specifice precum instrucțiunile programului, resurse folosite (precum fişiere deschise), unul sau mai multe fire de execuție şi alte informații necesare procesului de execuţie în paralel.

PID - sau Process ID, este un identificator de proces sub forma unui număr întreg unic (între 0 și 65535.)

**Instrucţiunile:**

**Ps**- Afişarea informaţiilor despre procese. O listare a proceselor poate fi efectuată prin comanda ps (ps - process status):

**ps [opţiuni]**

Principalele opţiuni sunt:

**-aux** - afişează informaţii despre toate procesele din sistem;

**-l** - afişează informaţii despre procese, diferită de –u;

**-u** - afişează informaţii despre procese, dar diferită de –l;

**-e** - afişează informaţii despre toate procesele din sistem utilizând sintaxa standardă.

**Pstree -** Comanda pstree afişează toate procesele care rulează sub forma unui arbore (tree). Dintre opțiunile cele mai des folosite:

**− n** - sortarea proceselor după identificatorul procesului (pid);

**− p** – afișarea PID, numere zecimale în paranteze, după fiecare nume de proces;

**− u** - afișarea identificatorului utilizatorului (uid).

**Top -** Comanda top este utilizată pentru monitorizarea proceselor în timp real, actualizând informațiile la fiecare 5 secunde. Informaţia afişată poate fi controlată de la tastatură. Tastaţi h şi primiți un „help”.

Comanda are multe opțiuni, dintre care:

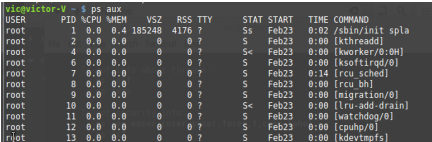
**− d** (delay) - specifică intervalul dintre actualizările ecranului;

**− n** - numărul de iterații;

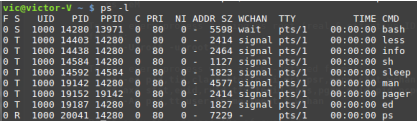
**− p** (process identifier) - identificatorul procesului, care va fi urmărit.

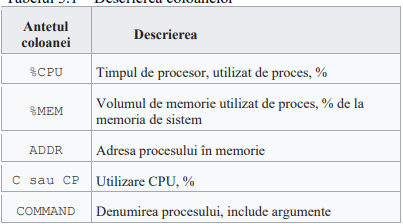
**+ INFO :**

**ps -aux**



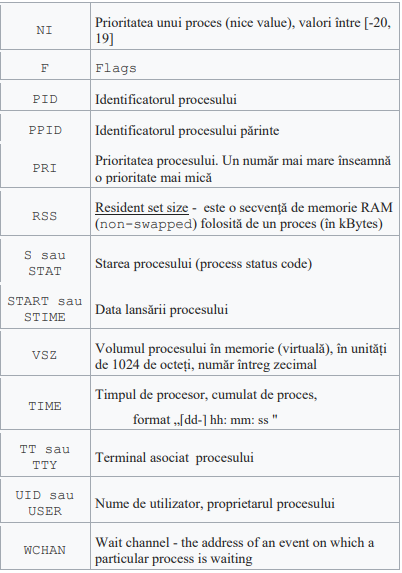
**ps -l**



− R – procesul se execută la moment;

− D – proces în regim „sleep” (în așteptarea de operațiuni de intrare /ieşire);

− I – procesul este inactiv;

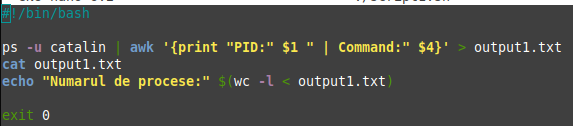
− S - proces întrerupt (așteptare pentru finalizarea unui eveniment);

− s - frecvent utilizat;

− < - prioritate înaltă;

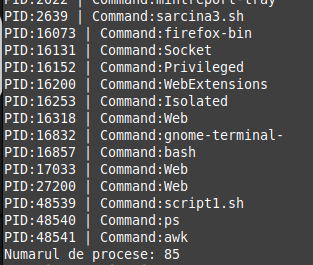
− Z – proces zombi.

