Escuela Politécnica Superior de Elche

GRADO EN INGENIERÍA INFORMÁTICA EN TECNOLOGÍAS DE LA INFORMACIÓN

Sistemas Operativos

Examen Práctico – Ordinaria 1er. Semestre – 12 Enero 2016

Se pide realizar dos programas, uno en bash (llamado partebash) y otro en C (llamado partec)

Puntuación sobre 10:

partebash: 5 puntos.

partec: 5 puntos

Podéis usar internet para consultar lo que necesitéis,
y copiar de vuestro directorio home todo lo que necesitéis.

Aviso importante:

Las cuentas están auditadas.
Si usáis el correo electrónico (webmail) o cualquier aplicación de comunicación, almacenamiento cloud o mensajería automáticamente el alumno estará suspendido.

En vuestro directorio home tenéis una carpeta que se llama **examen**.

Dentro de esta carpeta existe:

A) Un fichero llamado **quijote.txt** con un extracto del Quijote de Cervantes. No tocar.

B) Un ejecutable llamado **crypt** (será llamado por otros, no tenéis que hacer nada con el)

C) Un esqueleto (**partebash**) para la parte bash que ya tiene una función desarrollada.

D) Dos scripts que ya están programados y que sólo tenéis que usarlos en vuestro **partebash**. Son:

- Un script llamado **genpass** que recibe un número como parámetro y genera una cadena aleatoria numérica de dicha longitud.
- Un script llamado **encriptar** que llamará a **crypt** y que recibe dos parámetros:
  a) el nombre de un fichero
  b) un password numérico

  El script **encriptar** abre el fichero y encripta su contenido en base al password que recibe. Genera un fichero encriptado oculto llamado **.crypt** .

  Por ejemplo si llamamos a

  ```
  $ encriptar quijote.txt 521
  ```

  generará un fichero llamado .crypt cuyo contenido ha sido encriptado con el password 521

E) Una serie de ficheros que componen un esqueleto para la parte C.

- **partec.c** : fichero main para la parte c que debéis desarrollar (prototipo).
- **functions.c** : fichero donde ubicar funciones nuevas si las necesitáis. Ya contiene algunas que deberéis usar.
- **functions.h**: fichero de cabecera donde ubicar los prototipos de las funciones que defináis, ya contiene algunas.
- **sem.h y sem.c** : La librería de semáforos (ya completa no tocar)

F) Un **makefile** ya completo para que podáis compilar rápidamente la parte C.
Basta con ejecutar **make** para compilar.

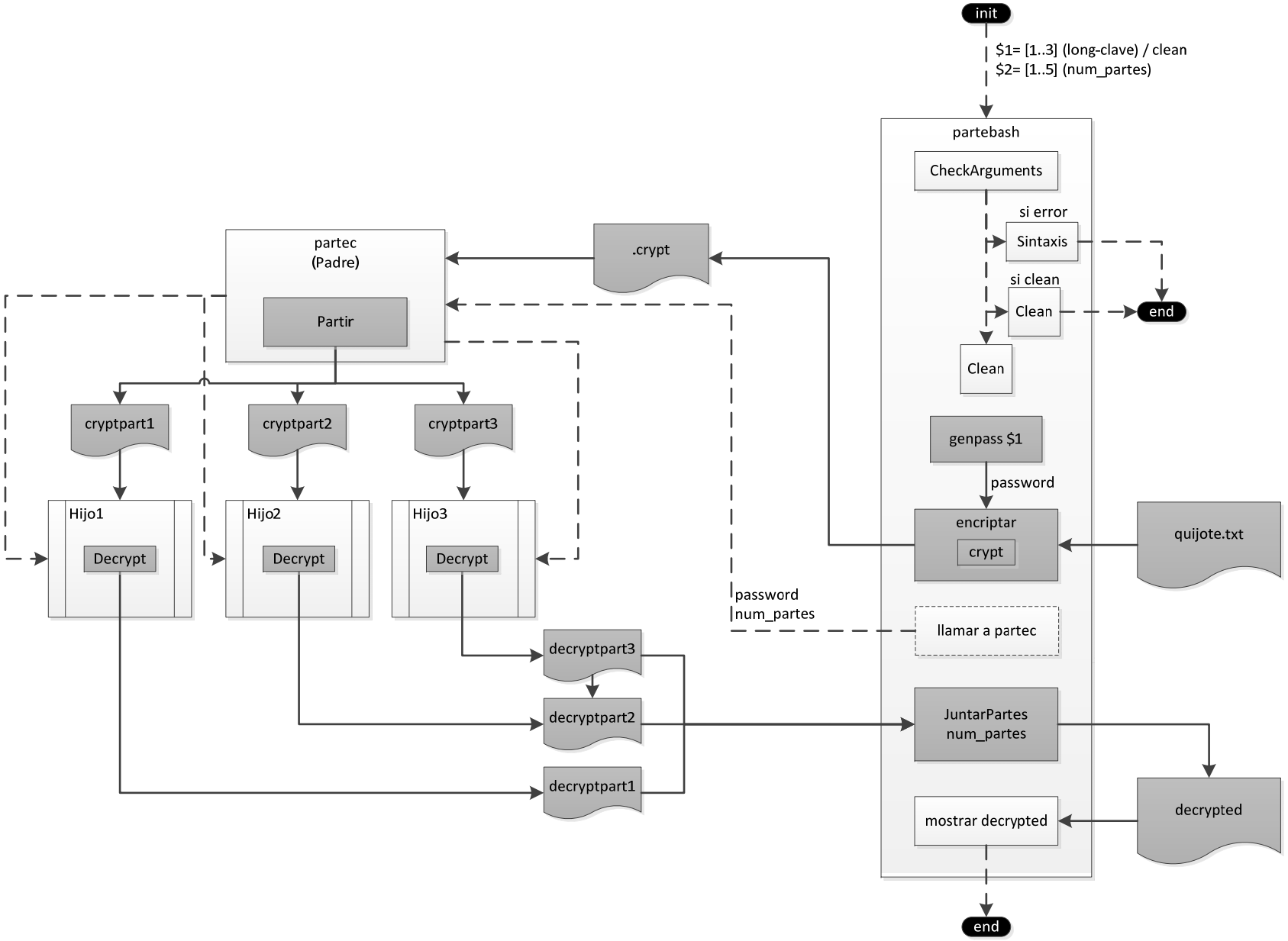
G) Un fichero **partecp** ya compilado que podéis usar para probar la **partebash**.

