Objetivos

Objetivo general

Resolver un problema básico de gestión de procesos que sirva para que el alumno conozca y maneje las primitivas POSIX para llevar a cabo dicha gestión.

Objetivos específicos

El alumno deberá adquirir las habilidades necesarias para la creación y espera de procesos, así como el correcto tratamiento de las señales básicas relativas a la finalización de la aplicación.

Implementar la solución planteada mediante las primitivas POSIX para la gestión de procesos, haciendo uso de la funcionalidad necesaria para resolver el problema dependiente del dominio planteado.

Especificación

Construya un sistema compuesto por tres ejecutables que simule el funcionamiento que se detalla a continuación. El sistema contará con tres tipos de procesos:

- ***Manager***: este proceso será responsable de crear un número determinado de procesos de tipo *procesador* y de tipo *contador*, gestionando de manera adecuada su finalización y liberando los recursos previamente reservados. Este proceso recibirá por la línea de órdenes la ruta de dos ficheros.
  - <archivo_texto>: será abierto por el proceso *manager*.
    - Leerá su contenido línea a línea.
    - Por cada línea creará un proceso *contador*.
  - <archivo_patrones>: este será procesado por el proceso *manager.*
    - Leerá su contenido palabra a palabra.
    - Por cada palabra insertará un nodo en una lista empleando la estructura de datos creada en la práctica *P1.1.*[1].
    - Procesando dicha lista, creará un proceso *procesador* por cada nodo de la misma.

- ***Procesador***: estos procesos recibirán en el momento de su creación la ruta del fichero <archivo_texto> y un patrón. Su función consistirá en comprobar si este patrón se corresponde con alguna de las palabras que conforman el texto contenido en el fichero <archivo_texto>.

- ***Contador***: estos procesos recibirán en el momento de su creación un número de línea y una línea. Su función consistirá en contar el número de palabras que conforman la línea recibida.

[1]Las estructuras de datos *TLista* y *Nodo* implementadas en la práctica *P1.1.* deberán ser debidamente modificadas para que puedan trabajar con cadenas (inicialmente fueron diseñadas para trabajar con enteros). **OJO con los punteros**.

Las rutas de los archivos <archivo_texto> y <archivo_patrones> serán indicados por el usuario a través de la línea de órdenes al ejecutar el único proceso de tipo *manager*:

```
./exec/manager <archivo_texto> <archivo_patrones>
```

A continuación, se muestra un ejemplo de <archivo_texto> y de <archivo_patrones>.

```
Pablito clavó un clavito en la calva de un calvito
Un clavito clavó Pablito en la calva de un calvito
          ¿Qué clavito clavó Pablito?
```

Figura 1: Ejemplo <archivo_texto>

```
Pablito clavito calvito
```

Figura 2: Ejemplo <archivo_patrones>

La finalización de la simulación tendrá lugar si se cumple una de las dos condiciones siguientes:

- Todos los procesos de tipo *procesador* y *contador* finalizan su ejecución. El proceso *manager*, tras detectar esta situación, liberará recursos.
- El usuario pulsa la combinación de teclas *Ctrl + C*. El proceso *manager*, tras detectar este evento, enviará una señal de finalización a todos los procesos de tipo *procesador* y *contador* que estén en ejecución y liberará recursos.

Ejemplo de ejecución

Una vez resuelto el ejercicio, si se ejecuta el proceso *manager* con los siguientes argumentos (*make test*),

```
./exec/manager data/solution.txt data/patrones.txt
```

el resultado debe ser similar al siguiente (cambiará el PID de los procesos *procesador* y *contador*, el orden de impresión y los valores generados de manera aleatoria):

```
[MANAGER] 6 procesos creados.
[PROCESADOR 14483] Patron 'Pablito' encontrado en linea 1
[PROCESADOR 14483] Patron 'Pablito' encontrado en linea 2
[PROCESADOR 14484] Patron 'clavito' encontrado en linea 1
[PROCESADOR 14484] Patron 'clavito' encontrado en linea 2
[PROCESADOR 14484] Patron 'clavito' encontrado en linea 3
[CONTADOR 14480] La linea '0' tiene 10 palabras
[CONTADOR 14481] La linea '1' tiene 10 palabras
[CONTADOR 14482] La linea '2' tiene 4 palabras
[PROCESADOR 14485] Patron 'calvito' encontrado en linea 1
[PROCESADOR 14485] Patron 'calvito' encontrado en linea 2
[MANAGER] Proceso CONTADOR terminado [14480]...
[MANAGER] Proceso CONTADOR terminado [14481]...
[MANAGER] Proceso CONTADOR terminado [14482]...
[MANAGER] Proceso PROCESADOR terminado [14483]...
[MANAGER] Proceso PROCESADOR terminado [14484]...
[MANAGER] Proceso PROCESADOR terminado [14485]...

[MANAGER] Terminacion del programa (todos los procesos terminados).
```

Sugerencias

Apóyese en las funciones estándar del lenguaje C para la manipulación de archivos. Estudie la función *strtok()*, incluida en <*string.h*>, para llevar a cabo el procesamiento a nivel de línea.

Bibliografía básica

- Kernighan B. W., Ritchie D. M. - *El lenguaje de programación C* - Ed. Prentice Hall 1991.
- Páginas del manual electrónico de UNIX.
- Rochkind, M.J. - *Advanced Unix Programming (2 nd Edition)* - Ed. Addison-Wesley 2004.
- Robbins, K.A., Robbins, S – *UNIX Programación Práctica* – Ed. Prentice Hall 1997.

