PROGRAMACIÓN AVANZADA

Tecnológico de Monterrey, Campus Querétaro

Actividad colaborativa - Manejo de señales

Escribe un programa llamado execute que recibe como parámetros de entrada el nombre de tres archivos de texto. Cada uno de los archivos de texto contendrá la instrucción a ejecutar en el siguiente formato:

```
ls, ls -l -a -R
```

Separado por coma, aparecerá el comando a ejecutar, seguido de la instrucción completa a ejecutar. La instrucción deberá ejecutar vía algún comando `exec`. La forma de ejecución es:

```
$ .\execute file1 file2 file3
```

El programa deberá mandar a ejecutar las órdenes de la siguiente manera:

1. Si recibe la señal `SIGUSR1`, ejecutará la orden 1 (que se encuentra en el archivo `file1`).
2. Si recibe la señal `SIGUSR2`, ejecutará la orden 2 (que se encuentra en el archivo `file2`).
3. Si recibe la señal `SIGPWR`, ejecutará la orden 3 (que se encuentra en el archivo `file3`).

IMPORTANTE: El proceso execute solo termina hasta recibir la señal `SIGINT` (CTRL-C). El programa debe liberar correctamente toda la memoria dinámica que utilice.

Ejemplos de uso:

```
$ .\execute
usage: \execute file1 file2 file3
--------------------------------------------------------------------------
$ .\execute noexiste1 file2.txt file3.txt
.\execute: the file noexiste1 does not exist
--------------------------------------------------------------------------
$ .\execute file1.txt noexiste2 file3.txt
.\execute: the file noexiste2 does not exist
--------------------------------------------------------------------------
$ .\execute file1.txt file2.txt noexiste3
.\execute: the file noexiste3 does not exist
--------------------------------------------------------------------------
$ .\execute empty.txt file2.txt file3.txt
.\execute: the file empty1.txt is empty
--------------------------------------------------------------------------
$ .\execute file1.txt empty.txt file3.txt
.\execute: the file empty2.txt is empty
--------------------------------------------------------------------------
```

```
$ .\execute file1.txt file2.txt empty.txt
.\execute: the file empty3.txt is empty
--------------------------------------------------------------------------
$ ./execute file1.txt file2.txt file3.txt &
[1] 17614
Waiting for a signal...

$ kill -s SIGUSR1 17614
empty.txt  file1.txt  file3.txt execute file2.txt  solution.c
$ ps -f
UID        PID  PPID  C STIME TTY          TIME CMD
pperezm  17614 27703  0 17:02 pts/2    00:00:00 ./execute file1.txt file2.txt fi
pperezm  17652 27703  0 17:03 pts/2    00:00:00 ps -f
pperezm  27703 27694  0 12:19 pts/2    00:00:06 bash

$ kill -s SIGUSR2 17614
PID TTY          TIME CMD
17614 pts/2    00:00:00 execute
17656 pts/2    00:00:00 ps
27703 pts/2    00:00:06 bash
$ ps -f
UID        PID  PPID  C STIME TTY          TIME CMD
pperezm  17614 27703  0 17:02 pts/2    00:00:00 ./execute file1.txt file2.txt fi
pperezm  17657 27703  0 17:03 pts/2    00:00:00 ps -f
pperezm  27703 27694  0 12:19 pts/2    00:00:06 bash

$ kill -s SIGPWR 17614
Linux
$ ps -f
UID        PID  PPID  C STIME TTY          TIME CMD
pperezm  17614 27703  0 17:02 pts/2    00:00:00 ./execute file1.txt file2.txt fi
pperezm  17670 27703  0 17:03 pts/2    00:00:00 ps -f
pperezm  27703 27694  0 12:19 pts/2    00:00:06 bash

$ kill -s SIGUSR2 17614
PID TTY          TIME CMD
17614 pts/2    00:00:00 execute
17675 pts/2    00:00:00 ps
27703 pts/2    00:00:06 bash

$ kill -s SIGUSR1 17614
empty.txt  file1.txt  file3.txt execute file2.txt  solution.c

$ ps -f
UID        PID  PPID  C STIME TTY          TIME CMD
pperezm  17614 27703  0 17:02 pts/2    00:00:00 ./execute file1.txt file2.txt fi
pperezm  17677 27703  0 17:03 pts/2    00:00:00 ps -f
pperezm  27703 27694  0 12:19 pts/2    00:00:06 bash
```

```
$ kill -s SIGINT 17614

Ending...

$ ps -f
UID        PID  PPID  C STIME TTY          TIME CMD
pperezm  17690 27703  0 17:03 pts/2    00:00:00 ps -f
pperezm  27703 27694  0 12:19 pts/2    00:00:06 bash
[1]+  Hecho                  ./execute file1.txt file2.txt file3.txt
```

Rúbrica de evaluación:

Ponderación	
+10 puntos	Verifica que el programa reciba la cantidad correcta de parámetros. En caso de que no sea así, el programa despliega un mensaje adecuado y termina, regresando -1 como resultado de su ejecución.
+10 puntos	Verifica que los archivos de entrada existan. En caso de que no sea así, el programa despliega un mensaje adecuado y termina, regresando -2 para el `file1`, -3 para el `file2` y -4 para el `file3` como resultado de su ejecución.
+10 puntos	Verifica que los archivos de entrada contengan información (size <>0). En caso de que no sea así, el programa despliega un mensaje adecuado y termina, regresando -5 para el `file1`, -6 para el `file2` y -7 para el `file3` como resultado de su ejecución.
+15 puntos	Al recibir `SIGUSR1` ejecuta la instrucción 1, pero el programa principal no termina.
+15 puntos	Al recibir `SIGUSR2` ejecuta la instrucción 2, pero el programa principal no termina.
+15 puntos	Al recibir `SIGPWR` ejecuta la instrucción 3, pero el programa principal no termina.
+15 puntos	Solo termina hasta recibir `SIGINT`, eliminan correctamente la memoria dinámica asignada.