

Nombre: Josué Merino

NRC: 2537

Examen Práctico 2

Se crea un archivo .c para introducir el código, mediante el siguiente comando:



Se crea el archivo .c con el nombre caso_2.c, se digita el siguiente código y se lo guarda:

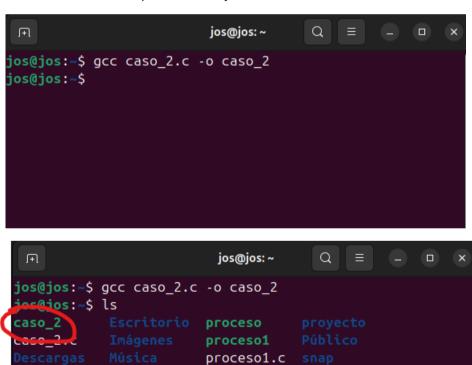
```
jos@jos: ~
                                                caso_2.c *
 GNU nano 7.2
#include <stdlib.h
#include <stdio.h>
int main(){
        size_t size = 400 * 1024 * 1024;
void *ptr = malloc(size);
        if(!ptr){
                 printf("Error: No se pudo asingar memoria \n");
return 1;
        printf("400 MB asignados. El programa esta generando carga constante.\n");
        while(1){
                 sleep(10);
        free(ptr);
        return 0;
                                                                                    ^C Ubicación
  Ayuda
                   Guardar
                                    Buscar
                                                  ^K Cortar
                                                                      Ejecutar
   Salir
                   Leer fich.
                                    Reemplazar
                                                     Pegar
                                                                      Justificar
                                                                                       Ir a línea
```

Se confirma que el archivo se creó:





Se procede a la ejecución del archivo:



Plantillas proceso.c

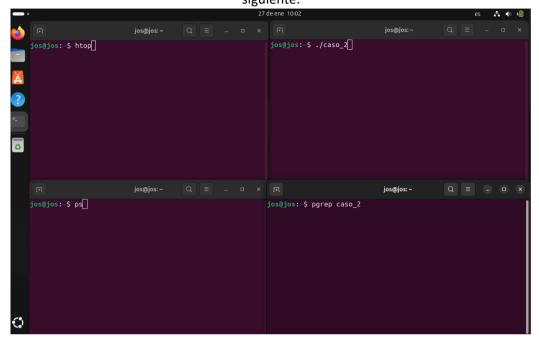
jos@jos:~\$



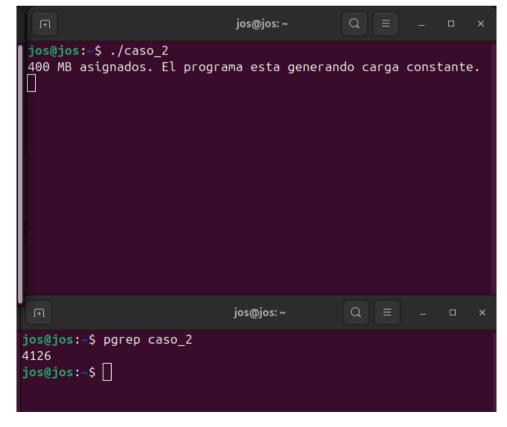
INNOVACIÓN PARA LA EXCELENCIA

Para el siguiente paso el entorno de interfaces de líneas de comandos que se manejará es el siguiente:

ECUADOR



Se procede a ejecutar con: ./caso_2:



Se verifica el proceso utilizando el comando ps -a:



Mediante el htop visualizamos los recursos que consume el proceso caso 2:

```
3%] Tasks: 111, 347 thr, 95 kthr; 1 running
                                                                          Load average: 0.03 0.14 0.11 Uptime: 00:23:19
                                704M 68200
                                               50852
                                                                        0:00.00
2851 jos
2852 jos
                                704M 68200
                                               0852
                                                                        0:00.00
                                               50852
3840
                                                                       0:00.01 /usr
0:00.07 bash
                                704M 68200
2854 jos
                     20
20
                                                                        0:00.71 gjs /usr/share/gnome-shell/extensions/ding@rastersoft.
                            0 2815M 62700
3310
                                               47984
                                                                        0:00.00 gjs /usr/share/gnome-shell/extensions/ding@raster
0:00.00 gjs /usr/share/gnome-shell/extensions/ding@raster
                              2815M 62700
                                               47984
                            0 2815M 62700
                                               47984
                     20
20
                               2815M 62700
                                               47984
      jos
     ios
                                      68200
                                       5248
                                                3840
                                                3840
                                                                 0.1 0:00.02 ba
```

Explicación y Resumen

El caso de estudio #2 sugiere una visualización de un proceso que asigna 400 MB de memoria y ejecuta un bucle infinito para simular una carga mediante el comando **htop** constante en el sistema.

Para empezar se realiza la creación del archivo en **lenguaje C** para posteriormente ejecutar y visualizar el proceso. La creación se la realizó con el comando: **nano caso_2.c** . Dentro de este archivo se procede a digitar el código que utiliza librerías como: stdlib, unistd y stdio.

Una vez finalizada la codificación se ejecuta el archivo mediante el comando gcc caso_2.c -o caso_2; posterior a esto se diseña el entorno de interfaz de líneas de comandos que facilitará



la visualización de la práctica. El entorno consiste en cuatro interfaces en las cuales cada una

ejecutará un comando/instrucción diferente.

En la segunda interfaz se ejecuta el proceso, con: ./caso_2

Una vez realizado esto, en la tercera interfaz se ejecuta: **ps –a,** en la que se visualiza el PID del proceso caso_2, en este caso: 4126.

En la cuarta interfaz se ejecuta el comando: **pgrep caso_2**, donde aparece específicamente el PID del caso_2 (caso contrario a ps –a donde se listaban todos los procesos en ejecución)

Finalmente en la primera interfaz se ejecuta el comando **htop** para monitorear visualmente el proceso, encontrándose este al final de la interfaz junto con sus características y consumos.

Conclusión

Se ha finalizado la práctica, obteniendo los resultados esperados correctamente. Se ha verificado que el proceso que asigna 400 MB de memoria y ejecuta un bucle infinito para simular una carga constante en el sistema dentro de la terminal visual (generada por el comando **htop**) ocupa 402MB de memoria, 2 MB más de la asignada dentro del programa realizado en C.

Esta práctica no solo familiariza al estudiante a retener comandos básicos de Linux como: nano, htop, ls, ps –a, pgrep, etc. Y por otro lado se visualiza el consumo de memoria que tiene un proceso y como se visualiza en la terminal **htop** en Linux. Entendiendo de esta manera la forma en la que funciona la gestión de memoria en un sistema operativo.