Plazos de entrega

Esta práctica deberá:

- Entregarse como máximo el día 21 de marzo del 2023 a las 23:59 horas.
- No se tendrán en cuenta las prácticas entregadas fuera de plazo.
- No se admitirán prácticas entregadas por email.
- La entrega de la práctica tendrá que realizarse en formato electrónico.
- La entrega será mediante una tarea en el espacio virtual de la asignatura en *Campus Virtual*. Si algún alumno no está registrado en dicha plataforma, deberá informar al profesor de prácticas para que le añada al curso.
- El fichero comprimido subido al sistema deberá cumplir la especificación que se indica en la sección Normas de Presentación.

Normas de presentación

El seguimiento de las normas de presentación de la práctica es **obligatorio**, especialmente la parte relativa a la compilación automática mediante *make* de todos los ejecutables necesarios para la resolución de la misma. Obviamente, las rutas del fichero *makefile* deberán ser relativas para permitir su compilación en diferentes computadores sin necesidad de realizar ningún cambio. En este curso únicamente será posible entregar la práctica mediante *Campus Virtual*, es decir, **no hay que realizar ninguna impresión** de la misma.

Únicamente se subirá un fichero comprimido en formato zip por un solo miembro de la pareja. Debe cumplir el siguiente patrón de nombrado:

primerApellido1_segundoApellido1_alumno1_primerApellido2_segundoApellido2_alumno2.zip

Al descomprimir el fichero, se deberá obtener la siguiente estructura de directorios:

/include	Contendrá los ficheros de cabecera .h
/src	Contendrá los ficheros fuente .c
/obj	Contendrá los ficheros objeto (se deberán crear mediante make).
/exec	Contendrá los ejecutables (se deberán crear mediante make).
/data	Contendrá los ficheros para ejecutar las pruebas.
/doc	Contendrá un fichero .pdf con la memoria de la práctica.
/	El directorio raíz contendrá únicamente dos ficheros: • Un archivo makefile que deberá permitir compilar todos los programas relacionados con la práctica mediante la llamada directa a make. • Un fichero ascii (author.txt) que contendrá nombre y apellidos de los autores.

La **memoria** de la práctica debe incluir una discusión de la solución planteada, haciendo hincapié en aquellas partes del código fuente que considere más relevante. Así mismo, la memoria ha de incluir una sección en la que se indique brevemente cómo compilar y ejecutar.

Evaluación

Se valorarán dos aspectos: funcionalidad y calidad. La funcionalidad valora que se resuelva el problema conforme a sus especificaciones y que sea consistente, es decir, funcione adecuadamente en todas y cada una de las ejecuciones. Si solo funciona a veces, se considera que no funciona. Si existen casos no contemplados o faltan mecanismos de gestión de errores, la puntuación se verá afectada negativamente, aunque el programa nunca falle. En cuanto a la calidad del código, se valoran los siguientes aspectos:

- Estructura y organización del código, mediante clases, funciones o métodos.
- Elección de nombres significativos para funciones, variables, etc.
- Evitar uso de variables globales innecesarias.
- Evitar anidación injustificada de estructuras de control.
- Evitar uso incorrecto de tipos de datos, código redundante, código muerto, etc.
- Evitar bloques (funciones/métodos) demasiado largos.

La **valoración de la calidad es negativa**, es decir, restará sobre la nota final obtenida en funcionalidad. Esto implica que una práctica completamente funcional puede no obtener la máxima nota. Es importante destacar que el alumno **debe ser capaz de defender su trabajo**, por lo que en cualquier se le puede **solicitar que defienda el trabajo** presentado.

Penalizaciones

Se contemplan 2 motivos por los que la puntuación total de la actividad puede sufrir penalizaciones:

- Errores de sintaxis que impiden ejecutar el programa.
- Incumplir otras condiciones del enunciado.

En el caso de errores de sintaxis, si supone cambios menores, se pedirá al alumno presentar una versión corregida.

Condiciones

El alumno debe tener especial cuidado con la **incorporación de código ajeno** en su programa. Está PERMITIDO siempre que se cumplan las siguientes condiciones:

- Los fragmentos copiados y su autor están claramente identificados.
- El alumno comprende y es capaz de explicar con TODO DETALLE la funcionalidad del código copiado.
- El autor del código original permite la copia mediante la licencia correspondiente y esta aparece claramente junto al código original.
- El autor del original mantiene su código públicamente accesible (y el alumno proporciona el enlace en forma de comentario).
- El autor del original no es ni ha sido alumno de la UCLM.

Infringir cualquiera de estas normas al utilizar código ajeno se considerará plagio. Cometer plagio implica una calificación de 0 en la actividad *«Realización de prácticas de laboratorio»* según el artículo 9 de la *«Guía de Evaluación del Estudiante»*:

> *Art. 9. Realización fraudulenta de pruebas de evaluación.*
> *1. La constatación de la realización fraudulenta de una prueba de evaluación o el incumplimiento de las instrucciones fijadas para la realización de la prueba dará lugar a la calificación de suspenso (con calificación numérica de 0) en dicha prueba. En el caso particular de las pruebas finales, el suspenso se extenderá a la convocatoria correspondiente.*