Mai multe informaţii pot fi obţinute: **ps --help** sau man ps. Comanda ps arată doar procesele active la momentul în care ați introdus această comandă, adică este o “fotografie” a proceselor la momentul respectiv.

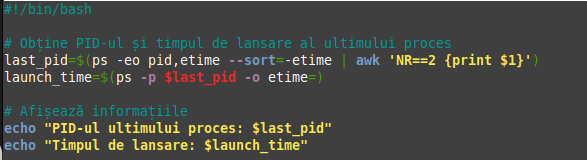
1. Aflaţi numărul de procese inițiaţializate de utilizatorul ***user*** și introduceți într-un fișier perechea „PID: comanda” a proceselor inițializate. Afișați conținutul acestui fișier şi numărul de procese.
2. 

* ps -u catalin | awk '{print "PID:" $1 " | Command:" $4}' > output1.txt: Această comandă combină două utilitare în linie de comandă.
  + ps -u catalin: Afișează informații despre procesele asociate utilizatorului "catalin".
  + awk '{print "PID:" $1 " | Command:" $4}': Procesează ieșirea comenzii ps, utilizând utilitarul AWK pentru a extrage și formata informațiile. Aceasta va afișa PID-ul (ID-ul de proces) și comanda asociată fiecărui proces, separate prin "|". Aceste informații sunt redirecționate în fișierul "output1.txt" cu operatorul >.
* echo "Numarul de procese:" $(wc -l < output1.txt): Afișează numărul de linii din fișierul "output1.txt", care reprezintă numărul de procese găsite.
  + $(wc -l < output1.txt) este o substituție de comenzi care returnează numărul de linii din fișier.

Opțiunea **-u** catalin filtrează rezultatele comenzii ps pentru a afișa doar procesele asociate utilizatorului "catalin". Apoi, cu ajutorul utilitarului awk, sunt extrase și formatate informațiile relevante despre aceste procese, care sunt ulterior redirecționate în fișierul "output1.txt".



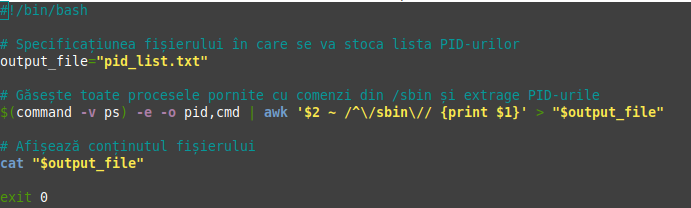
1. **2.** Afișați PID-ul procesului, ultimul lansat (cu timpul de lansare).



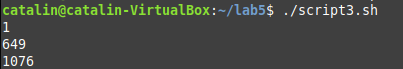
* last\_pid=$(ps -eo pid,etime --sort=-etime | awk 'NR==2 {print $1}'): Această comandă combină mai multe funcționalități:
  + ps -eo pid,etime --sort=-etime: Afisează PID-ul și timpul de execuție (elapsed time) pentru toate procesele în sistem, sortate în ordine descrescătoare după timpul de execuție. Astfel, primul rând va conține informații despre ultimul proces lansat.
  + awk 'NR==2 {print $1}': Procesează ieșirea comenzii ps utilizând AWK pentru a extrage PID-ul din al doilea rând (NR==2) și îl stochează în variabila last\_pid cu ajutorul construcției $(...).
* launch\_time=$(ps -p $last\_pid -o etime=): Aceasta utilizează comanda ps pentru a obține timpul de execuție al procesului cu PID-ul salvat în variabila last\_pid.
  + ps -p $last\_pid -o etime=: Afisează timpul de execuție (elapsed time) al procesului cu PID-ul specificat.
  + launch\_time=$(...): Stochează timpul de execuție al procesului în variabila launch\_time cu ajutorul construcției $(...).
* Comanda **ps** este folosită pentru a afișa informații despre procesele care rulează în sistemul de operare.
* Opțiunea **-e** indică faptul că se vor afișa toate procesele,
* **-o** este folosită pentru a specifica câmpurile (colonile) pe care dorim să le afișăm în ieșire.
* **pid**: ID-ul de proces (PID).
* **etime**: Timpul total scurs de la lansarea procesului (elapsed time).



