**Tarea Programada #1**

Programar en Linux los siguientes ejercicios de programación en C++, de forma individual, los ejercicios deben comprimirse en un rar y subirlos al aula virtual, cada ejercicio debe indicar ya sea en un manual o en su documentación como se debe ejecutar y con que valores se deben probar para que tenga éxito la ejecución.

Todas los ejercicios deben ser resistentes a fallos.

1. Escriba un programa que calcule la tabla de temperaturas Fahrenheit y sus equivalentes Celsius. Desde 0 a 300 grados F, en incrementos de 20 grados. Utilice la fórmula C = (5/9) \* (F - 32)

Asegúrese de imprimir los datos con tabulaciones y utilice un encabezado.

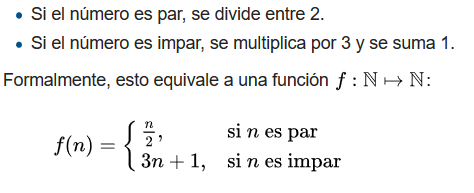
1. Escriba un programa que cuente la cantidad de palabras de una entrada dada por el usuario. Utilice la secuencia CTRL+C para indicar que se terminaron de digitar los datos - también conocido como EOF - En Linux la secuencia es CTRL+D. Revise la función getchar() de stdio.h)
2. Escriba un programa que elimine los espacios en blanco y tabuladores de una entrada dada por el usuario. Utilice memoria dinámica para almacenar en un buffer el texto resultante (sin espacios en blanco ni tabulaciones) para imprimirlo en pantalla. Básese en el ejercicio #2.
3. Escriba un programa para revisar los errores de sintaxis rudimentarios de un programa en C: paréntesis, llaves y corchetes no alineados. El programa debe leer un archivo .c de entrada, y la salida debe ser el(los) número de línea(s) donde existen errores. Si el archivo no contiene errores, entonces la salida no muestra nada.
4. Escriba una versión simplificada de la función "sort" de UNIX. Utilice un array de apuntadores a cadenas de caracteres y quicksort para ordenar. El flujo del programa es:

Lea la entrada del usuario (utilice la técnica del ejercicio 2, recuerde que ENTER no termina, simplemente crea una nueva línea). Cada línea se almacena en el array de apuntadores a cadenas (es decir, cada puntero del array referencia a cada línea dada por el usuario)

Se utiliza la función quicksort para ordenar las líneas. En este punto, note que es más fácil intercambiar los punteros, que mover cada línea

Imprima el resultado

1. La conjetura de Collatz es un problema que todavía no se ha podido demostrar. La situación consiste en tomar cualquier número entero positivo y aplicarle la siguiente fórmula:



El resultado de la conjetura siempre va a ser 1. Más información:

<https://es.wikipedia.org/wiki/Conjetura_de_Collatz>

Escriba un programa en C que utilice la llamada al sistema fork() o createprocess(). El número inicial se debe digitar desde la línea de comandos (a manera de un argumento cuando se ejecuta el programa). Luego, el proceso hijo debe calcular la conjetura de Collatz e imprimirla en pantalla. Por ejemplo, suponga que el programa se ejecuta en Linux de la siguiente manera:

prompt$ ./collatz 8

La salida del programa debe ser:

8, 4, 2, 1

Debe forzar al proceso padre que espere a que el hijo termine su ejecución. Asegúrese de realizar la comprobación de errores con el número digitado en la línea de comandos, también debe revisar que la cantidad de argumentos sea la esperada (por ejemplo, llamar al programa sin argumentos causa un error).

1. Escriba un programa en C que permita crear N procesos hijos e imprima el PID y el PPID de cada uno de los procesos hijos y además del proceso padre. La cantidad de procesos hijos que se deben crear se toma como un argumento desde la línea de comandos. Por ejemplo, invocar al programa de la siguiente manera:

prompt$ ./phijos 3

debe mostrar en pantalla lo siguiente:

Hijo #1: PID = AAAA, PPID = XXXX

Hijo #2: PID = BBBB, PPID = XXXX

Hijo #3: PID = CCCC, PPID = XXXX

Note que el PPID de cada hijo debe ser idéntico (al ser el mismo proceso padre). Debe forzar al proceso padre que espere a que el hijo termine su ejecución. Asegúrese de realizar la comprobación de errores con el número digitado en la línea de comandos, también debe revisar que la cantidad de argumentos sea la esperada (por ejemplo, llamar al programa sin argumentos causa un error).

1. Realice una aplicación para ejemplificar el uso de Mutex en Pthreads. Simular el saldo de una cuenta bancaria luego de realizar operaciones de depósito, con un máximo de 10 depósitos simultáneos, debe dar el resultado del balance final de la cuenta.
2. Crear una aplicación para calcular la secuencia de Fibonacci utilizando hilos. El hilo padre recibe por parámetro el numero para generar la secuencia, mientras que el otro hilo realiza el calculo de la secuencia el resultado se almacena en un array compartido entre todos los hilos.
3. Desarrollar un programa que muestra el uso de tuberías (pipes) para enviar información entre el proceso padre y el proceso hijo. El texto que se va a enviar se envía por parámetro al programa.

**Evaluación**

|  |  |
| --- | --- |
| Pregunta | Escala |
| 1 | Menor |
| 2 | Menor |
| 3 | Menor |
| 4 | Menor |
| 5 | Menor |
| 6 | Mayor |
| 7 | Mayor |
| 8 | Mayor |
| 9 | Mayor |
| 10 | Mayor |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Escala | Valores | | | | |
| Excelente | Muy Bueno | Bueno | Regular | Deficiente |
| Menor | 5 | 4 | 3 | 2 | 1-0 |
| Mayor | 10 | 9-7 | 6-5 | 4-3 | 2-0 |

|  |  |
| --- | --- |
| Valor | Significado |
| Excelente | El programa se ejecuta perfectamente, realizando todo lo que se le solicita. |
| Muy bueno | El programa se ejecuta perfectamente, realizando casi todo lo que se le solicita, sin errores. |
| Bueno | El programa se ejecuta, tiene algunos errores y realiza algunas cosas que se solicitan |
| Regular | El programa no se ejecuta, el código intenta hacer algo como lo que se les solicita. |
| Deficiente | El programa no se ejecuta, no realiza en código nada que indique se esta intentando hacer lo solicitado. |