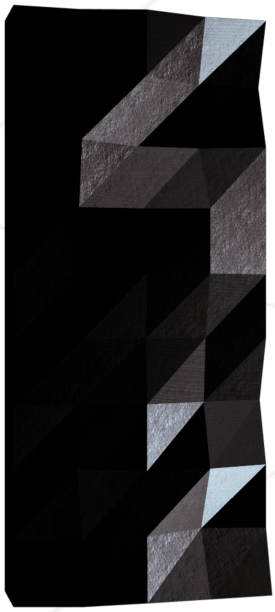

Minitalk

Resumen: El propósito de este proyecto es programar un pequeño programa de intercambio de datos utilizando señales UNIX.

Índice general

Capítulo I

Avance

cis-3-Hexen-1-ol, también conocido como (Z)-3-hexen-1-ol y alcohol de hojas, es un líquido aceitoso incoloro con un olor intenso a hierba y hojas verdes recién cortadas.
Se produce en pequeñas cantidades por la mayoría de las plantas y actúa como un atrayente para muchos insectos depredadores.
cis-3-hexen-1-ol es un compuesto de aroma muy importante que se utiliza en sabores de frutas y verduras y en perfumes. La producción anual es de alrededor de 30 toneladas.

Capítulo II

Reglas comunes

- Su proyecto debe estar programado respetando la Norma. Si tiene archivos o funciones extras, entrarán dentro de la verificación de la norma y, como haya algún error de norma, tendrá un 0 en el proyecto.
- Sus funciones no pueden pararse de forma inesperada (segmentation fault, bus error, double free, etc.) salvo en el caso de un comportamiento indefinido. Si esto ocurre, se considerará que su proyecto no es funcional y tendrá un 0 en el proyecto.
- Cualquier memoria reservada en el montón (heap) tendrá que ser liberada cuando sea necesario. No se tolerará ninguna fuga de memoria.
- Si el proyecto lo requiere, tendrá que entregar un Makefile que compilará sus códigos fuente para crear la salida solicitada, utilizando los flags `-Wall`, `-Wextra` y `-Werror`. Su Makefile no debe hacer relink.
- Si el proyecto requiere un Makefile, su Makefile debe incluir al menos las reglas `$(NAME)`, `all`, `clean`, `fclean` y `re`.
- Para entregar los extras, debe incluir en su Makefile una regla `bonus` que añadirá los headers, bibliotecas o funciones que no estén permitidos en la parte principal del proyecto. Los extras deben estar dentro de un archivo `_bonus.{c/h}`. Las evaluaciones de la parte obligatoria y de la parte extra se hacen por separado.
- Si el proyecto autoriza su `libft`, debe copiar sus códigos fuente y y su Makefile asociado en un directorio libft, dentro de la raíz. El Makefile de su proyecto debe compilar la biblioteca con la ayuda de su Makefile y después compilar el proyecto.
- Le recomendamos que cree programas de prueba para su proyecto, aunque ese trabajo **no será ni entregado ni evaluado**. Esto le dará la oportunidad de probar fácilmente su trabajo al igual que el de sus compañeros.
- Deberá entregar su trabajo en el git que se le ha asignado. Solo se evaluará el trabajo que se suba al git. Si Deepthought debe corregir su trabajo, lo hará al final de las evaluaciones por sus pares. Si surge un error durante la evaluación Deepthought, esta última se parará.
- Los ejecutables deben llamarse `client` y `server`.

- Debes gestionar los errores con cuidado. Bajo ninguna circunstancia tu programa puede terminar inesperadamente (segfault, bus error, double free, etc).
- Tu programa no puede tener leaks de memoria.
- En la parte obligatoria se te permite utilizar las siguientes funciones:
  - `write`
  - `signal`
  - `sigemptyset`
  - `sigaddset`
  - `sigaction`
  - `kill`
  - `getpid`
  - `malloc`
  - `free`
  - `pause`
  - `sleep`
  - `usleep`
  - `exit`

Capítulo III

Parte obligatoria

- Debes crear un programa de comunicación de la forma cliente-servidor.
- El servidor debe lanzarse primero, tras lanzarse debe mostrar su PID.
- El cliente tomará como parámetros:
  - El PID del servidor.
  - La string que debería mandarse.
- El cliente debe comunicar la string pasada como parámetro al servidor. Una vez la string se haya recibido, el servidor debe mostrarla.
- La comunicación entre tus programas debe hacerse SOLO utilizando señales UNIX.
- El servidor debe ser capaz de mostrar la string suficientemente rápido. Por rápido queremos decir que si piensas que es muy larga, probablemente lo sea.
- Tu servidor debe poder recibir strings de distintos clientes consecutivamente, sin necesidad de reiniciar.
- Solo puedes utilizar estas dos señales: `SIGUSR1` y `SIGUSR2`.

`Una string de 100 caracteres en 1 segundo es COLOSAL`

Capítulo IV

Extras

- Añade un pequeño sistema de acknowledgment.
- Soporta los caracteres Unicode.

Capítulo V

Entrega y evaluación de pares

Como es habitual, entrega tu trabajo en el repositorio `Git`. Solo el trabajo en tu repositorio será evaluado.