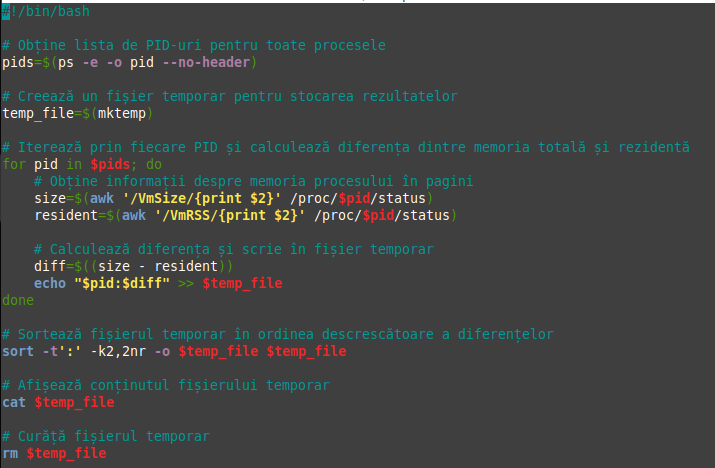
1. **3.** Introduceți într-un fișier o listă cu PID-urile proceselor, pornite cu comenzile localizate în /sbin/. Afișați conținutul acest fișier.



* $(command -v ps) -e -o pid,cmd | awk '$2 ~ /^\/sbin\// {print $1}' > "$output\_file": Aceasta este comanda principală care face majoritatea operațiilor:
  + $(command -v ps): Acest subcomandă (command -v ps) este folosită pentru a obține calea către comanda ps. command -v este o modalitate de a obține calea completă a unei comenzi.
  + -e: Afișează informații despre toate procesele, nu doar cele asociate cu terminalul curent.
  + -o pid,cmd: Specifică câmpurile (colonile) pe care dorim să le afișăm în ieșire. În acest caz, dorim să vedem PID-ul și comanda asociată fiecărui proces.
  + | awk '$2 ~ /^\/sbin\// {print $1}': Utilizează AWK pentru a filtra procesele și a extrage doar PID-urile proceselor ale căror comenzi încep cu "/sbin/". Expresia $2 ~ /^\/sbin\// verifică dacă al doilea câmp (comanda) începe cu "/sbin/". Dacă condiția este îndeplinită, se va afișa PID-ul procesului.
  + > "$output\_file": Redirecționează ieșirea rezultată în fișierul specificat de variabila output\_file.
  + ^ - În comanda awk: $2 ~ /^\/sbin\// verifică dacă al doilea câmp începe cu "/sbin/".
* $2: Acesta reprezintă al doilea câmp (coloană) din fiecare linie a fișierului de intrare.
* ~: Este operatorul pentru a verifica dacă un șir se potrivește cu un anumit șablon sau expresie regulată.
* /^\/sbin\//: Este o expresie regulată care specifică că al doilea câmp ($2) trebuie să înceapă cu șirul "/sbin/".
* {print $1}: Această secțiune specifică acțiunea care trebuie întreprinsă atunci când condiția este îndeplinită. În acest caz, se afișează primul câmp (coloană) al liniei curente.



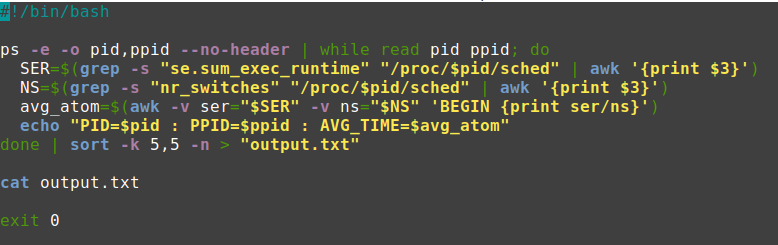
**4.** Pentru fiecare proces, calculați diferența dintre memoria totală (statm: size) şi rezidentă (statm: resident) ale memoriei de proces (în pagini). Introduceți într-un fișier linii de tip „PID:diferența”, sortate în ordinea descrescătoare a acestor diferențe. Afișați conținutul acestui fișier.