Escuela Politécnica Superior de Elche

GRADO EN INGENIERÍA INFORMÁTICA EN TECNOLOGÍAS DE LA INFORMACIÓN

Sistemas Operativos

Examen Práctico – Ordinaria 1er. Semestre – 12 Enero 2016



Escuela Politécnica Superior de Elche

GRADO EN INGENIERÍA INFORMÁTICA EN TECNOLOGÍAS DE LA INFORMACIÓN

Sistemas Operativos

Examen Práctico – Ordinaria 1er. Semestre – 12 Enero 2016

SE PIDE:

PARTE BASH (1.5 ptos.)

Programar un shell script llamado **partebash** y hacerlo ejecutable. (Disponéis de un esqueleto en el directorio examen).

partebash recibirá como parámetros dos números A y B que deben cumplir los siguientes requisitos:

- Sólo puede recibir dos parámetros.
- El primer parámetro podrá ser la palabra "clean" o bien la longitud del password, un numérico A entre 3 <= A <= 6
- El segundo parámetro son las partes en las que se trocea y será un numérico B entre 1 <= B <= 5
- Si el primer parámetro es la palabra clean, no se tiene en cuenta el segundo parámetro (si lo hubiera) y se llamará a Clean (función que tendréis que implementar).

El programa deberá comprobar los requisitos y mostrar un mensaje de error indicando el motivo del error.

Si no hay error en los parámetros y el primer parámetro es **clean**, deberéis llamar a la función Clean que debéis implementar:

Clean() borrará el fichero **.crypt**, los ficheros **cryptpartN** y **decryptpartN** y el fichero **decrypted**.

Si no hay error en los parámetros y si se recibe un numérico en el primer parámetro, el programa llamará a **genpass** con el número A para obtener un password (un password de entre 1 y 3 dígitos)

El password generado por **genpass** se utilizará para encriptar el fichero **quijote.txt**, para lo que nuestro script **partebash** llamará a

```
encriptar quijote.txt <password>
```

Esto encriptará el fichero y generará un fichero llamado **.crypt**

Ahora llamaremos a un programa ejecutable (que haremos en la parte C) llamado **partec** pasándole como parámetro el número B (segundo parámetro) y el password, de esta forma:

```
partec password B   ////podéis usar partecp para probar
```

Al terminar el programa **partec** tendremos en el directorio actual, B ficheros llamados **decryptpartN** donde N es el número de parte entre 1 y B

El programa mostrará el contenido de todos los ficheros desencriptados juntos, para lo cual llamará a la función **JuntarPartes()** que generará un fichero llamado **decrypted** con la unión de todos los **decryptpartN**.

Ya sólo tendrá que mostrar el contenido de **decrypted**.

PARTE C (1.5 ptos.)

También disponéis de un esqueleto de **partec** en el directorio examen.

El programa **partec** utilizará la librería de semáforos que se ha empleado en la versión 3 de la práctica y que ya está completa en el prototipo que se os entrega, sólo hay que usarla.

El programa **partec** recibirá dos parámetros:

- Un número de 1 a 3 dígitos. (password)
- Un número con valor entre 1 y 5 (B)

No es necesario comprobar los parámetros, se supone que han sido comprobados y vienen bien de **partebash**

El programa deberá crear tantos hijos como indique el segundo parámetro.

Código Hijos:

Los hijos deberán esperar en un semáforo (uno para cada hijo) a que el padre le de paso después de partir fichero **.crypt** en B partes.

Entonces cada hijo cogerá una parte y la desencriptará llamando a la función **Decrypt()**.

La función **Decrypt()** (que ya está hecha) recibe como parámetro el nombre del fichero **cryptpartN** y el **password**. Esta función generará un fichero llamado **decryptpartN** para cada parte desencriptada.

Cuando un hijo termina deberá hacer **signal** sobre una barrera en la que estará esperando el padre. Una vez hace **signal** sobre la barrera, puede terminar sin necesidad de comunicar nada al padre.

Código Padre:

Después de haber creado a los hijos, el padre llamará a la función **Partir()** pasándole como parámetro el número de partes recibidas en el parámetro B.

La función P**artir()** (que ya está hecha, sólo tenéis que llamarla) genera B partes del fichero **.crypt** llamadas **cryptpartN** siendo **N** el número de parte.

Entonces, se deberá quedar esperando (**wait**) en una barrera.

Cuando todos los hijos hayan terminado de encriptar (hagan **signal**) el padre saldrá de la barrera.

Entonces comprobará que todos sus hijos han terminado antes de poder terminar él (para no dejarlos huérfanos). Para ello ejecutará un bucle esperando la terminación de los hijos

No debe capturar el valor de terminación, no hace falta, simplemente esperar la terminación de todos para poder terminar él.