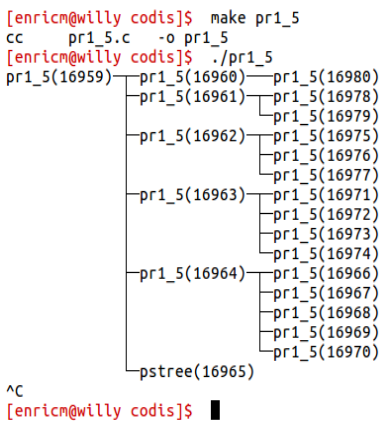


- A) **[0,5 puntos]** Cread un **Makefile** que permita generar todos los programas del enunciado a la vez y cada uno de ellos por separado. Añadid una regla (clean) para borrar todos los binarios y/o ficheros objeto y dejar solo los ficheros fuentes. Los programas ejecutables deben generarse si, y sólo si, ha habido cambios en los ficheros fuentes.
- B) **[0,5 puntos]** Todos los programas a desarrollar en este control deben implementar la función `Usage()` así como realizar el tratamiento de errores en las llamadas al sistema.
- C) **[3,0 puntos]** Escribid un programa (`pr1_5.c`) que cree cinco procesos, donde a su vez el primer proceso creado creará un hijo, el segundo dos, ..., y el quinto cinco. A continuación el proceso inicial esperará un segundo (utilizad la rutina `sleep(1)`) y creará un proceso hijo que ejecutará la orden `pstree` con tres parámetros: `-c`, `-p` y el pid del proceso inicial. Finalmente, todos los procesos ejecutarán la llamada al sistema `pause()` para bloquearlos indefinidamente. Para acabar la ejecución, deberéis pulsar Ctrl C. Como referencia, a la derecha se muestra el resultado que se espera obtener.
- 
- ```

[enricm@willy codis]$ make pr1_5
cc pr1_5.c -o pr1_5
[enricm@willy codis]$./pr1_5
pr1_5(16959)
├── pr1_5(16960)
│ ├── pr1_5(16961)
│ │ ├── pr1_5(16962)
│ │ │ ├── pr1_5(16963)
│ │ │ │ ├── pr1_5(16964)
│ │ │ │ │ ├── pr1_5(16965)
│ │ │ │ │ ├── pr1_5(16966)
│ │ │ │ │ ├── pr1_5(16967)
│ │ │ │ │ ├── pr1_5(16968)
│ │ │ │ │ └── pr1_5(16969)
│ │ │ └── pr1_5(16970)
│ │ └── pr1_5(16971)
│ └── pr1_5(16972)
├── pr1_5(16973)
├── pr1_5(16974)
├── pr1_5(16975)
└── pr1_5(16976)

```
- D) **[1,0 punto]** Escribid una nueva versión del programa anterior (llamadlo `pr1_5b.c`) en el que substituyáis las invocaciones a la llamada al sistema `pause()` por invocaciones a `sigsuspend()`. En el proceso padre, el signal `SIGINT` deberá estar bloqueado hasta que la ejecución de `pstree` haya finalizado. Los procesos hijos deberán crearse con el signal `SIGINT` ya bloqueado y deberán desbloquear el signal en el momento de hacer el `sigssuspend`. Para todos los procesos, el resto de signals estará siempre bloqueado.
- E) **[3,0 puntos]** Escribid un programa (`comm.c`) que espere como parámetro el nombre de una pipe con nombre. En caso que no exista, el programa la creará. A continuación, el proceso inicial creará un hijo que mutará al programa `date` para escribir en la named pipe la fecha y hora actual. Cuando el hijo haya acabado, el proceso inicial debe indicar por su salida estándar cuántos caracteres ha escrito el hijo en la named pipe.
- F) **[2,0 puntos]** Escribid un programa (`grepN.c`) que espere como parámetros, y en este orden, un número natural ( $N > 0$ ), un carácter (`c`) y el nombre de un fichero ya existente (si no existe, el programa deberá mostrar un mensaje de error y finalizar inmediatamente). El programa debe indicar por su salida estándar cuántas veces aparece el carácter `c` en el contenido del fichero. Para leer el fichero debe utilizarse un buffer de `N` bytes que haya sido alcatado dinámicamente.

### Qué se tiene que hacer

- El Makefile
- El código de los programas en C con la función `Usage()` en cada programa

### Qué se valora

- Que sigáis las especificaciones del enunciado
- Que el uso de las llamadas a sistema sea el correcto y se comprueben los errores de **todas** las llamadas al sistema

- Código claro y correctamente indentado
- Que el Makefile tenga bien definidas las dependencias y los objetivos
- La función Usage() que muestre cómo debe invocarse correctamente al programa en caso que los argumentos recibidos no sean adecuados.

### Qué hay que entregar

**En este examen habrá dos entregas.** Una a los 45 minutos del examen (Entrega-MITAD), en la que tendrás que entregar todo lo que hayas hecho hasta ese instante, y otra al final del examen (Entrega-FINAL), que se corresponde a la entrega final del examen.

NOTA: Se considera entrega completa sólo si se han hecho las dos entregas.

Cada entrega consistirá en un único fichero tar.gz con los \*.c y el Makefile:

```
tar zcvf clab2.tar.gz Makefile *.c
```