* pids=$(ps -e -o pid --no-header): Aceasta obține lista de PID-uri pentru toate procesele care rulează pe sistem și salvează rezultatul în variabila pids. Opțiunile folosite pentru comanda ps sunt:
  + -e: Afișează informații despre toate procesele.
  + -o pid: Specifică să se afișeze doar PID-ul fiecărui proces.
  + --no-header: Specifică să nu afișeze linia de antet cu numele coloanelor.
* temp\_file=$(mktemp): Aceasta creează un fișier temporar folosind comanda mktemp și salvează calea către acest fișier în variabila temp\_file. Fișierul temporar va fi utilizat pentru a stoca rezultatele intermediare.
* for pid in $pids; do ... done: Acest bloc for iterează prin fiecare PID obținut în pasul anterior și realizează operațiile următoare pentru fiecare proces:
  + size=$(awk '/VmSize/{print $2}' /proc/$pid/status): Afișează, cu ajutorul awk, dimensiunea totală a memoriei procesului (VmSize) în pagini și stochează rezultatul în variabila size. VmSize face match cu linia care contine textul vmrss
  + resident=$(awk '/VmRSS/{print $2}' /proc/$pid/status): Afișează, tot cu awk, dimensiunea memoriei rezidente (VmRSS) în pagini și stochează rezultatul în variabila resident.
  + diff=$((size - resident)): Calculează diferența dintre dimensiunea totală și dimensiunea rezidentă și stochează rezultatul în variabila diff.
  + echo "$pid:$diff" >> $temp\_file: Scrie în fișierul temporar o linie care conține PID-ul procesului și diferența calculată între dimensiunea totală și cea rezidentă.
* sort -t':' -k2,2nr -o $temp\_file $temp\_file: Sortează fișierul temporar în ordinea descrescătoare a diferențelor calculate. Opțiunile pentru comanda sort sunt:
  + -t':': Specifică separatorul ":" pentru câmpurile de sortare.
  + -k2,2nr: Specifică că sortarea se face după al doilea câmp ($diff), numeric (n), în ordine descrescătoare (r).
  + k2,2 sortarea dupa a doua coloana
  + -o $temp\_file: Specifică să se suprascrie fișierul temporar cu rezultatele sortării
* rm $temp\_file: Șterge fișierul temporar pentru a elibera resursele utilizate.



5.



**ps**: Comanda pentru a afișa informații despre procese.

**-e**: Listează informații despre toate procesele.

**-o pid,ppid**: Specifică formatul de ieșire pentru a afișa doar coloanele PID (identificatorul procesului) și PPID (identificatorul procesului părinte).

**--no-header**: Suprimă linia cu antet în rezultatul afișat.

**|**: Operatorul de canal, utilizat pentru a trimite ieșirea comenzii precedente ca intrare către următoarea comandă.

**while read pid ppid**; do: Inițiază un buclă while care citește fiecare linie a rezultatului (PID și PPID) și le atribuie variabilelor pid și ppid.

**SER=**: Atribuie rezultatul comenzii variabilei SER.

**grep -s "se.sum\_exec\_runtime" "/proc/$pid/sched":** Caută linia care conține "se.sum\_exec\_runtime" în fișierul "/proc/$pid/sched", unde $pid reprezintă ID-ul procesului curent.

**awk '{print $3}'**: Extrage și afișează a treia coloană a rezultatului.

**avg\_atom=**: Atribuie rezultatul comenzii variabilei avg\_atom.

**awk -v ser="$SER" -v ns="$NS" 'BEGIN {print ser/ns}':** Calculează timpul mediu per schimbare de context folosind valorile stocate în variabilele SER și NS.

**-v** specifica ca este variabila

**sort -k 5,5 -n:** Ordonează rezultatul în funcție de a cincea coloană (AVG\_TIME) în ordine numerică.